BUSINESS WEEK
March 14, 1964
Fifty cents
A McGraw-Hill publication

Reprint of the article:

A computer that grows with you

NEW PRODUCTS

A computer that grows with you

Latest machine is designed to serve 128 terminals at once, and to have subsystems added as required. Called the PDP-6, it's made by company that serves mainly scientists

Designer Gordon Bell and Richard Best, chief engineer of Digital Equipment Corp., believe the PDP-6 can outperform computers that cost twice as much.

A very small company this week unveiled a computer that, in some respects, is the biggest ever.

Digital Equipment Corp., of Maynard, Mass., put the machine together. It's called the PDP-6, and it costs $2-million if you want all the available bells and whistles. And these are considerable, particularly its outsized central memory with a capacity of 262,000 words (compared with 64,000 in the larger IBM scientific computers), which, among other things, allows the PDP-6 to serve up to 128 input-output stations simultaneously.

Most computers are designed to work one problem after another, very fast. What's unusual about the PDP-6 is that it is designed to work on a whole lot of problems at once, though at moderate speed. Digital Equipment believes it is the first commercially available computer to offer time-sharing and multi-processing as standard features—a concept that for years has fascinated computer experts at MIT [BW Feb. 1'64,54], as well as DEC's scientist executives.

Maverick. DEC is a maverick of a company that's virtually unknown in business data-processing circles, though its products are about as familiar as Hershey bars to scientists and engineers involved in computer and electronic research and development.

The company was set up seven years ago on a $70,000 shoestring provided by Boston's venture capital group, American Research & Development Corp. DEC turned out to be one of ARDC's more successful investments—it was running in the black within a year, and on net sales of nearly $10-million in the fiscal year ending last June 30, reported net earnings after taxes of $1.1-million.

DEC does about half its business by mail order, selling hundreds of different electronic circuit modules —little grey boxes packed with transistors—that count, switch, amplify, or mix electrical signals. It's hard to walk through a computer or elec-

tronics laboratory without running into the DEC trademark. Since DEC modules are used primarily in designing and testing computers, it was almost inevitable that the company eventually would build computers, too.

By the dozen. Eschewing complicated acronyms like WHIRLWIND or MANIAC, DEC calls its computers "Programmed Data Processors," and they are known as PDP-1 through PDP-6. The PDP-1, introduced in 1960, is a relatively fast machine. It caught on with scientific laboratories in the Boston area due to its low price—$120,000—and because it's relatively simple to add on to it and attach such special devices as cathode-ray tubes that handle graphic as well as numerical information.

Last year, the company brought out the PDP-5, a small-scale computer selling for $27,000. At the Fall Joint Computer Conference, in Las Vegas, the computer was programmed so visitors could try their hand at shooting down a simulated satellite—a moving circle displayed on the computer's display tube. Successful shots were rewarded by a croaking rendition of Taps played by the computer's memory circuits. Inside one cabinet was a hand-lettered sign reading "$218,880 a dozen." The company's quantity discounts start with six units.

The PDP-5 has been selling briskly—by the half dozen—ever since. So has an important accessory that DEC developed to go with it—a very small magnetic tape unit with 5-in. reels called the Micro Tape. This device fills the gap between relatively slow punched paper tape readers and printers and regular high-speed tape units.

No super service. DEC breaks with other computer industry practices. It doesn't lease its equipment. Nor is it anxious to maintain it. It prefers to have customers send their own employees to a computer maintenance training course that DEC offers.

In an industry known for super service, DEC's take-it-or-leave-it approach comes as a breath of fresh air. So does the attitude of the company's two top executives, Pres. Kenneth Olsen and Vice-Pres. Harlan Anderson: After due consideration they decided they wouldn't have their picture taken with their new computer because they don't believe in personal aggrandisement.

Nor do they believe in pell-mell growth: "Our biggest problem is restraint," says Olsen. The company has never needed new equity capital, and Olsen doesn't think he will need it in the future, since he wants DEC to grow slowly enough to maintain his employees' "spirit of camaraderie."

Intimate approach. So far, DEC's customers have been largely scientific laboratories and universities. To some extent, these clients form a loose fraternity that traces its origins back to MIT and Lincoln Laboratory, where Olsen and Anderson also learned their computer savvy.

The company encourages the intimate approach to sales, keeps track of every inquiry, letter, and order on a computer tape, and refers to the name file each time correspondence comes in. DEC does almost no advertising, but puts much effort into its mail-order operation, doing all its own technical writing, printing, and art work in a big print shop. Its entire plant is in part of the former American Woolen Co. textile mill—a century-old, rambling brick complex that's now a rabbit-warren of small companies.

Computer-man's computer. Olsen and Anderson are confident that the PDP-6 will find a market. Says designer Gordon Bell: "We tried to think out what a computer should be, then designed a system that could grow gracefully and easily." He believes the PDP-6 systems will outperform systems that cost twice as much.

The elements of the computer system are built on a modular basis, and are interconnected by a network of high-speed circuits called busses. The design is such that subsystems added to the network do not have to work synchronously; that makes it easier to add other elements as the system grows.

Obviously, the machine is a computer-man's computer, and is aimed at the relatively few customers who feel themselves limited by present commercial equipment. Since the machine was just introduced, DEC claims no sales. All Olsen will say is: "There has been considerable interest on the part of people who are interested in buying more than one."

Wider market. Scientific users make up most of DEC's clients, but this, says Olsen, may not always be so. "As business applications get more sophisticated, the machine that's required differs less from the scientific machines. This way, we might get into business applications."

And perhaps by the time DEC gets big enough to supply a lot more customers, there may be lots more customers sophisticated enough to buy low-cost, highly advanced equipment that comes without much service or software.

Page 1

A computer that grows with you

Latest machine is designed to serve 128 terminals at once, and to have subsystems added as required. Called the PDP-6, it's made by company that serves mainly scientists

Designer Gordon Bell and Richard Best, chief engineer of Digital Equipment Corp., believe the PDP-6 can outperform computers that cost twice as much.

A very small company this week unveiled a computer that, in some respects, is the biggest ever.

Digital Equipment Corp., of Maynard, Mass., put the machine together. It's called the PDP-6, and it costs $2-million if you want all the available bells and whistles. And these are considerable, particularly its outsized central memory with a capacity of 262,000 words (compared with 64,000 in the larger IBM scientific computers), which, among other things, allows the PDP-6 to serve up to 128 input-output stations simultaneously.

Most computers are designed to work one problem after another, very fast. What's unusual about the PDP-6 is that it is designed to work on a whole lot of problems at once, though at moderate speed. Digital Equipment believes it is the first commercially available computer to offer time-sharing and multi-processing as standard features—a concept that for years has fascinated computer experts at MIT [BW Feb. 1'64,54], as well as DEC's scientist executives.

Maverick. DEC is a maverick of a company that's virtually unknown in business data-processing circles, though its products are about as familiar as Hershey bars to scientists and engineers involved in computer and electronic research and development.

The company was set up seven years ago on a $70,000 shoestring provided by Boston's venture capital group, American Research & Development Corp. DEC turned out to be one of ARDC's more successful investments—it was running in the black within a year, and on net sales of nearly $10-million in the fiscal year ending last June 30, reported net earnings after taxes of $1.1-million.

DEC does about half its business by mail order, selling hundreds of different electronic circuit modules —little grey boxes packed with transistors—that count, switch, amplify, or mix electrical signals. It's hard to walk through a computer or elec-

Page 2

tronics laboratory without running into the DEC trademark. Since DEC modules are used primarily in designing and testing computers, it was almost inevitable that the company eventually would build computers, too.

By the dozen. Eschewing complicated acronyms like WHIRLWIND or MANIAC, DEC calls its computers "Programmed Data Processors," and they are known as PDP-1 through PDP-6. The PDP-1, introduced in 1960, is a relatively fast machine. It caught on with scientific laboratories in the Boston area due to its low price—$120,000—and because it's relatively simple to add on to it and attach such special devices as cathode-ray tubes that handle graphic as well as numerical information.

Last year, the company brought out the PDP-5, a small-scale computer selling for $27,000. At the Fall Joint Computer Conference, in Las Vegas, the computer was programmed so visitors could try their hand at shooting down a simulated satellite—a moving circle displayed on the computer's display tube. Successful shots were rewarded by a croaking rendition of Taps played by the computer's memory circuits. Inside one cabinet was a hand-lettered sign reading "$218,880 a dozen." The company's quantity discounts start with six units.

The PDP-5 has been selling briskly—by the half dozen—ever since. So has an important accessory that DEC developed to go with it—a very small magnetic tape unit with 5-in. reels called the Micro Tape. This device fills the gap between relatively slow punched paper tape readers and printers and regular high-speed tape units.

No super service. DEC breaks with other computer industry practices. It doesn't lease its equipment. Nor is it anxious to maintain it. It prefers to have customers send their own employees to a computer maintenance training course that DEC offers.

In an industry known for super service, DEC's take-it-or-leave-it approach comes as a breath of fresh air. So does the attitude of the company's two top executives, Pres. Kenneth Olsen and Vice-Pres. Harlan Anderson: After due consideration they decided they wouldn't have their picture taken with their new computer because they don't believe in personal aggrandisement.

Nor do they believe in pell-mell growth: "Our biggest problem is restraint," says Olsen. The company has never needed new equity capital, and Olsen doesn't think he will need it in the future, since he wants DEC to grow slowly enough to maintain his employees' "spirit of camaraderie."

Intimate approach. So far, DEC's customers have been largely scientific laboratories and universities. To some extent, these clients form a loose fraternity that traces its origins back to MIT and Lincoln Laboratory, where Olsen and Anderson also learned their computer savvy.

The company encourages the intimate approach to sales, keeps track of every inquiry, letter, and order on a computer tape, and refers to the name file each time correspondence comes in. DEC does almost no advertising, but puts much effort into its mail-order operation, doing all its own technical writing, printing, and art work in a big print shop. Its entire plant is in part of the former American Woolen Co. textile mill—a century-old, rambling brick complex that's now a rabbit-warren of small companies.

Computer-man's computer. Olsen and Anderson are confident that the PDP-6 will find a market. Says designer Gordon Bell: "We tried to think out what a computer should be, then designed a system that could grow gracefully and easily." He believes the PDP-6 systems will outperform systems that cost twice as much.

The elements of the computer system are built on a modular basis, and are interconnected by a network of high-speed circuits called busses. The design is such that subsystems added to the network do not have to work synchronously; that makes it easier to add other elements as the system grows.

Obviously, the machine is a computer-man's computer, and is aimed at the relatively few customers who feel themselves limited by present commercial equipment. Since the machine was just introduced, DEC claims no sales. All Olsen will say is: "There has been considerable interest on the part of people who are interested in buying more than one."

Wider market. Scientific users make up most of DEC's clients, but this, says Olsen, may not always be so. "As business applications get more sophisticated, the machine that's required differs less from the scientific machines. This way, we might get into business applications."

And perhaps by the time DEC gets big enough to supply a lot more customers, there may be lots more customers sophisticated enough to buy low-cost, highly advanced equipment that comes without much service or software. **End**

DIGITAL EQUIPMENT CORPORATION


PDP-6

Programmed Data Processor-6 is a medium scale general purpose computer with a high operating speed, unique logical organization, and growth capability. It is designed for use as a research tool in controlling experiments, collecting data, and automatically analyzing data. In addition, the computer is exceptionally well suited for use in scientific computation centers. The PDP-6 prototype is now under construction, and production deliveries will begin in early 1964.

PDP-6

8 ACCUMULATORS — Increase effective computing speeds

15 INDEX REGISTERS — Provide extensive capability for address modification in program loops

16-WORD FAST MEMORY — Has a 0.5-microsecond cycle time, is used interchangeably for accumulators, index registers, list pointers, program flags, and small program loops

16,384-WORD MAIN MEMORY — Has a 4-microsecond cycle time, is asynchronous with Central Processor

262,144 WORDS MAXIMUM — All directly addressable. Memory can be expanded and mixed in speed. Overlapping increases system speed

520 INSTRUCTIONS — Well organized and complete, provide excellent manipulative power. Features include character manipulation, list processing, and floating point hardware

CENTRAL PROCESSOR

The PDP-6 Central Processor is designed to operate asynchronously with respect to memory. After the memory has transmitted the requested word to the Central Processor, each operates at its maximum speed independent of the other. Eight accumulators and 15 index registers are stored in the first eight memory addresses of the Fast Memory. Most instructions can simultaneously and directly address any of the accumulators and any memory word and use any index register. Thus in 8 microseconds the Processor can make up to four memory references in carrying out one instruction. The asynchronous design assures that all parts of the system are running at top speed and that waiting times are minimized.

The Central Processor also provides control for in-out transfers. For slow in-out devices, like typewriters, the information is moved from the typewriter buffer under program control. The typewriter subroutine normally would be initiated by a program interrupt originating with the typewriter.

For higher speed in-out devices, like magnetic tape, the Central Processor handles the data movement automatically to memory with no program required for individual word transfers. This type of block transfer is started by the program, but then continues with no interruption of normal computing other than the memory cycle required to store or fetch the in-out word.

For some real time applications where absolutely no interruption of normal computing can be allowed, a separate In-Out Processor is used. It moves information in or out of one memory module while the Central Processor is computing with another memory module.

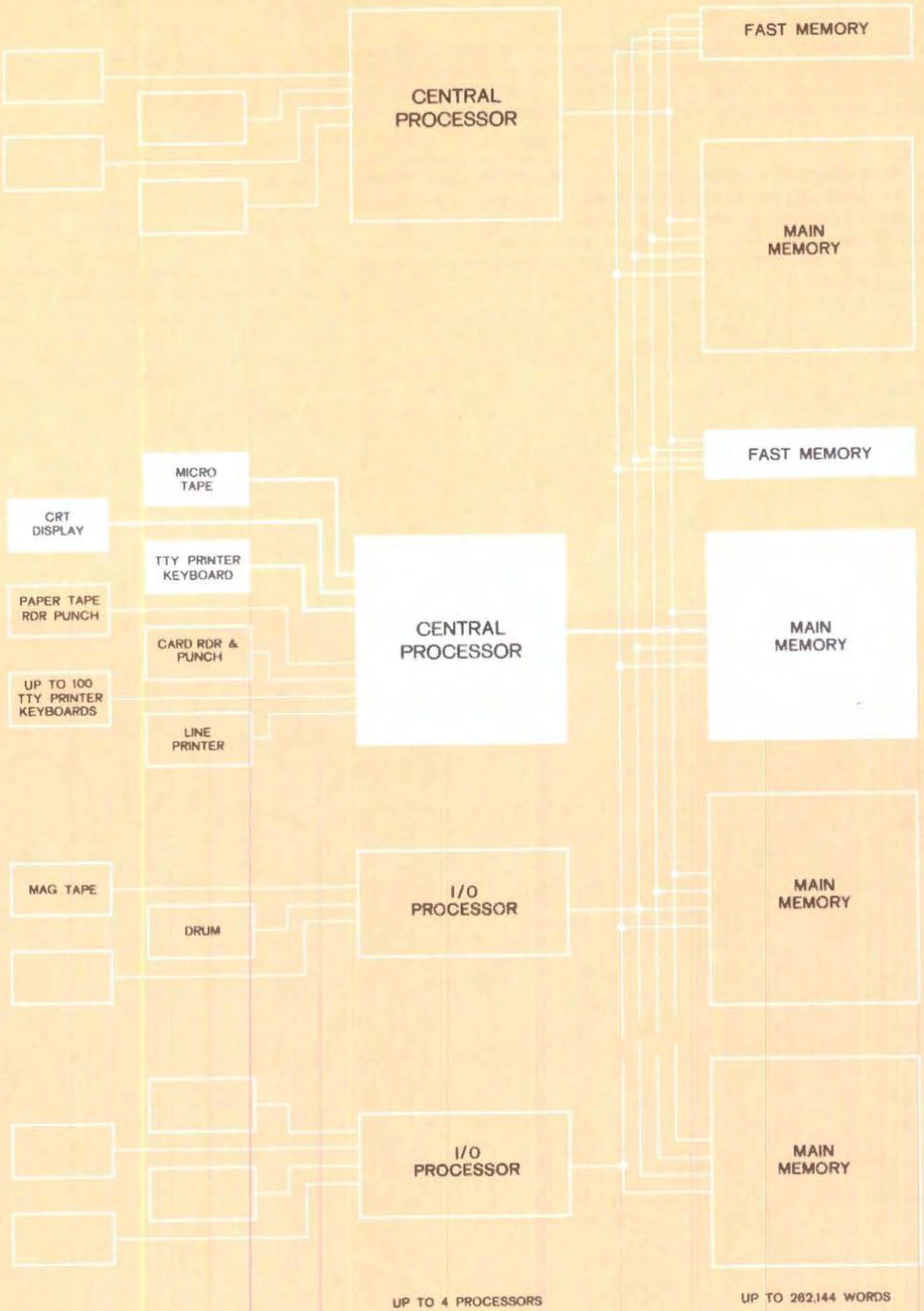
The instruction times for a basic configuration will vary, depending on the way in which the two memories are used. The following examples illustrate add times.

Example 1: add instruction is in Main Memory
operand is in Main Memory
indexing occurs
Complete Execution Time — 8 μsec

Example 2: add instruction is in Fast Memory
operand is in Main Memory
indexing occurs
Complete Execution Time — 3.5 μsec

MEMORY

PDP-6 memory is divided into two parts. Main Memory consists of 8192-word modules having a 4-microsecond cycle time. Fast Memory consists of 16-word modules having a 0.5-microsecond cycle time. Each memory module has facilities for connecting as many as four processors. These can be either Central Processors or In-Out Processors. Memory modules are self contained and can be operated simultaneously in parallel. This can occur with a single processor or with multiple processors.


CENTRAL PROCESSOR
FAST MEMORY
MAIN MEMORY
MICRO TAPE
CRT DISPLAY
TTY PRINTER KEYBOARD
PAPER TAPE RDR PUNCH
CARD RDR & PUNCH
UP TO 100 TTY PRINTER KEYBOARDS
LINE PRINTER
CENTRAL PROCESSOR
FAST MEMORY
MAIN MEMORY
MAG TAPE
DRUM
I/O PROCESSOR
MAIN MEMORY
I/O PROCESSOR
MAIN MEMORY
UP TO 4 PROCESSORS
UP TO 262,144 WORDS

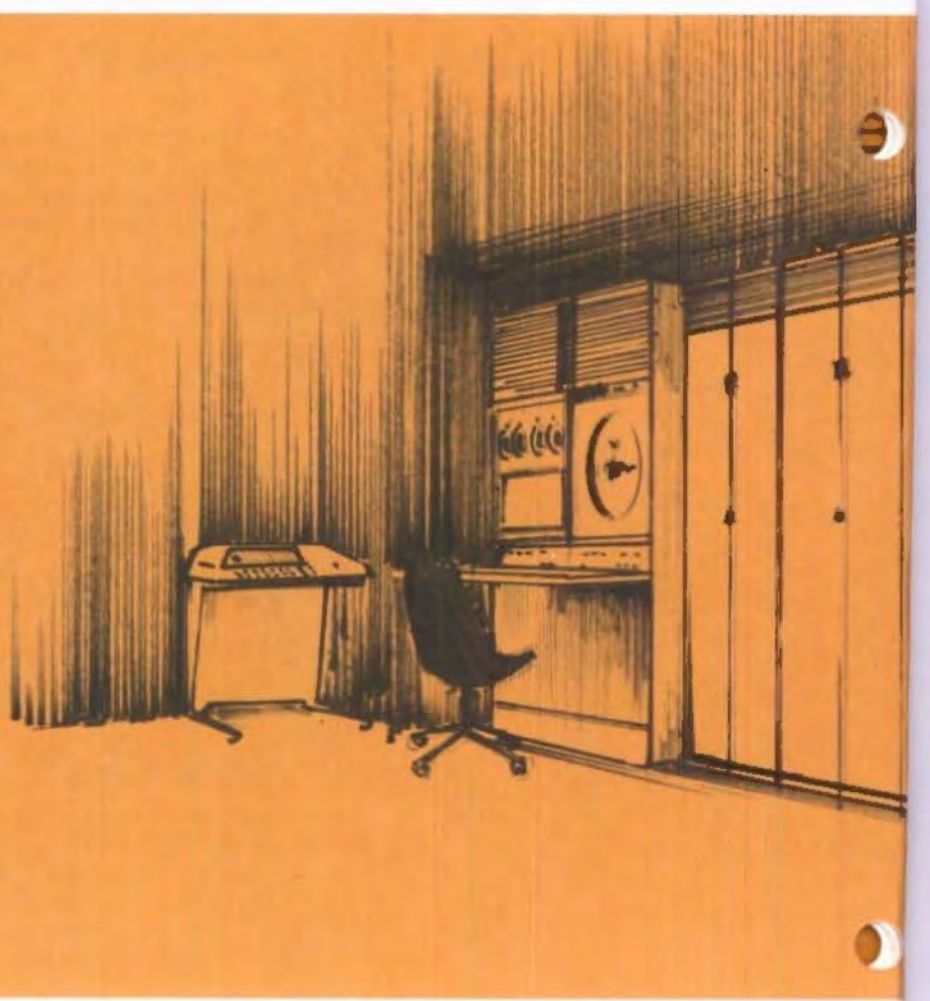
The Fast Memory is addressable with normal memory addresses. Registers 0-7 can be addressed as index registers and/or accumulators. Memory overlap is automatically achieved due to the asynchronous relationship between the processors and memory.

This short program illustrates one use of the PDP-6 Fast Memory (0.5 microsecond) to increase the speed of performing iterative operations. Four instructions move Table 1 to Table 2 (table size 1000) in the times indicated. Times for performing the same operations exclusively in the Main Memory (4-microsecond) are shown for comparison.

Move "table 1" to "table 2," of length, size

	INSTRUCTIONS			FAST AND MAIN MEM.	MAIN MEM. ONLY
	6,mov	i	1000	(4 + 0.5)*	(4 + 4)*
y	5,mov		table 1,6	4 + 0.5	4 + 4
	5,mov	d	table 2,6	4 + 0.5	4 + 4
	6,soa	g	y	1	4
				10 μsec	20 μsec
				× 1000 = 10 ms	× 1000 = 20 ms

*occurs first time only



INPUT-OUTPUT EQUIPMENT

Three types of input-output devices are included in the standard PDP-6: a dual Micro Tape Transport (fixed address magnetic tape for high speed loading and readout, as well as program updating), a Teletype Printer-Keyboard, and a Precision CRT Display with Light Pen for direct, high speed man-machine communication.

A variety of optional devices can be added from Digital's standard line of peripheral equipment: high speed line printer, card reader and punch, perforated tape reader and punch, magnetic tape transports and controls, serial and parallel drums, ultra-precision and special purpose digital CRT displays, and other digital equipment compatible with PDP-6.

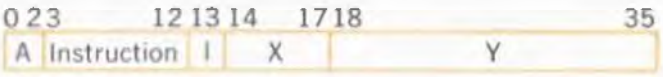
SOFTWARE

A complete software package is furnished with PDP-6. It includes an algebraic compiler, a symbolic assembler, an executive system, and utility routines. The executive system permits editing of Micro Tape, calls the compiler and assembler when needed, and calls subprograms that facilitate program checkout.

INSTRUCTIONS

PDP-6 has an exceptionally powerful set of 520 instructions, including such special features as immediate, direct, and indirect addressing, push-down list, character handling, memory modification and conditional branching, and operand preservation by one instruction. While the number of instructions is large, they are logically related and consistent. Good organization results in straightforward programming.

The primary instruction format is shown below. The instruction part is made up of the operation and the mode. Normally the operation specifies the action to be taken and the mode determines either the location of the results or conditions to be tested.



Each instruction word has an effective address, computed using I, X, and Y. Y is the address part, which may specify any of 262,144 locations. When the X, or index, part is non-zero, the contents of the specified index register are added to Y to form a new address. When the I, or indirect bit, is 0, the address thus computed is the effective address. In addition each instruction word specifies, with the three bits in A, one of eight accumulators to be used by the instruction.

When the indirect bit is 1, the address computed above is used to reference a memory location whose contents are used to form a new address. This procedure is repeated until a location is referenced whose I bit is 0.

Direct or indirect instructions designate as the effective address a memory register which contains the data to be used. A third type of instruction can specify (by choice of mode) an immediate address, in which case Y itself is taken as the number to be used.

The accumulators and index registers are stored in identical fast memory registers. Their addresses correspond as shown below.

Fast Memory Addresses	Accumulator Addresses	Index Register Addresses	
0	0	10	unused
1	1	11	1
:	:	:	
7	7	17	7
10			
:			
17			

DATA TRANSMISSION

FULL WORD TRANSMISSION

Operations

mov	Move full word
move	Move full word with right and left halves exchanged
movc	Move complement of full word
movn	Move negative of full word
movm	Move absolute magnitude of full word
mmov	Multiple move. Move full words 0, 1 . . .A

HALF WORD TRANSMISSION

Operations

movr	Move right half only
movl	Move left half only
mvlr	Move left half to right half
mvrl	Move right half to left half

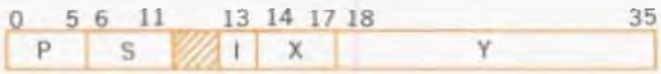
FULL OR HALF WORD TRANSMISSION

Modes

—	Load accumulator with direct address
i	Load accumulator with immediate address
d	Deposit accumulator with direct address
x	Exchange accumulator with direct address

Full flexibility for making full word or half word transfers between memory and accumulators is built into the PDP-6

CHARACTER MANIPULATION — Characters of arbitrary size are located by a position pointer word of the following format:



I, X, and Y are used in the usual manner to locate the word containing the character. P is the position of the character such that the rightmost bit of the character is 35-P. S is the size of the character, which may be from 1 to 36 bits.

ldc	Load accumulator (right-justified) with character specified by the pointer word. The effective address of the instruction locates the pointer word.
dpc	Deposit character (right-justified) of the accumulator in character position specified by pointer, as above.
cao	Add one character position to character pointer word. The position is moved to the right by the size of the character. If the result is off the right end of the word, the Y part is incremented and the P part is reset to the leftmost character position.
ldci	Same as ldc, but increments pointer word by one character after loading.
dpci	Same as dpc, but increments pointer word by one character after depositing.

PUSH DOWN LIST

psh	Takes the word at the effective address and puts it on the push-down list. The accumulator address part points to the next free location at the completion of the instruction. The left half of the accumulator is counted up by one.
pshs	Push and skip. Same as psh, but skips if the accumulator bits 0-17 are not zero
pop	Subtract one from each half of the accumulator. Take the word specified by the address of the accumulator and put it at the effective address.
pops	Pop and skip. Same as pop, but skips if contents of accumulator bits 0-17 are not zero.

INPUT-OUTPUT — Set initial conditions in I/O equipment, interrogate status, and transfer data between devices and memory.

inm	Input word from device n to memory
inz	Input, mask with immediate, and skip if zero
inn	Input, mask with immediate, and skip if non-zero
inch	Input character from device n to memory
outm	Output word to device n from memory
outi	Output immediate to device n
outch	Output character to device n from memory

ARITHMETIC

FIXED POINT ARITHMETIC — Data (either immediate or direct) and accumulator can be combined using 2's complement arithmetic. Mode specifies location of the results. In general the carry flag will be set if the result is greater than 2^{36}; the overflow flag will be set if the result is greater than 2^{35}.

Operations

add	Add full word
addr	Add to right half only
addl	Add to left half only
addrl	Add right half to left half
addlr	Add left half to right half
adds	Add both halves swapped
sub	Subtract full word
subr	Subtract from right half only
subl	Subtract from left half only
subrl	Subtract right half from right half
sublr	Subtract left half from right half
subs	Subtract right half from left half
iml	Integer multiply
idv	Integer divide
imlu	Integer multiply unsigned
idvu	Integer divide unsigned
mul	Multiply, giving double length result. Low order part goes to A + 1 when appropriate
umul	Unsigned multiply, double length result
div	Divide, giving quotient and remainder. Remainder goes to A + 1 when appropriate
undiv	Unsigned divide.

Modes

—	Direct address combined with accumulator, results to accumulator
d	Direct address combined with accumulator, results to memory
b	Direct address combined with accumulator, results to both memory and accumulator
i	Immediate address combined with accumulator, results to accumulator

FLOATING POINT ARITHMETIC AND CONVERSION — Floating point numbers can be combined using a word format of 1 bit for sign, 8 bits for exponent, and 27 bits for fraction. The mode specifies location of the results. Conversion is fixed-to-floating and floating-to-fixed. The effective address is used as a scaling constant.

Instructions

fad	Floating Add
fsb	Floating Subtract
fmp	Floating Multiply
fdv	Floating Divide

Modes

—	Accumulator is combined with the direct address, results to Accumulator
d	Same, but results to direct address
b	Same, but results to both direct address and accumulator
r	Same but results to accumulator, remainder and low significant bits to accumulator plus one

Instructions

flo	Convert fixed to floating
fix	Convert floating to fixed
dflo	Convert double length fixed to floating
dfix	Convert double length floating to fixed
fsc	Floating point scale

LOGICAL

Operations

and	And
ior	Inclusive Or
xor	Exclusive Or
anc	And complement. Clear selected bits

Modes

—	Direct address combined with accumulator, results to accumulator
d	Direct address combined with accumulator, results to direct address
b	Direct address combined with accumulator, results to direct address and accumulator address
i	Immediate address combined with accumulator, results to accumulator

SHIFTING

There are three kinds of shifting, as follows:

rotate — bits leaving one end enter at the other

arithmetic shift — performs 2's complement multiplication by powers of 2. The sign is unchanged. When going to the right, the first bit shifted in is the Exclusive Or of the sign and overflow, and remaining bits are the sign. When going left, overflow may be set.

logical shift — bits leaving one end are lost, and zeros enter the other end

Shifts may be of an accumulator, A, or of a combined word consisting of A and A + 1.

ral	Rotate accumulator left
rar	Rotate accumulator right
aal	Arithmetic shift accumulator left
aar	Arithmetic shift accumulator right
lal	Logical shift accumulator left
lar	Logical shift accumulator right
rcl	Rotate combined registers left
rcr	Rotate combined registers right
acl	Arithmetic shift combined registers left
acr	Arithmetic shift combined registers right
lcl	Logical shift combined registers left
lcr	Logical shift combined registers right

MEMORY AND ACCUMULATOR MODIFICATION AND TESTING

There are two sets of like instructions in this group:

Memory modification instructions — Modify the direct addressed register. The result is transmitted to the specified accumulator, except that a zero in the A position specifies no accumulator is to be changed. If the condition specified by the mode is met, the next instruction is skipped.

Accumulator modification instructions — Modify the specified accumulator. If the condition specified by the mode is met, control is transferred to the effective address.

Memory Skips	*Accumulator Transfers*	*Operations*
skp	tra	No modification, test full word
skpl	tral	No modification, test left half only
skpr	trar	No modification, test right half only
clm	cla	Clear register
aom	ado	Add one to full word
aoml	adol	Add one to left half, test left half only
aomr	ador	Add one to right half, test right half only
aomb	adob	Add one to both halves, test full word
som	sbo	Subtract one from full word
soml	sbol	Subtract one from left half, test left half only
somr	sbor	Subtract one from right half, test right half only
somb	sbob	Subtract one from both halves, test full word
msw	asw	Swap two halves of word, test full word
cmm	cma	Logical complement
ngm	nga	Negate (take 2's complement)
mgm	mga	Take magnitude (2's complement if negative)

Modes

Memory Skips	*Accumulator Transfers*	*Skip if Condition Met*
—	—	Never
s	t	Always
z	z	Equals zero
n	n	Not equal zero
ge	ge	Greater than or equal to zero
g	g	Greater than zero
le	le	Less than or equal to zero
l	l	Less than zero

COMPARE ACCUMULATOR WITH MEMORY AND SKIP — Compare a data word, which may be an immediate or direct address, with an accumulator by subtraction, and skip according to the mode selection. The results of the comparison do not affect memory or the accumulator.

Instructions

cfw	Compare accumulator with word
cfi	Compare accumulator with address
clh	Compare left half of accumulator with left half of direct addressed word
cli	Compare the left half of accumulator with zero (or left half of immediate address)
crh	Compare right half of accumulator with right half of word
cri	Compare right half of accumulator with address
clr	Compare left half of accumulator with right half of addressed word
clri	Compare left half of accumulator with immediate address
cao	Add one to accumulator, then compare the full words
caoi	Add one to accumulator, then compare the accumulator with the immediate address

Modes

—	Never skip
s	Always skip
e	Skip if equal
n	Skip if not equal
ge	Skip if accumulator is greater than or equal to data
g	Skip if accumulator is greater than data
le	Skip if accumulator is less than or equal to data
l	Skip if accumulator is less than data

ACCUMULATOR BIT MODIFICATION AND TESTING — Bits of an accumulator that are masked by bits of a data word can be tested and/or modified. The data word may be either direct or immediate. Mode specifies whether to use direct or immediate address for data word (normal or right and left halves exchanged) and conditions on which to skip.

Instructions

tst	Test only, do not change accumulator
clb	Clear bits in accumulator masked one by data word
cmb	Complement bits in accumulator masked one by data word
stb	Set bits in accumulator masked one by data word
stst	Test only with the data word, right and left halves exchanged. The data word and accumulator are unchanged
sclb	Clear bits in the accumulator masked one by the data word, right and left halves exchanged
scmb	Complement bits in the accumulator masked one by the data word, right and left halves exchanged
sstb	Set bits in the accumulator, masked one by the data word, right and left halves exchanged

Modes

—	Perform specified operation using directly addressed data
o	Skip if any accumulator bits masked one by the directly addressed data word are one, then perform the specified operation
z	Skip if all accumulator bits masked one by the directly addressed data word are one. Then perform the specified operation
s	Always skip, use directly addressed data word
i	Use immediate addressed data word
io	Same as o, but immediate addressed data word
iz	Same as z, but immediate addressed data word
is	Same as s, but immediate addressed data word

MISCELLANEOUS

SUBROUTINE CALLING

tsp	Transfer and save program counter. The PC and effective address are saved in the right and left halves of the accumulator, respectively. Control is transferred to the effective address.
tsa	Transfer and save accumulator. The accumulator is stored in the effective address. The PC and effective address are saved in the accumulator, as above. Control is transferred to the effective address plus one.
cal	Call subroutine. The accumulator is stored in memory register 20. The PC and effective address are saved in the accumulator as above.
trr	Transfer and restore accumulator. This is the return instruction that matches tsa. The accumulator is replaced by the word addressed in its left half. Control is transferred to the effective address.

EXECUTE

xct	Execute instruction at effective address

NO OPERATION

nop	No operation is performed

HALT

hlt	Halt and transfer

The above specifications are subject to change without notice.

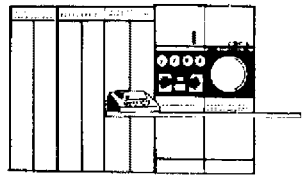
DIGITAL EQUIPMENT CORPORATION · MAYNARD, MASSACHUSETTS

F-61

Printed in U.S.A.

5M 5/63

PDP-6
PRICE LIST
FEBRUARY 1, 1964



ARITHMETIC PROCESSOR TYPE 166

Fast Memory 0.5 Microsecond - 16 Words
16 Accumulators
15 Index Registers
Floating Point Arithmetic
I-O Teleprinter
7 Channel Priority Interrupt I-O System
$176,100

CORE MEMORY TYPE 163B

8,192 36-bit words $ 80,000

CORE MEMORY TYPE 163C

16,384 36-bit words $126,000

Stores either 8,192 or 16,384 36-bit words with a memory cycle time of 2.0 microseconds. Includes one memory bus interface. For each additional processor, an additional interface is required. Each interface services one memory-processor.

Each additional interface (maximum of 3) $ 2,700

MAGNETIC DRUM AND PROCESSOR TYPE 236

Controls from one to four drum units. Each drum unit consists of 768 tracks of 49,152 bits each for a total storage of 37.8 x 10^6 bits. The drum rotates at 1150 rpm (52 milliseconds per revolution), and provides a 36-bit word transfer every 6.4 microseconds. Total drum storage is 1,048,576 36-bit words. $ 95,000

PAPER TAPE AND CARD EQUIPMENT

HIGH SPEED PAPER TAPE READER AND CONTROL TYPE 760

Reads perforated paper tape photoelectrically at 400 characters per second. $ 9,000

HIGH SPEED PAPER TAPE PUNCH AND CONTROL TYPE 761

Punches 8-hole paper tape at 63.3 characters per second. $ 5,500

CARD READER AND CONTROL TYPE 461

Provides on-line reading of 80-column punched cards at 200 or 800 cards per minute in either alphanumeric or binary codes.

200 cards per minute $ 16,500
800 cards per minute $ 27,200

CARD PUNCH AND CONTROL TYPE 460

Permits on-line punching of 80-column cards at 100 or 300 cards per minute.

100 cards per minute $ 29,000
300 cards per minute $ 42,000

PRINTERS

AUTOMATIC LINE PRINTER AND CONTROL TYPE 646

Prints 300 lines per minute, 120 columns per line, 64 characters per column. $ 30,000

HIGH SPEED AUTOMATIC LINE PRINTER AND CONTROL TYPE 680

Prints 1000 lines per minute, 120 columns per line, 64 characters per column. $ 47,500

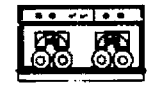
MAGNETIC TAPE EQUIPMENT

MAGNETIC TAPE TRANSPORT TYPE 570

Reads and writes IBM compatible tape at a recording density of 200 or 556 characters per inch. Tape speed is 75 or 112.5 inches per second with transfer rates from 15 to 62.5 KC. $ 30,400

MAGNETIC TAPE CONTROL TYPE 516

Automatically controls up to eight Type 570 magnetic tape transports. Permits reading, writing, forward/backward spacing, and rewind. Requires Type 136 Data Control. $ 15,000



DUAL MICROTAPE SYSTEM

Provides a fixed address magnetic tape facility for high speed loading, readout and program updating. Density is 375 ± 60 bits per track inch at a speed of 80 inches per second. Transfers 90,000 bits per second.

Type 555 Dual Transport (includes two independent tape drives) $ 7,400

Type 551 Control Unit (controls up to eight Type 555 Tape Transports. Requires Type 136 Data Control $ 14,000

OTHER EQUIPMENT

DATA CONTROL TYPE 136

Assembles and disassembles 36-bit words. May be used with up to four tape controls and two special purpose data handling devices. $ 10,000

DISPLAY MONITOR AND CONTROL TYPE 346

Plots points, lines, vectors, and characters on a 9⅜-inch square raster of 1024 points along each axis.

1½ microseconds is required per point in vector, increment and character modes; random point plotting rate of 35 microseconds. Uses fiber optic light pipe and photomultiplier system for fast detection of displayed information. $ 40,000

MULTI-USER STATION TYPE 627

Standard ASCII 8-level Teletypewriter, paper tape reader-punch, operating at 10 cps. $8,000

On Line

Prepares programs directly onto magnetic tape
Permits program execution and debugging
Permits tape editing
Provides hard copy printouts

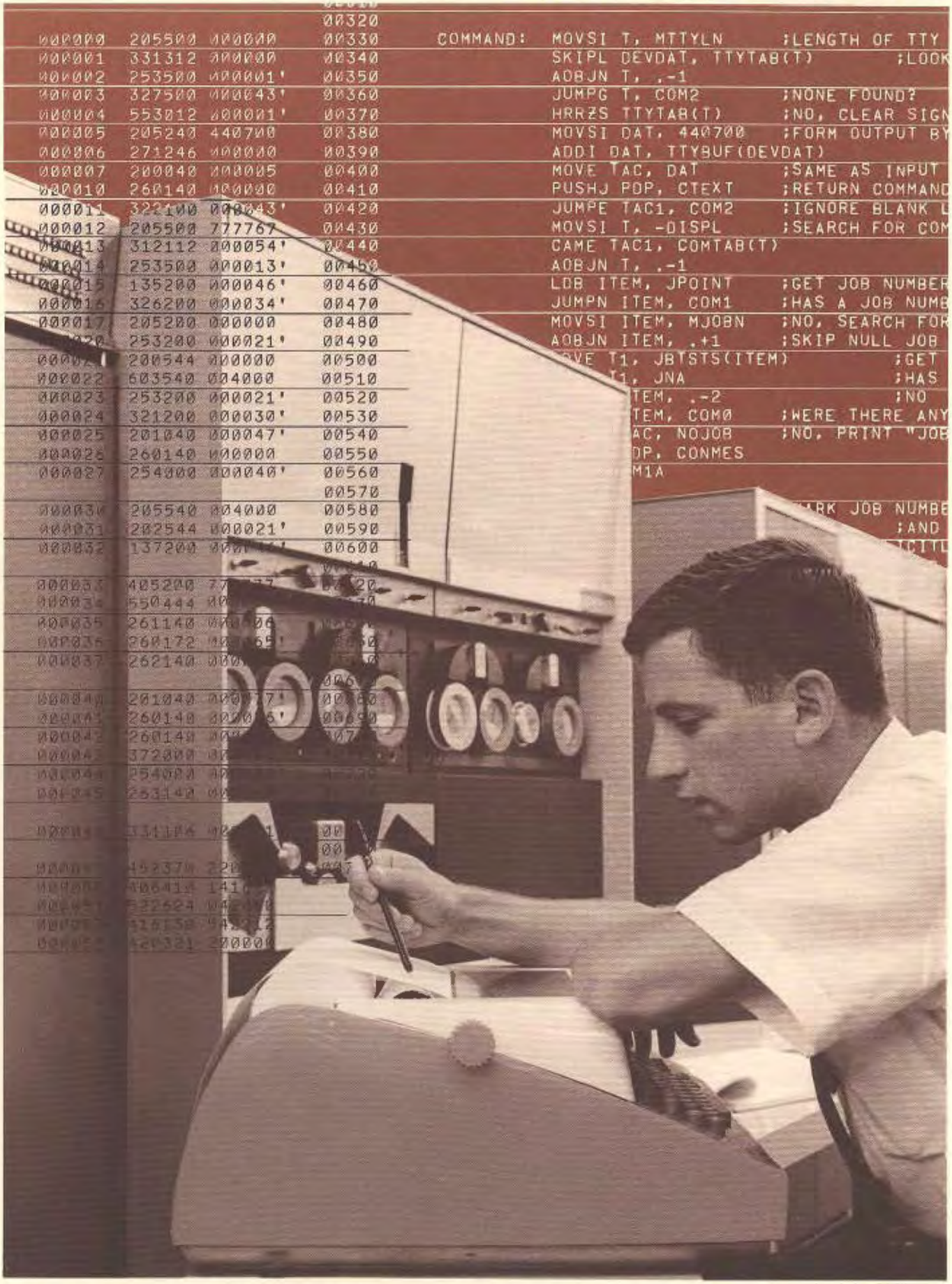
Off Line

Punches paper tape

Prices quoted are effective February 1, 1964, FOB Maynard, Mass., and apply in continental United States only. Federal, state or local taxes are not included. Option prices are for factory installation; field installation prices will be quoted on request. Quantity prices are quoted on request. All prices are subject to change without notice.

PROGRAMMED DATA PROCESSOR-6

PDP-6 TIME SHARING SOFTWARE



DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

COMPREHENSIVE TIME-SHARING....

The Programmed Data Processor-6 is an integrated hardware-software system which is capable of significantly lowering computation costs and improving program turn-around time. Efficient use of the central processor and peripheral equipment is made possible by concurrent operation of several user programs. The PDP-6 Monitor is a comprehensive control program which: 1) simplifies the use of the PDP-6's extensive asynchronous input-output capabilities, 2) maximizes the use made of the high-speed arithmetic processor, and 3) provides dynamic run control features necessary to allow concurrent use of the system by multiple users. This gives the user on-line interaction capabilities as well as efficient job-shop computation. These features lead to higher through-put rates at lower cost than is possible with conventional serially operated monitor systems.

The following sections are intended to familiarize the reader with the use of the Monitor and the salient features of the software system. Appropriate references at the end of each section should be used for a more detailed study of the PDP-6 time sharing system.

CONSOLE USE OF COMMON USER SERVICE PROGRAMS

Common user service programs are system library programs that perform file manipulation, editing, special desk calculating, and other functions of general interest. By adding programs to the system library, new facilities are immediately made available to all users.

CONSOLE USE OF THE SYSTEM FOR PROGRAMMING

A user may create and edit a program (or text), translate the program to machine language (using FORTRAN, MACRO, or some other translator), load and run the program, and, if necessary, directly communicate with the program through the DDT debugging language.

CONSOLE USE OF THE COMMON USER SERVICE PROGRAMS FOR DATA CONVERSION

The peripheral interchange programs accomplish general data conversion from medium to medium, e.g., card to tape, tape to printer, etc.

UNATTENDED STACK OR BATCH PROCESSING OF JOBS

A job stack runs as though it were console controlled. Jobs are stacked in a card reader or other input device and processed in sequence as time-shared jobs.

SPECIAL PURPOSE CONSOLE SERVICE

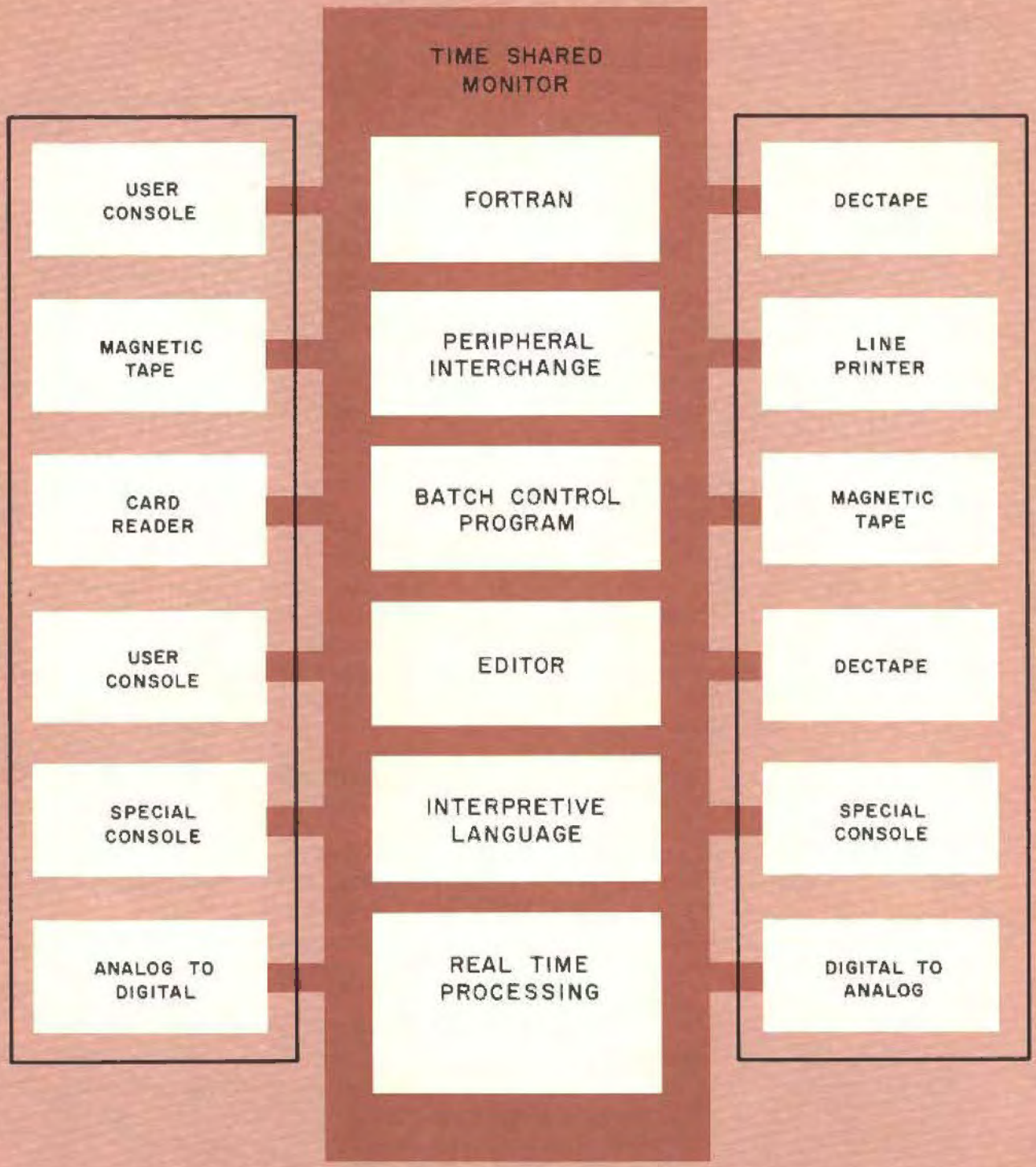
Monitor subroutines may be modified so that consoles requiring special monitor service may be included.

REAL TIME PROCESS SERVICE

Input/output routines to connect a special device with the programming system are easily added. A job may issue system input/output commands for a special device in the same manner as for conventional devices.

The first delivered

.....NOW

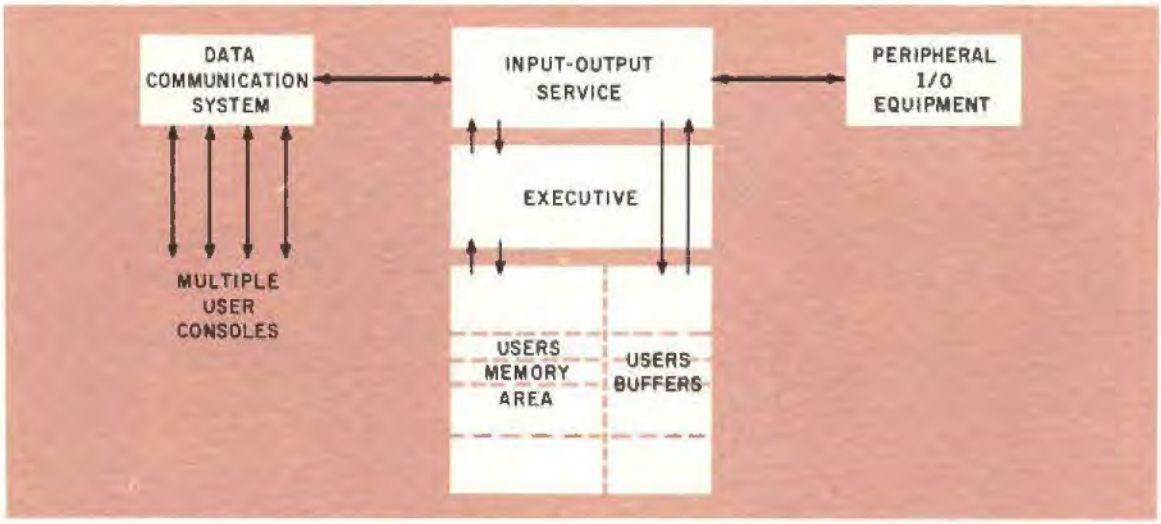


software system written specifically for time-sharing.

IJOB	Initialize the job to which the console is attached. The job will have no core or devices assigned to it.
PJOB	Print the job number to which the console is currently attached.
KJOB	Kills the job and returns to the system all resources assigned to it. This command should be typed whenever a user is through with a job in order to allow someone else to use the job number.
CORE N	Sets the total number of 1024-word (decimal) blocks of core to N (decimal) for the job to which the console is currently attached.
GET DEVICE:FILE	Gets a previously saved program from device "DEVICE" and places it in core. This is the mechanism for loading system programs such as the assembler. It is also useful for loading user programs which have been previously saved by the "SAVE" command. The program is not started.
SAVE DEVICE:FILE	Saves the core image on device "DEVICE". All devices are released, and the program counter is set to the program's starting address before the file is written.
START LOC	Starts execution of the program at relative octal location "LOC" in the job area or at the program starting address if "LOC" is not specified. No check is made to see if a program has been loaded. All succeeding input will be directed to the user's program.
CONT	Continues execution from wherever the program was stopped by <control> C. All succeeding input will be directed to the user's program.
DDT	Starts execution of user DDT in the job area. All succeeding input will be directed to "DDT" and the user's program.
ASSIGN DEV:NAME	Assigns physical device DEV to the job. No other job may use a device once it has been assigned.
	If DEV is the first three characters of one of the multiple devices (TTY, DTA, MTA) the Monitor will search for a free device. In any case, the Monitor will indicate the device assigned.
<CONTROL>C	Prints ↑C and returns control to Monitor command mode.
<RUBOUT>	Prints and deletes the last character typed.

TIME-SHARING MONITOR SYSTEM

Schedules multiple-user time-sharing of the system
Allocates facilities to particular users
Accepts input from and directs output to all system I/O devices
Relocates and protects user programs in available memory



The Monitor system is a collection of programs remaining permanently in memory to provide overall coordination and control of the total operating system. It performs several functions. First, it permits several users' programs to be loaded into core memory simultaneously. The Monitor makes use of the PDP-6 time-sharing hardware to prevent one user's program from interfering with other users' programs. Each program is run for a certain length of time; then the Monitor switches control to another program in a rotating sequence. Switching is frequent enough so that all programs appear to run simultaneously.

Another function of the time-sharing Monitor is to process input/output commands. Only one user at a time is permitted to operate each particular device. The input/output service routines preprocess data so that all devices appear identical to the user's program, thus simplifying coding. The Monitor makes use of the PDP-6 program interrupt system to overlap input/output operations with computation. If a user's program must wait for completion of an input or output operation, the Monitor automatically switches to another user's program. A program may be terminated temporarily by user intervention, or it may suspend its own operation. Temporary termination does not remove the program from memory. A program may be dumped on backing storage and discontinued under user control.

The facing page illustrates some of the commands defined in the Monitor system. Their specific use is illustrated for each of the common user service programs in the sections following:

SIZE: Permanently loaded into the first 5 x 1024 memory locations.

INPUT: Input service routines are provided for:

Device	System Mnemonic
card reader	CDR:
paper tape reader	PTR:
magnetic tape x	MTA x:
DECtape x	DTA x:
user Teletype	TTY:

OUTPUT: Output service routines are provided for:

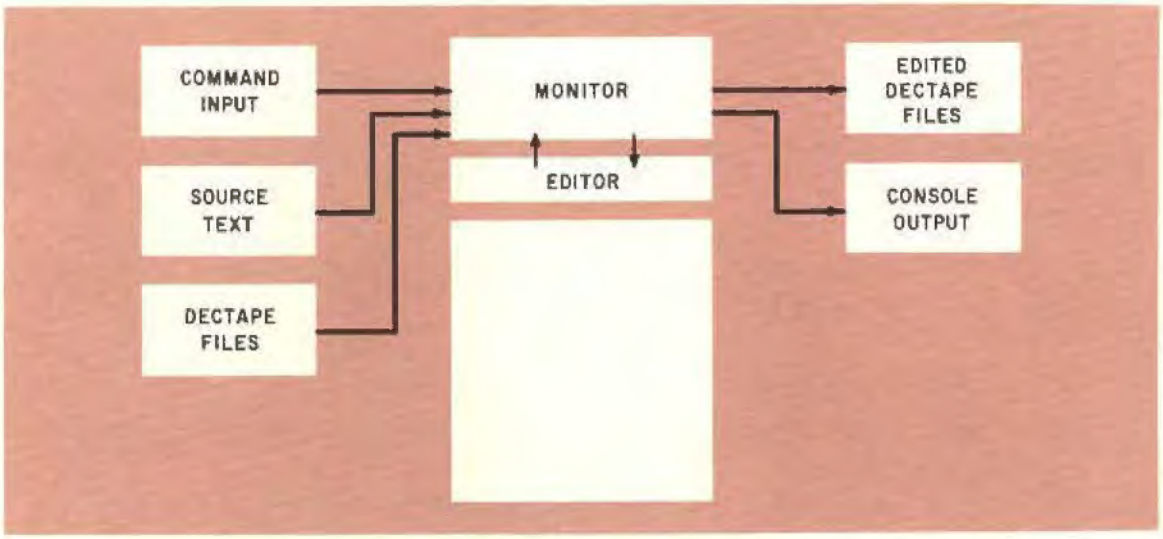
Device	System Mnemonic
magnetic tape x	MTA x:
DECtape x	DTA x:
punched paper tape	PTP:
line printer	LPT:
user Teletype	TTY:

REFERENCE: Multiprogramming System Manual (DEC-6-0-EX-SYS-UM-1P-PRE00).

```
IJOB                                    Initialize job

ASSIGN DTA                              Assign a DECtape
DEVICE DTA5 ASSIGNED

CORE 1                                  Assign 1,024 words of core

GET DTA0:EDITOR                         Load the editor
JOB SETUP

START                                   Start the job

S5,MATRIX                               Create a file called MATRIX on DECtape 5.

I10,10                                  Initialize sequencing at 10 and increment by 10.
00010     TITLE MATRIX
00020     X=A+B*C)
00030     DO 1 I=1,8
00040     X=(A+B)*C
.
.
.
.
00250     END                           Leave incrementing mode.
00260
I11                                     Insert line 11
00011     DIMENSION A(8)
D30                                     Delete line 30

P10,260                                 Print lines 10-250
00010     TITLE MATRIX
00011     DIMENSION A(8)
00020     DO 1 I=1,8
00040     X=(A+B)*C
.
.
.
.
00250     END
E                                       End file

↑C                                      Transfer control to monitor

KJOB                                    End the job
```

DECTAPE EDITOR

Provides a convenient means to prepare and edit any form of text
Eliminates the need for preparing punched cards off-line
Programs and text are kept on convenient DECtape reels
Takes virtually no processor time away from other programs



The DECtape Editor provides a means of creating, adding to, or deleting from sequence-numbered lines in files on DECtape. This text may be input for the FORTRAN compiler, the MACRO-6 assembler, or simply a convenient means of handling textual information.

The Editor provides means for selecting a tape unit, clearing the directory, adding a new file to the directory, or selecting a file currently in the directory.

When a file has been selected, the user may resequence it, print a line or many lines, enter new lines, delete existing lines, or replace existing lines.

A few of the specific Editor commands are:

Sx, name	Allow the user to access file "name" on DECtape unit "x"
In	Insert a line at n
Dn	Delete a line at n
Pn	Print line n
E	End the current file

The procedure for composing and editing a FORTRAN II program is illustrated on the facing page.

SIZE: 1 x 1024 memory locations
INPUT: Source text from user console
OUTPUT: Symbolically named files on DECtape

REFERENCE: Multiprogramming System Manual (DEC-6-0-EX-SYS-UM-1P-PRE00)
Chapter VII, The Editor

```
!JOB

CORE 22

GET DTA0:F2
JOB SETUP

START

DTA4:INVERS,LPT:←DTA5:MATRIX(T)

PROGRAM BREAK IS 00051

NO ASSEMBLY ERROR(S)

NO SOURCE ERROR(S)

EXIT

XJOB
```

```
                                        ;TITLE DEMO


                                        ;X=0
                                             EXTERNAL FORSE.
                                        %%%:


                                        %:
000000 015000000000                          RESET.

000001 403140000000 L                        SETZB 3,X#

                                        ;DO 1 I=1,10

000002 201140000001                          MOVEI 3,1
000003 202140000000 L                        MOVEM 3,I#
                                        %.1000:

                                        ;TYPE 10,SINF(X)

000004 201142000000 L                        MOVEI 3,%10
000005 017140100004                          OUT. 3,32772
                                             EXTERNAL SIN
000006 200740000001'L                        MOVE 15,X#
000007 260040000000 L                        PUSHJ 1,SIN
000010 200140000017                          MOVE 3,15
000011 020000000003                          DATA. 3
000012 021000000000                          FIN.

                                        ;1:X=X+.1

                                        %1:
000013 200140000000 L                        MOVE 3,[0.1]
000014 147140000006'L                        FADRB 3,X#
000015 350140000003'L                        AOS 3,I#
000016 307140000012                          CAGI 3,10
000017 254000000004'                         JRST %.1000

                                        ;10:FORMAT(F15.7)

                                        %10:
000020 254000000000 L                        JRST %.1001
000021 242146132534                          ASCII ?(F15.7)?
000022 335220000000
                                        %.1001:

                                        ;END

                                        %%%%:
000023 040000000000 L                        CALL [SIXBIT "EXIT"]
                                             END %
    LITERALS

000024 495851640000
000025 175631463146


PROGRAM BREAK IS 000030
NO  ASSEMBLY ERROR(S)
NO SOURCE ERROR(S)
```

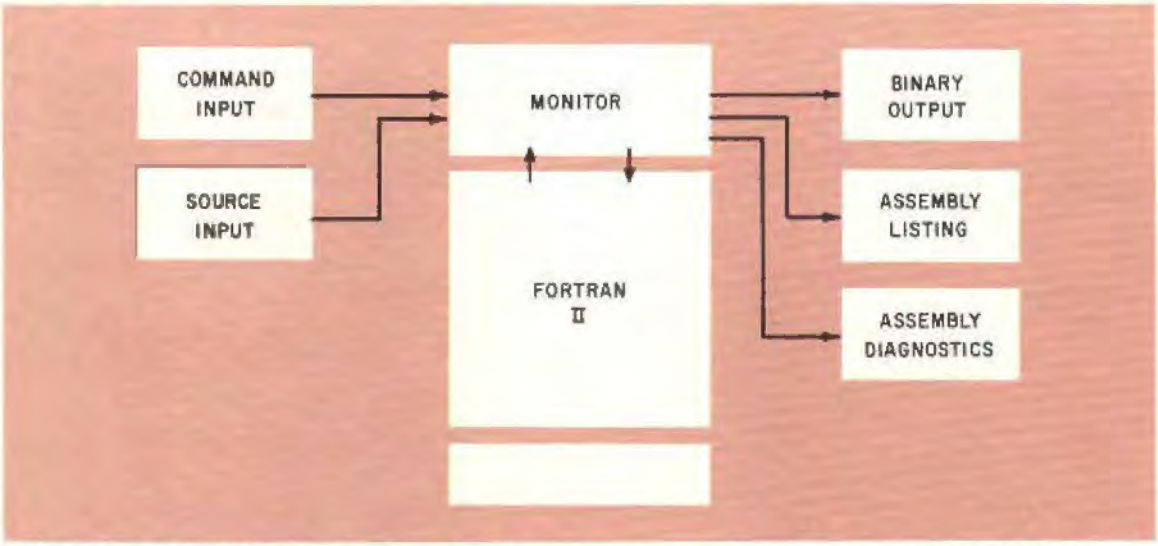
FORTRAN II COMPILER

One pass, syntax-directed compiler

Accepts input from any system input device including user console

Extended FORTRAN II language features include: Boolean operations, shifting capability, general format I/O conversions, N-dimensional arrays, multiple-equals statements

Optimized object code due to powerful PDP-6 instruction set



PDP-6 FORTRAN II operates in the time-sharing mode or independently. FORTRAN core requirements range from 10K to 22K depending on the degree of optimization of the object code desired as a result of compilation. This gives the user the choice of a smaller compiler to allow many users to time-share or a larger one to obtain a minimized object program. PDP-6 FORTRAN II language is compatible with most FORTRAN II systems. The source language provides substantial power and flexibility through a wide variety of arithmetic, control, Boolean, function (internal and external), subroutine call, and I/O statements.

Several unique features have been implemented to make FORTRAN more flexible for the time-sharing user. TYPE and ACCEPT statements offer a method whereby FORTRAN programs communicate with the user Teletype. Source formats are not limited to 80-column punched card fields, allowing FORTRAN source code to be prepared at the user Teletype console. Devices may be dynamically assigned at run time, thereby making maximum use of all peripheral equipment available in the system.

The example at the left illustrates a method of calling in a FORTRAN program. (The source FORTRAN program was previously prepared at a user's console and stored on DECtape.)

These commands take the source program from file MATRIX on DECtape 5 and compile the object code on DECtape 4 with the file name of INVERS. The listing of the compiled FORTRAN program is directed to the line printer, and is shown at the left.

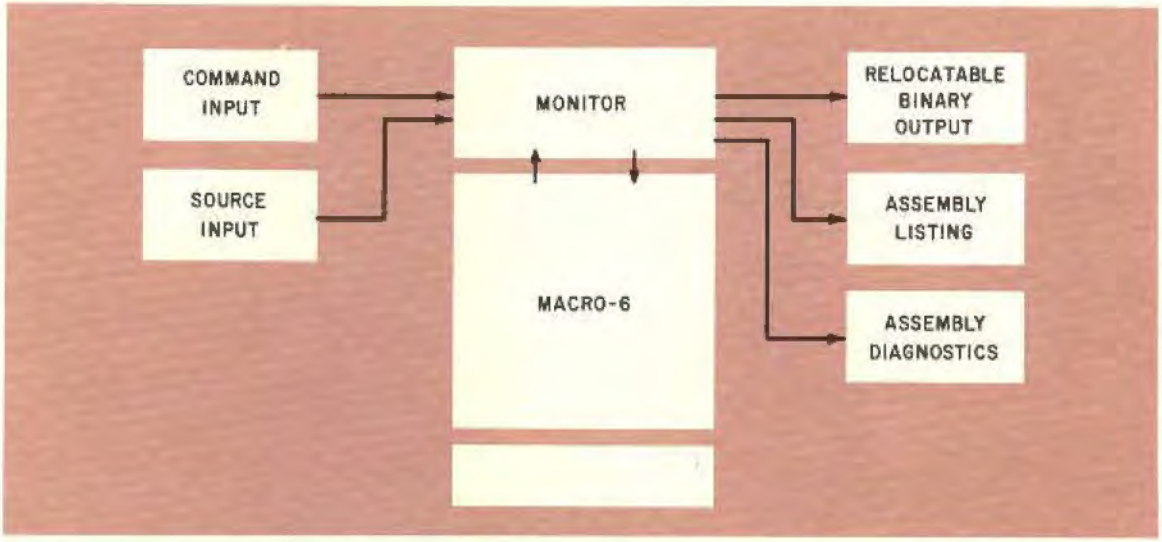
CORE SIZE:	10K to 22K	**OUTPUT:**	Relocatable binary code on DECtape, magnetic tape, or paper tape. Symbolic listing and symbol table on any system defined output device. Diagnostics on listing.
COMMAND FORMAT:	Binary file, listing ←Source file (input switch)		
INPUT:	Any defined system input device	**REFERENCE:**	FORTRAN II Language (DEC-6-0-TP-FII-LM-FP-PRE00)

```
IJOB

CORE 9

GET DTA0:MACRO
JOB SETUP

START

MTA4:OBJ,LPT:←DTA3:SOURCE

END OF PASS 2

↑C

KJOB
```

```
000000  213060 000017    SQR1:   MOVMS   A
000001  323740 000024'           JUMPLE  A,SQ2
000002  244740 777745            ASHC    A,-33
000003  275740 000201            SUBI    A,201
000004  241740 777777            AOJ     A,-1
000005  342740 000023'           HRRM    A,SQ1
000006  242740 777735            LSH     A,-43
000007  240000 777770            ASH     B,-10
000010  133017 000177            FSC     B,177(A)
000011  202000 000031'           MOVEM   B,ST
000012  160017 000025'           FMP     B,S1(A)
000013  140017 000027'           FAD     B,S2(A)
000014  200740 000031'           MOVE    A,ST
000015  170740 000000            FDV     A,B
000016  140000 000017            FAD     B,A
000017  132000 777777            FSC     B,-1
000020  200740 000031'           MOVE    A,ST
000021  170740 000000            FDV     A,B
000022  140740 000000            FADM    A,B
000023  132740 000000    SQ1:    FSC     A,0
000024  265040 000000    SQ2:    POPJ    P,

000025  200640 000000    S1:     0.8125
000026  200450 000000            0.578125
000027  177665 777635    S2:     0.382734
000030  177600 000000            0.421875

000031  000000 000000    ST:     0

                                 A=17
                                 P=1
                                 B=0

                                 ENTRY   SQRT

                                 END

THERE ARE NO ERRORS

 PROGRAM BREAK IS 000032

   SYMBOL TABLE

A               000017
B               000000
P               000001
S1              000025'
S2              000027'
SQ1             000023'
SQ2             000024'
SQRT            000000' (NT
ST              000031'

END OF ASSEMBLY
```

MACRO-6 ASSEMBLER

Sophisticated 2-pass assembly program
Accepts input from and directs output to any I/O device
Complete MACRO facilities
Produces machine language compatible with Linking Loader and DDT-6
Address arithmetic, automatic assignment of literals, text and byte manipulation



The primary function of MACRO-6 is to allow mnemonic instruction codes and programmer-created symbolic locations to be used in place of direct machine language. Features include address arithmetic, automatic assignment of program constants and temporary storage, and input of alphanumeric data. Macro instructions may be used as abbreviations for common sequences of code or for assembling complex word formats.

In addition, numbers may be expressed as binary, octal, decimal, or floating point. Text may be placed in a binary program by the use of the ASCII data generating statement. BYTE will cause a string of bytes to be assigned and packed into a word.

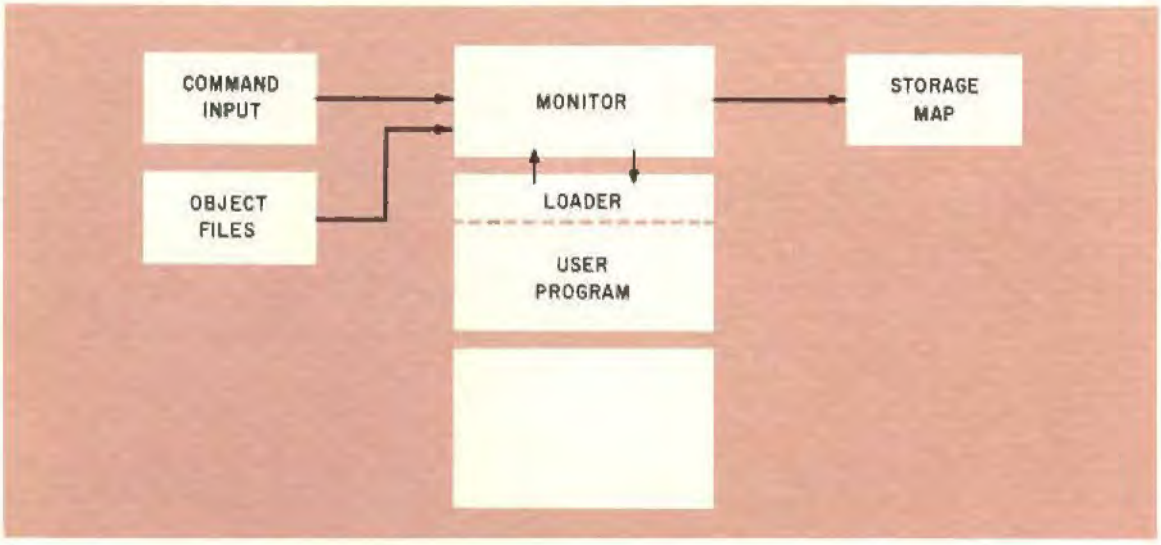
The command sequence at left illustrates an assembly from an edited file from DECtape with binary output on magnetic tape and listing on the line printer. The printout at left illustrates the format of the assembler listing and symbol table printed by the line printer.

SIZE:	9 x 1024 memory locations	**OUTPUT:**	Relocatable machine language on DECtape, magnetic tape, or punched paper tape. Symbolic listing, symbol table, and diagnostics on any output device
COMMAND FORMAT:	Binary output file, listing ←Source file 1, ... Source file N		
INPUT:	Source data from any system-defined input device	**REFERENCE:**	MACRO-6 Assembly Language Manual (DEC-6-0-TP-MAC-LM-FP-ACT01)

```
IJOB
CORE 4
GET DTA0:LOADER
JOB SETUP
START
TTY:←DTA4:(S)PG1,PG2,(WL)LB1,ARTLIB,(N)TSTDAT,(M)
STORAGE MAP
LOG         016602    000040
LOG         016602
SIN         016642    000063
COS         016645
COSD        016642
SIN         016646
SIND        016643
ATAN        016725    000061
ATAN        016725
FORSE.      017006    002031
CHINN.      017676
DEPOT.      020003
FORSE.      017006
IIB.        020012
LOADER FINISHED
START
↑C
KJOB
```

THE LINKING LOADER

Automatically loads and relocates programs produced by MACRO-6 or FORTRAN II
Produces storage map for user if desired
Creates symbol table in core for DDT if desired
Upon request will do library search
Storage used by loader recovered after loading



The Linking Loader accepts programs in a form produced by MACRO-6, FORTRAN II, and other system translators, and loads them in the user's memory area to be run. Through the use of control mode characters, the loader performs several special functions. Some of the control modes available are:

S	load local symbols for symbolic debugging with DDT
L	search files in library search mode
M	print a storage map
W	stop loading symbols
N	stop library search mode

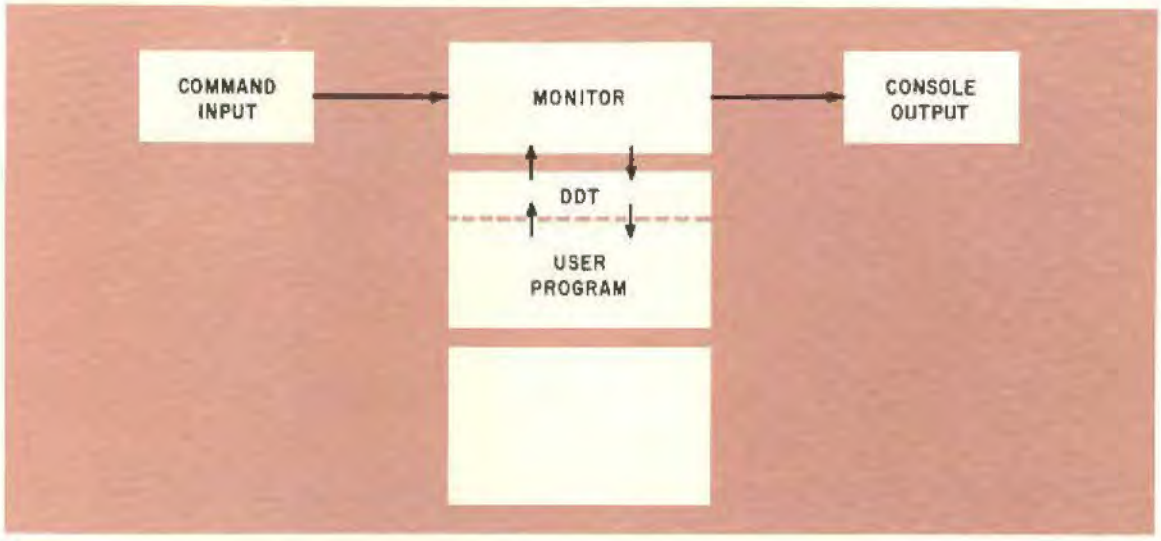
The printout on the facing page shows the commands necessary to load and run an object program. This sequence of commands loads from DTA4 files PG1 and PG2 with local symbols, LB1 and ARTLIB without locals as library files, and TSTDAT without local symbols. The user Teletype is specified for the storage map. The names with two numbers opposite are program names. The first number is the first location in the program, while the second number is the length of that program. Both numbers are octal. The names with one number opposite are global symbols and the number is the octal definition of the symbol.

SIZE:	1 x 1024 memory locations	**OUTPUT:**	Object files loaded into core memory. Storage map on the user-specified output device (if requested).
COMMAND FORMAT:	Storage map file ←(Mode) Source file 1, ... (Mode) Source file N.	**REFERENCE:**	Multiprogramming System Manual (DEC-6-0-EX-SYS-UM-IP-PRE00) Chapter VI, The Loader.
INPUT:	Object files from any system-defined input device.		

`IJOB`	Initialize the Monitor.
`CORE 4`	Command to Monitor to assign 4K of core memory.
`GET DTA0:LOADER` `JOB SETUP`	Command to get the Loader from the System tape (DTAO).
`TTY:←DTA0:DDTSYM,(I)USRDDT`	This command causes the Loader to put DDT-6 in core memory.
`LOADER FINISHED`	Typed by the Loader to indicate loading is complete.
`DDT`	Starts DDT.
`4000/ MOVE AC,L`	Typing a symbolic memory address followed by a forward slash causes the contents of the addressed register to be typed out on the teleprinter.
`X/ ADD 3,M  MOVE 1,A`	To change the contents of a particular address (e.g., register X) "open" register X by typing X/; typing the new information (here: MOVE 1, A) and a carriage return will cause the contents of X to become MOVE 1, A.
`5000$G`	A DDT-6 user may start his program at any address by typing the address (here: 5000) and following it by $G.
`5000$X;`	Use of convention shown defines the symbol X to be 5000. Location 5000 can henceforth be referred to using the symbol X. (The $ separates 5000 and X and the semicolon causes X to be defined as 5000 octal.)
`Y U`	If a symbol has not yet been defined, DDT will type a U (Undefined) following that symbol. The user may then define that symbol by the procedure described above.
`4000$B`	Typing an address (either numeric or symbolic) followed by $B inserts a breakpoint at that address. When the user's program attempts to execute the instruction at that address (4000) a program break occurs. Control returns to DDT.
`$P`	Typing this command restarts the program from the last breakpoint.

DYNAMIC DEBUGGING TECHNIQUE (DDT-6)

Monitors the status of a running program
Enables user to modify program instructions or data at any point during run time
User may stop program at predetermined points
Input-Output modes include:
symbolic, mnemonic, floating point, numbers in any radix greater than 1



DDT-6 is a powerful and easy-to-use on-line symbolic debugging system. DDT has a substantial turn-around time advantage over off-line debugging. Information required from the computer for debugging purposes is greatly reduced. DDT-6 can be used by programmers to pinpoint disastrous errors before the situation requires an all-of-core octal dump to recover any remaining traces of information. DDT-6 allows programmers to make symbolic changes to a program during run time. Insertion and deletion of instructions is a normal procedure for the DDT-6 user.

It is possible for several users to concurrently use DDT-6 to check out and run programs written in assembly language or FORTRAN at the symbolic assembly language level. The PDP-6 Time-Sharing Monitor provides the dynamic interaction needed to quickly debug a program without the high costs normally associated with on-line debugging with large systems.

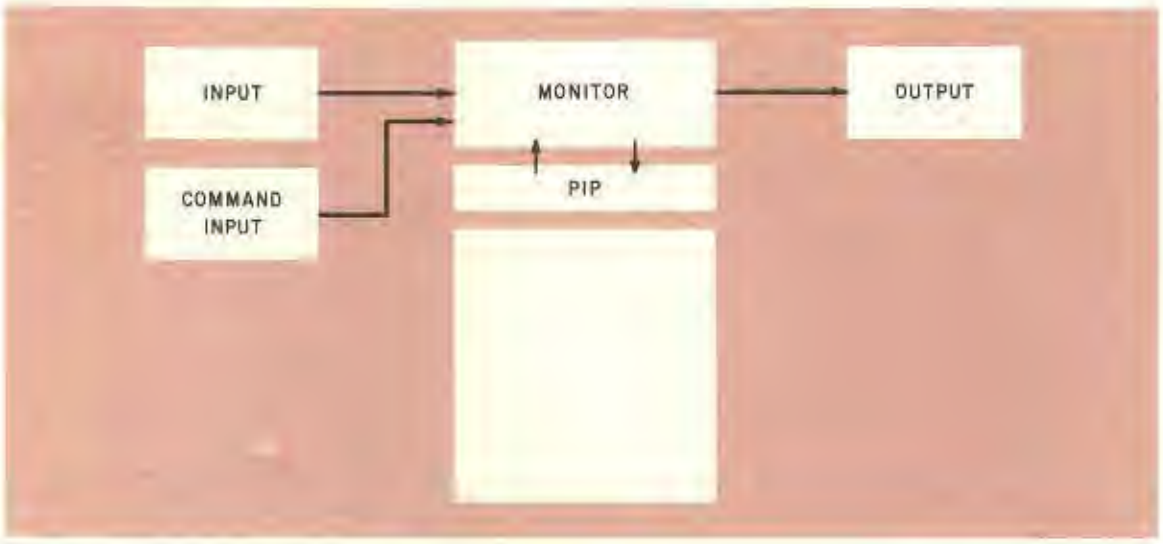
The PDP-6 Time-Sharing System allows the user to call upon DDT-6 whenever a debugging situation presents itself. More than 50 DDT-6 commands are available. Most commands are a single character to speed typing and to reduce chances of error. Several sample DDT-6 commands are shown at the left.

CORE SIZE:	3 x 1024 memory locations	**OUTPUT:**	Teletype and modified program in core
INPUT:	Control from user and/or main console	**REFERENCE:**	DDT-6 (DEC-6-0-UP-DDT-UM-FP-ACT-00)

`IJOB`	Start job
`CORE 1`	Assign 1024 words of core for PIP.
`GET DTA0:PIP` `JOB SETUP`	Requests to the Monitor to load PIP from the system tape (DTAO). The Monitor signal that PIP is in core.
`START`	Start up PIP to receive commands.
`LPT:←DTA1:XYZ,ABC.EXT`	Two files, XYZ and ABC.EXT on DECtape unit 1, are to be listed on the line printer by PIP.
`DTA2:SUBR/S←PTR:`	A file named SUBR is created on DECtape unit 2. The input data received from the paper tape reader (PTR) is transferred to DECtape as seven-bit ASCII characters with sequence numbers to be added by PIP.
`DTA1:(D)←ABC,TXT,XYZ`	Delete the files named ABC.TXT, XYZ.
`↑C`	
`KJOB`	

PERIPHERAL INTERCHANGE PROGRAM (PIP)

Performs I/O transfers and conversions between PDP-6 peripherals
Eliminates the requirement for a satellite computer
Transfers all data formats
Sequences a file of information
Suppresses trailing spaces from cards to speed later processing



The Peripheral Interchange Program performs any of the media conversions normally performed by an off-line peripheral computer. PIP operates concurrently with other PDP-6 jobs which are in the compilation or execution phase. Concurrent operation is made possible by the extremely flexible PDP-6 priority interrupt system.

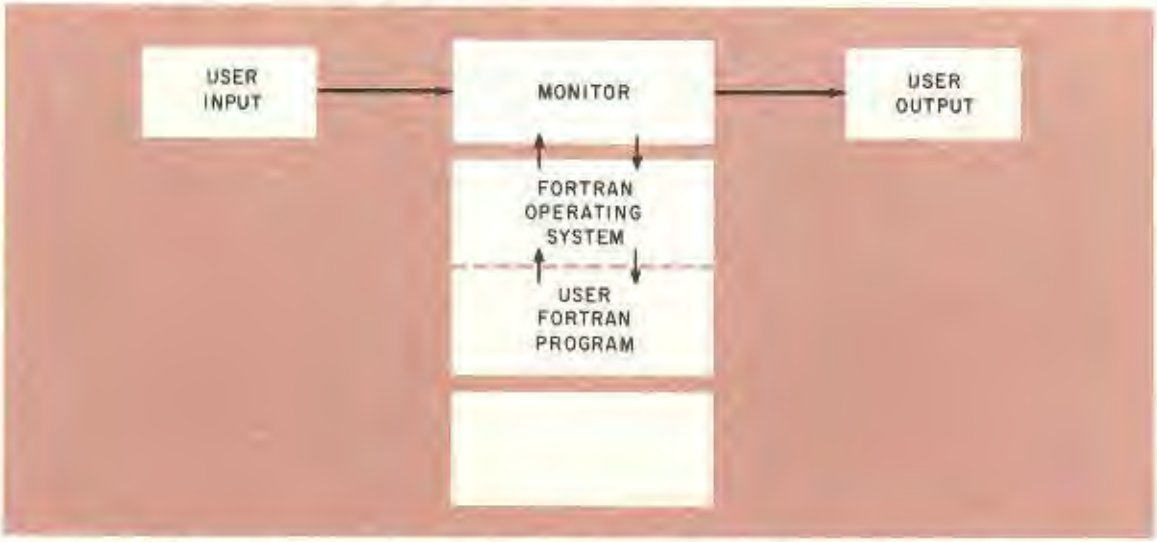
Commands for directing the information transfer operations of PIP name a source file and a destination file. The command form, from left to right,is: the output destination specification, the special character (←), and the input source specification. More than one input specification is allowed in a data transfer command. Special command qualifiers are used to specify modes of input or output. These include a binary mode for transfers of binary data, a mode for removing trailing blanks from card images, and a sequence mode for adding line sequence numbers to alphanumeric data.

An example of how the Peripheral Interchange Program is used from a console is illustrated by the printout and explanation on the facing page.

CORE SIZE:	1 x 1024 memory locations	**OUTPUT:**	Any output device the PDP-6 Monitor has available for use.
COMMAND FORMAT:	Destination file ←Source 1, ... Source N.	**REFERENCE:**	Multiprogramming System Manual (DEC-6-0-EX-SYS-UM-IP-PRE00) Chapter VIII, the Peripheral Interchange Program (PIP).
INPUT:	Source data from any input device the PDP-6 Monitor has available for use.		

FORTRAN OPERATING SYSTEM AND LIBRARY

Complete arithmetic package
Utility package including dump and chaining facilities
Directs output to any I/O device
Arithmetic functions and I/O conversion routines available to both FORTRAN and non-FORTRAN programs
Variable length input
I/O devices may be assigned at run time



The FORTRAN operating system scans formats and directs input-output between user FORTRAN programs and the Monitor. The operating system requires 1,500 memory locations and must be loaded with a FORTRAN program for execution.

The FORTRAN library is listed in the accompany table. These functions may also be called by machine language programs.

FORTRAN Library

Entry	Function	Size (octal)
EXP. 1	Integer number to integer power	15
EXP. 2	Floating point number to integer power	17
EXP. 3	Floating point number to floating point power	3
ACOS	Arc cosine	5
ASIN	Arc sine	11
SQRT	Square root	32
SINH	Hyperbolic sine	6
COSH	Hyperbolic cosine	6
TANH	Hyperbolic tangent	44
EXP	Exponential	52
LOG, LOG10	Logarithm base e, logarithm base 10	40
SIN, COS, SIND, COSD	Sine, cosine (radians) Sine, cosine (degrees)	63
ATANZ	Arc tangent of two arguments (y, x)	12
ATAN	Arc tangent	61
FLOAT	Float a fixed point number	6
XFIX	Fix a floating point number	5
INT	Floating point truncation	12
XINT	Floating point truncation (integer result)	11
MOD	Floating point remaindering function	12
DUMP, PDUMP	Selective core dump	250
CHAIN	Chaining routine for multiple core loads	200

DIGITAL SALES AND SERVICE

MAIN OFFICE AND PLANT

146 Main Street, Maynard, Massachusetts 01754
Telephone: From Metropolitan Boston: 646-8600
Elsewhere: AC617-897-8821
TWX: 710-347-0212 Cable: Digital Mayn. Telex: 092-027

DIGITAL SALES OFFICES

NORTHEAST OFFICE:
146 Main Street, Maynard, Massachusetts 01754
Telephone: AC617-646-8600 TWX: 710-347-0212

NEW YORK OFFICE:
1259 Route 46, Parsippany, New Jersey 07054
Telephone: AC201-335-0710 TWX: 510-235-8319

WASHINGTON OFFICE:
Executive Building
7100 Baltimore Ave., College Park, Maryland 20740
Telephone: AC301-779-1100

SOUTHEAST OFFICE:
Suite 91, Holiday Office Center
3322 Memorial Parkway, S.W., Huntsville, Ala. 35801
Telephone AC205-881-7730 TWX: 205-533-1267

ORLANDO OFFICE:
1510 E. Colonial Drive, Orlando, Florida 32803
Telephone: AC305-422-4511 TWX: 305-275-0641

PITTSBURGH OFFICE:
300 Seco Road, Monroeville, Pennsylvania 15146
Telephone: AC412-351-0700 TWX: 710-797-3657

CHICAGO OFFICE:
910 North Busse Highway, Park Ridge, Illinois 60068
Telephone: AC312-825-6626 TWX: 312-823-3572

DENVER OFFICE:
Suite 205
5200 South Quebec Way, Englewood, Colo. 80110
Telephone: AC303-771-1180 TWX: 910-444-2212

ANN ARBOR OFFICE:
3853 Research Park Drive, Ann Arbor, Mich. 48104
Telephone: AC313-761-1150 TWX: 810-223-6053

LOS ANGELES OFFICE:
8939 Sepulveda Boulevard, Los Angeles, Calif. 90045
Telephone: AC213-670-0690 TWX: 910-328-6121

SAN FRANCISCO OFFICE:
2450 Hanover, Palo Alto, California 94304
Telephone: AC415-326-5640 TWX: 910-373-1266

IN CANADA:
Digital Equipment of Canada, Ltd.,
150 Rosamund Street, Carleton Place, Ontario, Canada
Telephone: AC613-237-0772 TWX: 610-561-1650

IN EUROPE:
Digital Equipment GmbH, Theresienstrasse 29
Munich 2/West Germany
Telephone: 29 94 07, 29 25 66 Telex: 841-5-24226

Digital Equipment Corporation (UK) Ltd.
11 Castle Street
Reading, Berkshire, England
Telephone: Reading 57231 Telex: 851-84327

IN AUSTRALIA:
Digital Equipment Australia Pty. Ltd.,
89 Berry Street
North Sydney, New South Wales, Australia
Telephone: 92-0919 Telex: 790AA-20740
Cable: Digital, Sydney

DIGITAL SALES REPRESENTATIVES

IN THE SOUTHWEST:

DATRONICS INC.

7800 Westglen Drive, Houston, Texas 77042
Telephone: AC713-782-9851 TWX: 713-571-2154

DATRONICS INC.

Post Office Box 782, Kenner, Louisiana 70062
Telephone: AC504-721-1410

DATRONICS INC.

Post Office Box 13384, Fort Worth, Texas 76118
Telephone: AC817-281-1284 TWX: 817-281-3120

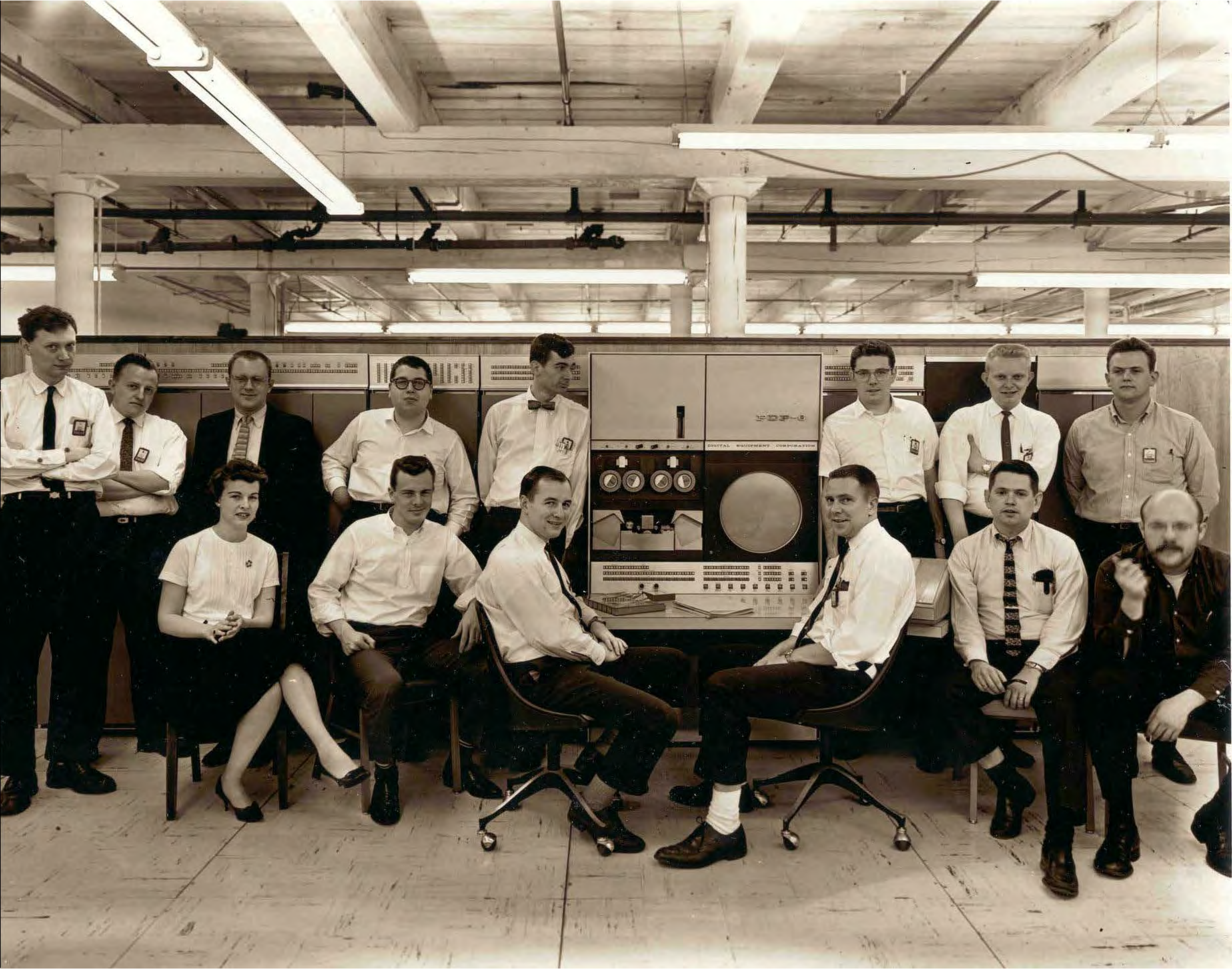
IN THE NORTHWEST:
SHOWALTER-JUDD, INC.
1806 South Bush Place, Seattle, Washington 98144
Telephone: 206-324-7911 TWX: 910-444-2212

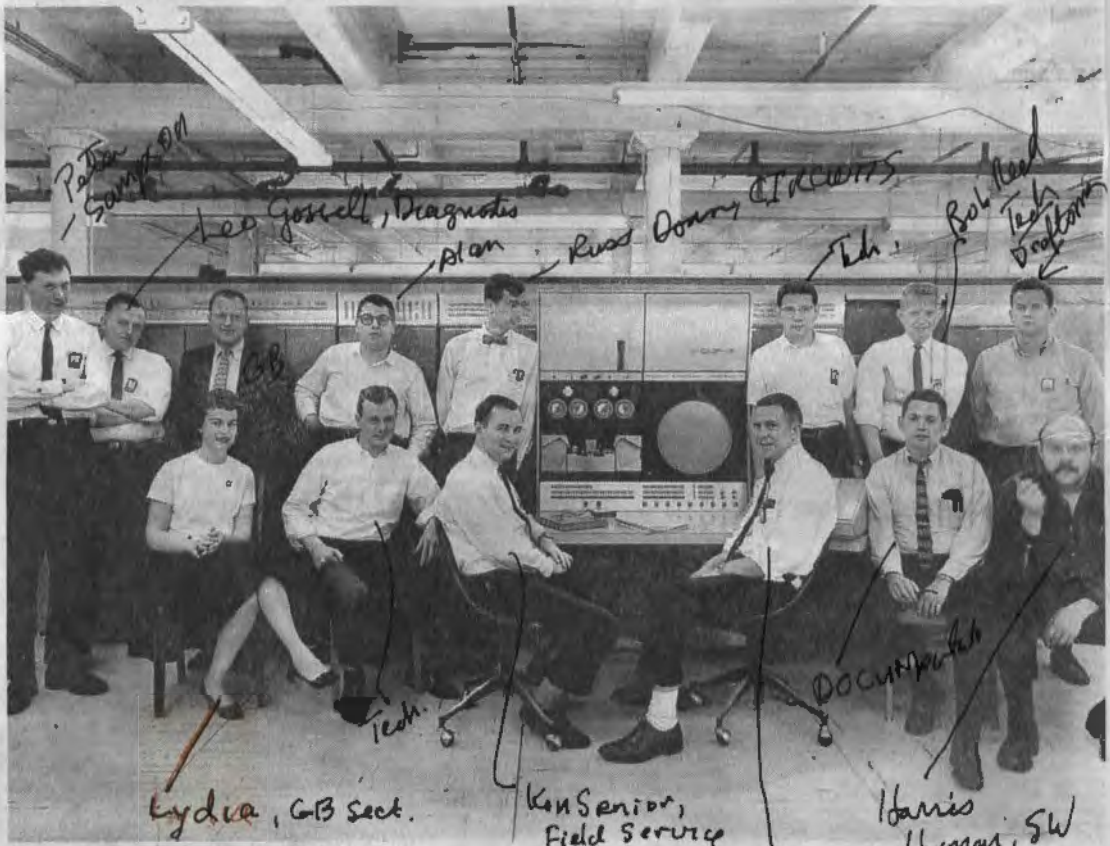
IN JAPAN:
RIKEI TRADING CO.,
12, 2-Chome, Shiba Tamura-cho, Minato-ku,
Tokyo, Japan
Telephone: 591-5246 Cable Rikeigood, Tokyo
Telex: 781-TK-4208 RIKEI

IN SWEDEN:
TELARE AB
Industrigatan 4, Stockholm K, Sweden
Telephone: 54 33 24 Telex: 854-10178

digital

5360 Printed in U.S.A. 10-5/65


PDP-6
DIGITAL EQUIPMENT CORPORATION



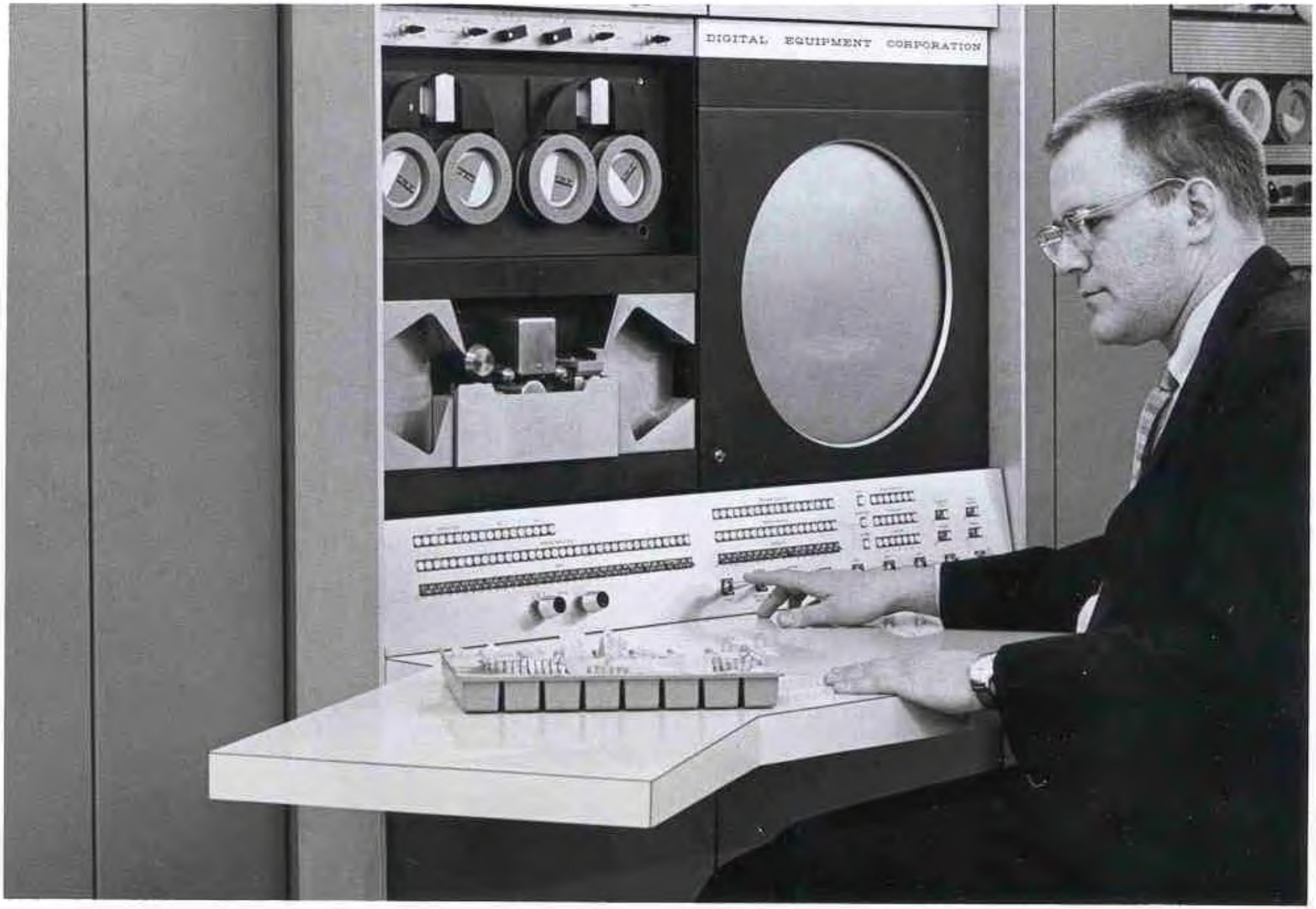
Alan Kotok (fourth from left in back), with fellow designers of the PDP, one of Digital Equipment Corp.'s early computers.

Figure 4.4 Moore poses with a broom at DEC in 1965. Dr. R. Smart, then General Manager of DEC Australia, stands to his left (foreground). Deputy Vice-Chancellor Birkett-Clews stands in the background.

Source: The University of Western Australia Computing Service.


555


555


DIGITAL EQUIPMENT CORPORATION

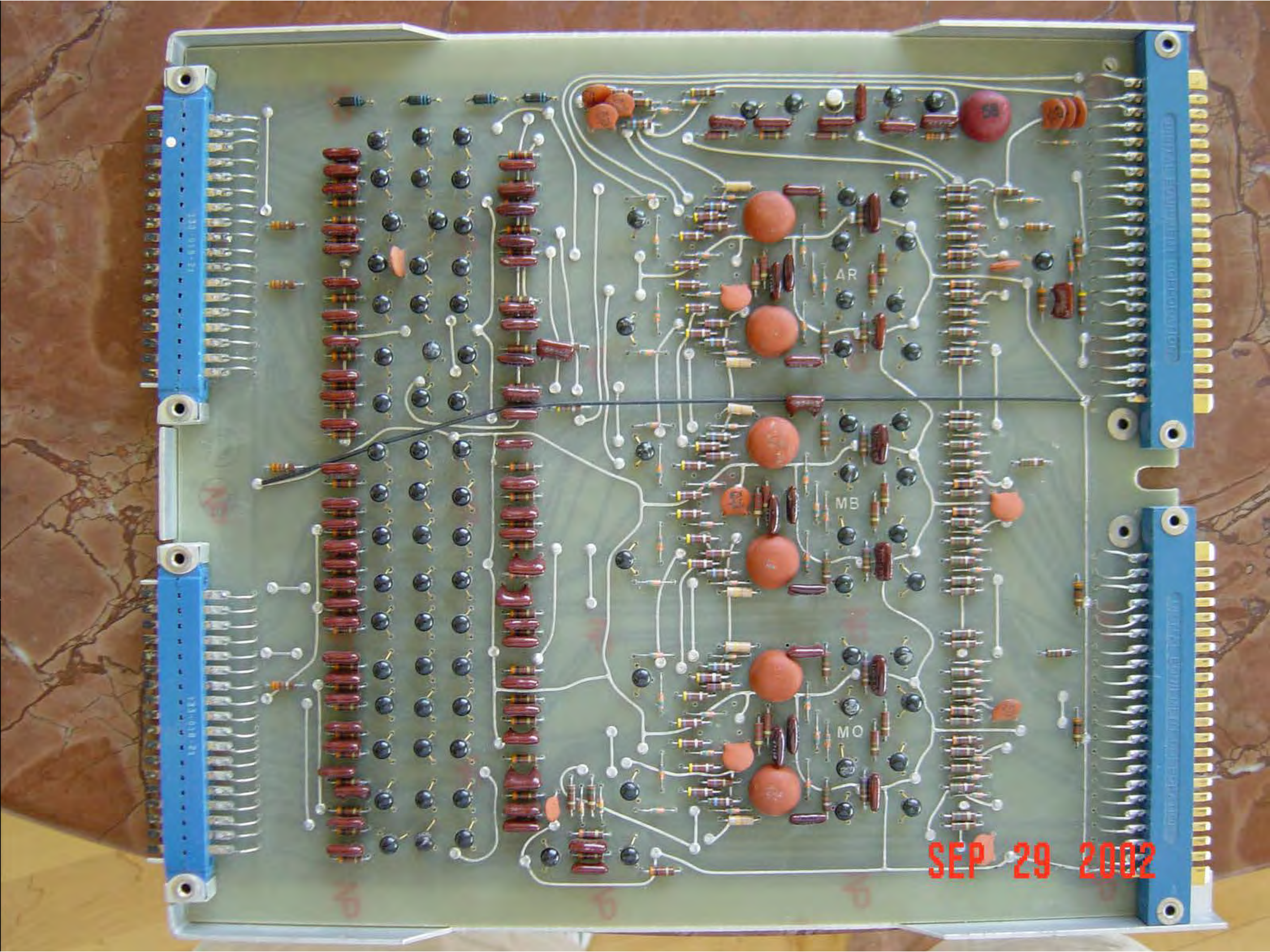
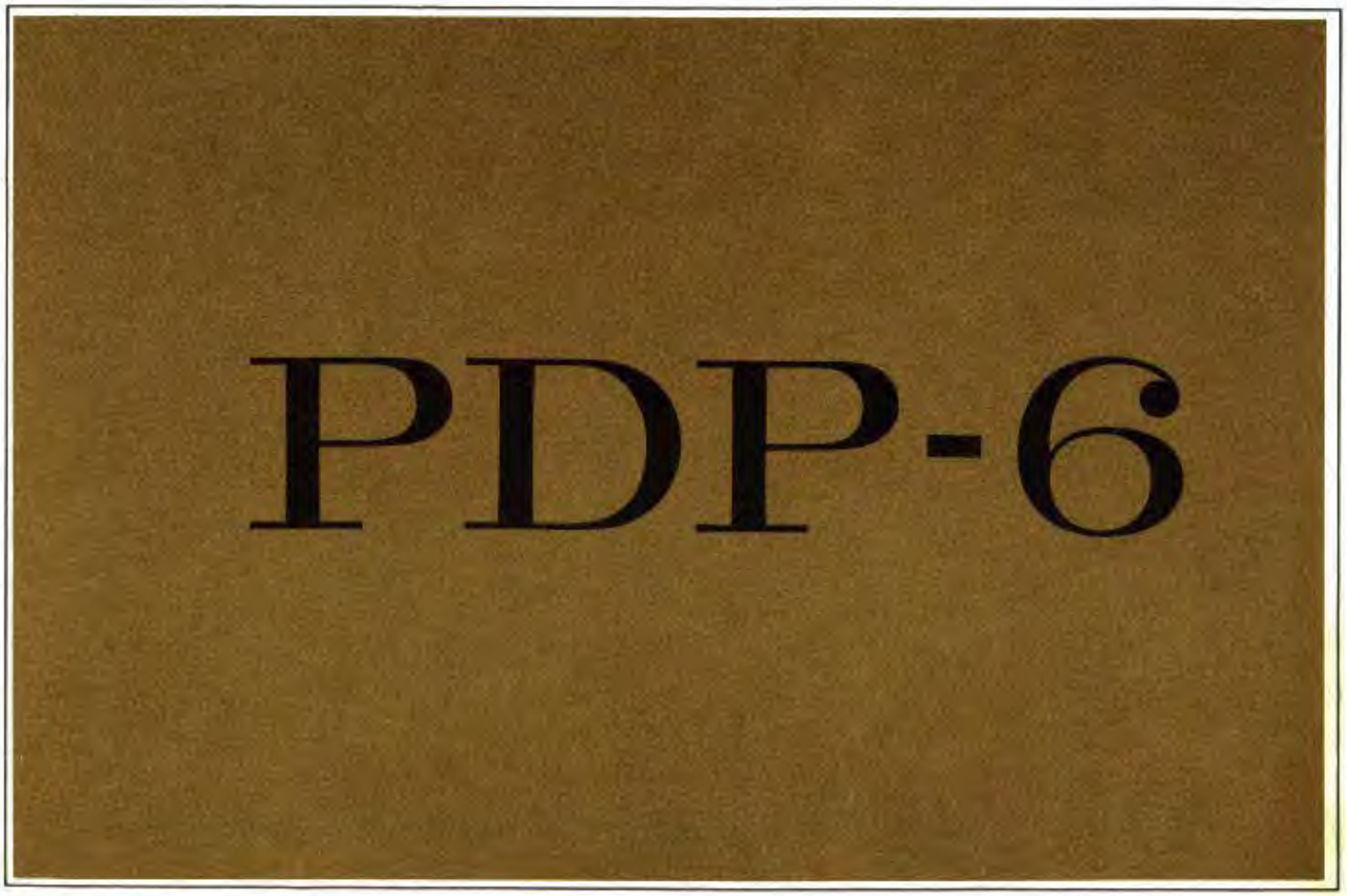

AR
MB
MO
SEP 29 2002

Figure 4.3 The UWA PDP-6 at DEC in Boston (before delivery).
Source: The University of Western Australia Computing Service.

F-65
8/64

PROGRAMMED DATA PROCESSOR-6 HANDBOOK



DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

PROGRAMMED DATA PROCESSOR-6 HANDBOOK

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

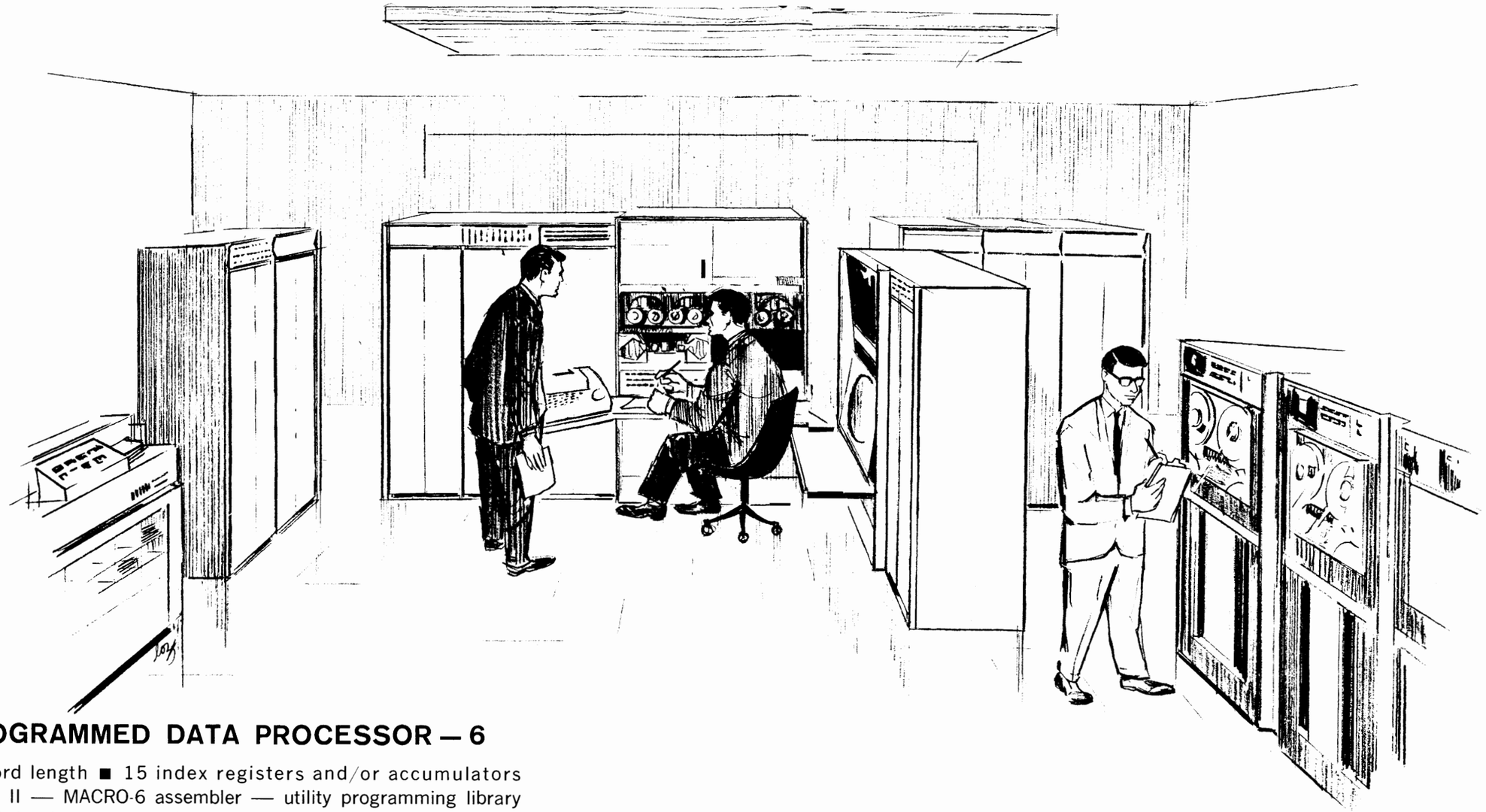


Figure 1 Typical PDP-6 System

PROGRAMMED DATA PROCESSOR — 6

36-bit word length ■ 15 index registers and/or accumulators
FORTRAN II — MACRO-6 assembler — utility programming library
Integrated hardware and software for time sharing ■ Microtape
Asynchronous operation, modular construction ■ Memory overlap
Core memories up to 262,144 words, 2 μsec, directly addressable
Fast memory 16 words, 0.4 microsecond ■ 128 input-output devices
363 instructions ■ fast floating point — multiply 14 μsec average
Program assignable operation codes ■ Byte manipulation, half word
Block transmission ■ Seven channel priority interrupt system
Programmed input-output transfers require no data channels
Multiple processors ■ Remote input-output ■ Mass memory

CONTENTS

CHAPTER 1
PDP-6 SYSTEM DESCRIPTION

CHAPTER 2
ARITHMETIC PROCESSOR TYPE 166

CHAPTER 3
MEMORY

CHAPTER 4
INPUT-OUTPUT

ILLUSTRATIONS

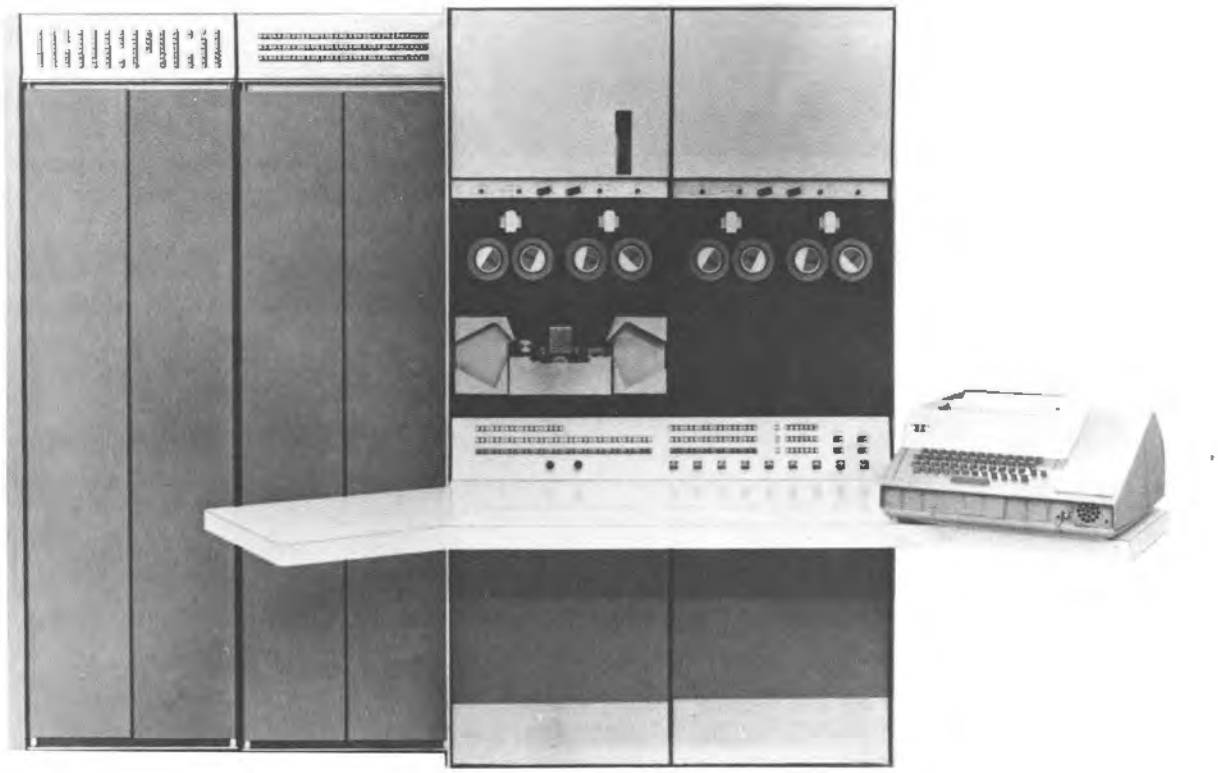


Figure 2 Arithmetic Processor Type 166

CHAPTER 1
PDP-6 SYSTEM DESCRIPTION

Programmed Data Processor-6 (PDP-6) is a general-purpose digital computing system designed for scientific data processing. The flexibility of this system permits the user to specify the data handling capacity and the exact configuration needed to meet his requirements. The system can be expanded with presently available equipment or, at a later date, with equipment yet to be developed. Faster memories, for example, can be added as they become available.

PDP-6 design eliminates the need for off-line conversion equipment. Conversion of programs from cards or paper tape to magnetic tape can be done concurrent with normal program running. Users at peripheral teleprinters can simultaneously prepare and debug their programs on line.

The PDP-6 system consists of processors, memories, and input/output devices. Since each is autonomous (no device is dependent upon another for its timing), a system configuration can include memory modules of different speeds, processors of different types sharing the same memory modules, and standard or unique input/output devices.

For maximum flexibility of system configurations, the PDP-6 system is built around two busses: processor-memory bus and processor-input/output bus. The memory bus permits each processor to directly address 262,144 words of core memory, automatically permits overlapping, and simplifies multiprocessor operation. An input/output bus of a processor can service up to 128 devices.

The operating system consists of a supervisory control program, system programs, and system subroutines. Included are a Symbolic Assembler and Macro Processor, a FORTRAN II Compiler, and debugging aids. A library of general utility programs is also provided.

Neither the processors nor any of the standard peripheral equipment require an air-conditioned environment or floor reinforcement. Ordinary 115-volt power is sufficient for all equipment.

Processors

A PDP-6 system can include any number of processors of the same or different types. The Type 166 is a 36-bit arithmetic processor with many powerful features, including 16 accumulators, 15 index registers, built-in floating point arithmetic, and byte operations capability. Memory protection and relocation registers are included for time-sharing operations.

The Type 167 I/O Processor gives direct memory access to high speed devices, such as drums, discs, and displays. It takes over local control of data transfers after receiving system commands and initial conditions from the arithmetic processor. Thereafter the two processors operate asynchronously, so that I/O transfers are carried out in parallel with arithmetic processing.

Up to three controls, such as the Type 236 Drum Control, can be connected to the Type 167 I/O Processor.

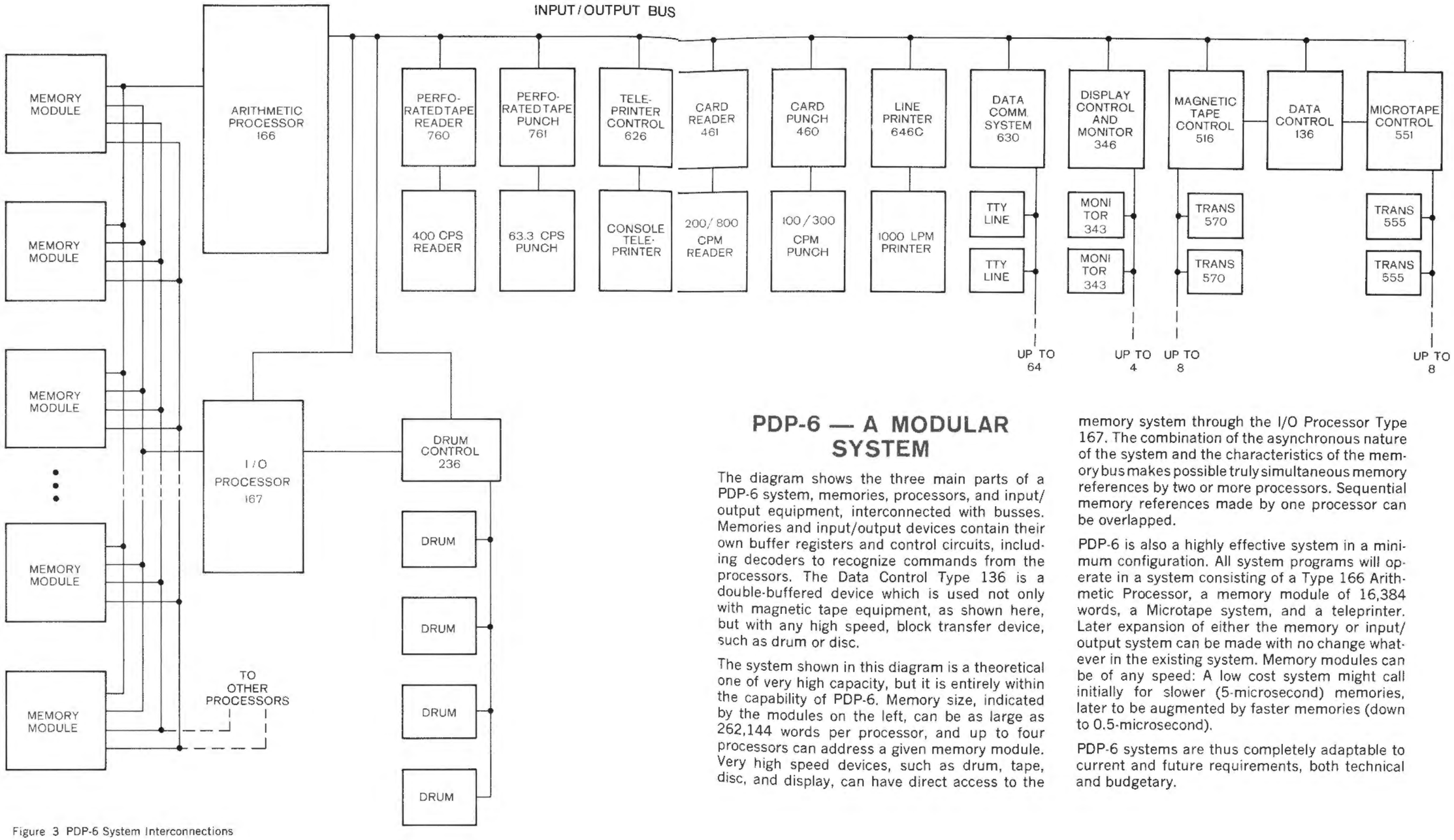


Figure 3 PDP-6 System Interconnections

PDP-6 — A MODULAR SYSTEM

The diagram shows the three main parts of a PDP-6 system, memories, processors, and input/output equipment, interconnected with busses. Memories and input/output devices contain their own buffer registers and control circuits, including decoders to recognize commands from the processors. The Data Control Type 136 is a double-buffered device which is used not only with magnetic tape equipment, as shown here, but with any high speed, block transfer device, such as drum or disc.

The system shown in this diagram is a theoretical one of very high capacity, but it is entirely within the capability of PDP-6. Memory size, indicated by the modules on the left, can be as large as 262,144 words per processor, and up to four processors can address a given memory module. Very high speed devices, such as drum, tape, disc, and display, can have direct access to the memory system through the I/O Processor Type 167. The combination of the asynchronous nature of the system and the characteristics of the memory bus makes possible truly simultaneous memory references by two or more processors. Sequential memory references made by one processor can be overlapped.

PDP-6 is also a highly effective system in a minimum configuration. All system programs will operate in a system consisting of a Type 166 Arithmetic Processor, a memory module of 16,384 words, a Microtape system, and a teleprinter. Later expansion of either the memory or input/output system can be made with no change whatever in the existing system. Memory modules can be of any speed: A low cost system might call initially for slower (5-microsecond) memories, later to be augmented by faster memories (down to 0.5-microsecond).

PDP-6 systems are thus completely adaptable to current and future requirements, both technical and budgetary.

Input/Output

The input/output bus consists of device selection, data, control, and status sense lines. A 7-channel program-assignable priority interrupt system signals the processor when input/output devices require service. Word count and memory address registers are located in the processor and are available to all devices. This reduces the cost of various input/output controls, and permits data block transfers between tapes, card readers, printers, displays, and other devices.

Memory

The PDP-6 core memory subsystem permits modular expansion using blocks of different sizes and speeds. The types 163B and 163C core memory modules contain 8,192 and 16,384 words, respectively. Each has a word length of 36 bits, a cycle time of 2 microseconds, and an access time of 0.8 microseconds. The Type 162 Fast Memory Module contains 16 words with a 0.4-microsecond cycle. Slower core memories, such as the 5-microsecond Type 161, can be used where economy is an overriding criterion.

The memory-processor bus permits memory cycle overlap, gives all processors direct access to memory, and permits easy expansion and modification of the memory subsystem. In addition, the bus allows the processors to remain connected to memory only as long as needed to transfer information: That is, a processor can put a word on the bus and resume operations as soon as the memory acknowledges, without waiting for the memory to store the word. Similarly, when reading a word out of memory, the processor takes the information and operates on it immediately, without waiting for the memory to finish the rewrite portion of its cycle.

Maximum system efficiency is achieved when sequential memory references address alternate memory modules. The addressed module places data on the bus as soon as it is available in the memory buffer and disconnects itself from the bus while rewriting, freeing the processor to store the result or seek the next instruction in a second memory module before the first one has completed rewriting. Utilizing such overlapping memory references, PDP-6 users can effectively cut in half the time required for average random accesses. Multiple connections between the bus and each memory module permit module sharing on a priority basis for multi-processor operations.

PROGRAMMING SYSTEM

The programming system for PDP-6 consists of a supervisory control program, system programs, and library routines. The entire system is designed to run on any PDP-6 system with at least 16,384 words of core memory, a console teleprinter, and a Microtape system. However, the programming system is modular. Parts of it can run on machines with smaller memory capacities. For example, programs can be assembled with MACRO6 using the above-mentioned input/output equipment and only 8,192 words of core memory.

A variety of programs are provided through the Digital program library, and a continuous in-house program design effort regularly improves and expands the library.

The Supervisory Control Program

This is the name given to a collection of programs remaining permanently in memory to provide overall coordination and control of the total operating system. The segments of the program are:

COMMAND CONTROL PROGRAM handles all commands addressed to the system from the user-consoles. These commands would include requests to log in or out, a request to use the edit program, requests to have a program placed on the run queue, requests to load a program, etc.

PROGRAM SCHEDULER is called at regular intervals to decide which program in memory is to be run. A running program is temporarily terminated each time its allotted time has run out or when it requires input/output operations with a device that is busy. A program may be terminated temporarily by user intervention to the scheduler, or it may suspend its own operation. Temporary termination does not remove the program from memory. A program may be dumped on backing storage and permanently discontinued by calling the scheduler and allocator.

FACILITIES ALLOCATOR is called any time an I/O device or memory space is required. It may be called from a user-console or by a running program. Under this program one user-console is designated the operator console. As such it has special facilities available which are not available to other consoles, such as line printer assignments. Storage is permanently assigned for all resident programs, that is, those programs that are in memory at all times. Finally, "logical" tape assignments are made. Two Microtape units are designated the system library and the system scratch tape. Two other tapes may be assigned as peripheral input tape, used to prepare jobs to be stacked from cards or paper tape, and system input tape, used to input a full tape. (Each user-console may require a Microtape unless programs requiring files are to be run.)

COMMAND DECODER preprocesses commands from the user-console. This program is used to convert parameters, etc., before the command is sent to the program for which the command is intended.

I/O CONTROL is called whenever an I/O device is to be used. This program assigns equipment, controls the I/O devices, controls data transfers between memory and the I/O device, and controls the buffering of data for the device. (Users provide the necessary buffering for devices which require file buffering.) All program I/O instructions are trapped, and the actual control of the I/O operation then passes to the I/O Control Program.

System Programs

These are the programs designed to implement system functions which may be requested from the user-console. This is in contrast to system subroutines which may be called by system programs or other programs. System programs are normally provided by Digital, but they may be provided by each installation for its users. The programs contain a mode by which they may be terminated to return the communication link to the system. Some of the system programs are described below.

EDITOR PROGRAM manipulates the text of a named file on a Microtape or in the user area of the drum (corresponding to Microtape). This file may be used for the creation of text or for later use as data or as a program to be translated by the FORTRAN Compiler, etc. The commands provided for the editor allow text to be created, deleted, or moved about.

PERIPHERAL CONVERSION PROGRAM handles all those jobs normally done by a separate peripheral processor. The priority interrupt system and multiple memory accumulators in the PDP-6 eliminate virtually all loss in running time. Such processing is done through the arithmetic processor.

INTER-CONSOLE MESSAGE PROGRAM switches message traffic between the various user-consoles. This program allows the user to request manual operations by the operator and receive acknowledgment. Such an operation would be the mounting/dismounting of user tapes.

LINKING LOADER PROGRAM accepts programs in a form produced by the translators, and produces an area of core memory loaded with the program. Upon request, it may also produce a storage map of the loaded programs along with symbol tables. Several programs may be linked together in loading. The loader requests special library tapes to be loaded, and verifies that the program has been completely loaded.

TRANSLATOR DISPATCHER is called to load the FORTRAN, MACRO6, or other translators. The translators are rather large programs that do not reside in memory, but are stored on the System Library tape until they are called into memory by the translator.

FORTRAN II COMPILER accepts FORTRAN II input statements and produces relocatable binary output coding for later loading by the Linking Loader. Compiling is done in one pass. PDP-6 FORTRAN II is an extension of the conventional FORTRAN II language to give the user more facilities and to take advantage of PDP-6 hardware. The ASCII character set is used. Subscripts may consist of statements (fixed or floating). Any number of dimensions may be used to specify an array. Signed integers have 36-bit values, but when used as subscripts are truncated to 18 bits.

MACRO6 ASSEMBLY PROGRAM translates MACRO6 input language to a relocatable binary output for the Linking Loader. MACRO6 is a two-pass assembly program, and the language provides for macro instruction definitions and usage. Literal assignments are made by brackets [ ]. Numbers may be expressed as binary, octal, decimal, and floating point. Text may be placed in a binary program by the occurrence of the "text" data generating statement, and "byte" causes a string of bytes to be assigned and packed into a word. The "repeat" control statement causes the statements following the control to be repeated "n" times.

DEBUGGING PROGRAM (DDT) is loaded with a program and allows all assembly language programs to be debugged. The program may be started or stopped, words in the program may be modified, and DDT may search the program looking for particular words. DDT may also be used in a "trace" or break point mode, and the program is run until a particular location (a break point) is encountered.

THE SYSTEM SUBROUTINES include:

1. I/O Format Control which provides for the various format statements used in the FORTRAN II language. These subroutines are also available to other programs and may be called from the systems library tape.

2. Arithmetic Subroutines which include all the arithmetic subroutines required for FORTRAN II, such as, sine, cosine, $\log_E$, $\log_{10}$, exponent, tangent, arctangent, and square root.

INPUT/OUTPUT EQUIPMENT

Digital offers a large selection of optional equipment for full utilization of the extensive input/output capacity of the system.

MICROTAPE TRANSPORT TYPE 555

A fixed-address magnetic-tape facility for high speed loading, readout, and on-line program debugging. Read, write, or search speed is 80 inches a second. Density is 375 bits an inch. Total storage is three million bits. Features phase recording, rather than amplitude recording; redundant, nonadjacent data tracks; and a pre-recorded timing and mark track.

MICROTAPE CONTROL TYPE 551

Controls up to eight Type 555 Microtape Transports. Searches in either direction for specified block numbers, then reads or writes data. Uses the Type 136 Data Control to assemble data and buffer transfers to the processor.

DATA CONTROL TYPE 136

Provides for assembly of 6, 12, 18, or 36-bit characters; six input/output devices can be controlled.

TELEPRINTER AND CONTROL TYPE 626

Permits on-line programming and debugging. Provides hardcopy outputs. Is standard Teletype equipment, operating at ten characters a second.

TELEPRINTER INTERFACE TYPE 630

Automatically scans up to 64 teleprinter (TTY) lines. Signals a program interrupt when teleprinter needs service.

CARD PUNCH CONTROL TYPE 460

Permits on-line punching of cards in any format, including IBM, at 100 or 300 cards a minute.

CARD READER AND CONTROL TYPE 461

Provides on-line reading of standard punched cards at 200 or 800 cards a minute in alphanumeric or binary codes.

HIGH SPEED PERFORATED TAPE PUNCH AND CONTROL TYPE 761

Punches 8-hole tape at 63.3 characters a second.

HIGH SPEED PERFORATED TAPE READER AND CONTROL TYPE 760

Reads perforated paper tape photoelectrically at 400 characters a second.

MAGNETIC TAPE CONTROL TYPE 516

Automatically controls up to eight tape transports Type 570 or IBM 729 series. Permits reading, writing, forward/backward spacing, rewind and unload, and rewind. Uses a Type 136 Data Control to assemble data and buffer transfers to the processor. Longitudinal and lateral parity are checked.

MAGNETIC TAPE TRANSPORT TYPE 570

Tape motion is controlled by pneumatic capstans and brakes, eliminating conventional pinch rollers, clamps, and mechanical arms. Tape speed is either 75 or 112.5 inches a second. Track density, program-selectable, is 200, 556, and 800

bits per inch. Tape width is one-half inch, with six data tracks and one for parity. Format is compatible with IBM NRZI. Dual heads permit read checking while writing.

I/O PROCESSOR TYPE 167

Establishes a data transmission path between main memory and the drum(s). Up to four drums can be connected to the I/O processor.

MAGNETIC DRUM AND CONTROL TYPE 236

Drum stores 1,048,576 36-bit words organized into 128 tracks, each with 8192 words consisting of 64 128-word blocks. A word is transferred in 6.4 microseconds, and the drum revolution time is 52 milliseconds.

DISPLAY CONTROL AND MONITOR TYPE 346

Plots points, lines, vectors, and characters on a 9⅜-inch-square raster of 1024 points along each axis. Time between points plotted is 1.5 microseconds in the vector, increment, and character modes. In random point plotting, a time of about 35 microseconds is required per point.

DISPLAY MONITOR TYPE 343

Provides additional cathode ray tube display for multiple consoles.

HIGH SPEED LIGHT PEN TYPE 370

Detects data displayed by the Types 346 and 343 and inputs identifying signal to the computer.

AUTOMATIC LINE PRINTER AND CONTROL TYPE 646C

Prints 1000 lines a minute, 120 columns a line, any one of 64 characters a column.

AUTOMATIC LINE PRINTER AND CONTROL TYPE 646A

Prints 300 lines a minute, 120 columns a line, any one of 64 characters a column.

ANALOG-TO-DIGITAL CONVERTER TYPE 138

Transforms an analog voltage to a binary number, selectable from 6 to 11 bits. Conversion time varies, depending on the number of bits and the accuracy required. Twenty-one combinations of switching point accuracy and number of bits can be selected on the front panel.

MULTIPLEXED ANALOG-TO-DIGITAL CONVERTER TYPE 138/139

The Type 139 Multiplexer Control permits up to 64 channels of analog information to be applied singly to the input of the Type 138 Analog-to-Digital Converter. Channels can be selected in sequence or by individual addresses.

HIGH-SPEED ANALOG-TO-DIGITAL CONVERTER TYPE 142

Transforms an analog voltage to a signed, 10-bit binary number in 6 microseconds. Conversion accuracy is $\pm 0.15\% \pm 1/2$ least significant bit.

ANALOG-DIGITAL-ANALOG CONVERTER SYSTEM TYPE ADA-1

Performs fast, real-time, data conversion between digital and analog computers. Maximum sample rate for D/A conversion is 200 kc; for A/D and interlaced conversions, 100 kc. Digital word length is 10 bits. Actual conversion times are 5 microseconds for A/D and 2 microseconds for D/A. Semiautomatic features enable the converter system to perform many of the functions that a computer normally performs for other converter interfaces.

CHAPTER 2
ARITHMETIC PROCESSOR TYPE 166

The Type 166 is a general purpose processor which performs all of the arithmetic and logical operations in a PDP-6 system and transfers data to and from input-output devices. It is connected to the memory and input-output subsystems through busses and bus interfaces.

The logic block diagram, Figure 4, shows the interconnection of the principal registers of the Type 166, and their functions are described below. An unusual feature of the 166 is that when appropriate the contents of the AR and/or the MQ are transmitted to memory at the end of each instruction. Thus the state of the machine, except for the PC, the four flags, the memory protection and relocation registers, and the executive mode status, is constantly in memory, and there are no instructions that directly reference processor registers except the PC and the flags. Using the AR for the effective address calculation will never destroy a partial result in a calculation. Further, the processor is free to carry out any operation, for example, an I/O operation called for by an interrupt.

PROCESSOR REGISTERS

MA	(Memory Address) An 18-bit register that specifies the desired location to the memory system.
MB	(Memory Buffer) A 36-bit data buffer that communicates with the memory system. The MB is also used as the addend register in arithmetic operations and contains one of the operands in logical operations.
AR	(Arithmetic Register) A 36-bit register used in indexed address calculations, as an operand register for logical instructions, and in all arithmetic and shifting instructions. The accumulator specified by the instruction is moved to the AR during most instructions.
MQ	(Multiplier Quotient) A 36-bit register used as an extension of the AR in operations requiring 72 bits.
MI	(Memory Indicator) A 36-bit register used to display the contents of memory registers. The MI register is set equal to the contents of the memory location specified by the ADDRESS switches each time the processor references the location.
PC	(Program Counter) An 18-bit register that contains the memory location from which the next instruction is to be taken.
IR	(Instruction Register) An 18-bit register that contains the instruction code, accumulator number, indirect bit, and index register number for the instruction being executed.
SC	(Shift Counter) A 9-bit register that contains the number of times a particular part of an instruction is to be repeated. This applies to the shifting, floating point, multiply, divide, and byte manipulation instructions.
FE	(Floating Exponent) A 9-bit register used to store temporary results during floating calculations.

MP (Memory Protection) An 8-bit register used by the executive system in the user mode. This register contains the largest address that can occur in a user's program, that is, the size of the block of memory allocated to the user. When making a reference to memory, the MP and MA registers are compared. If the MA is larger than the MP, the memory cycle does not take place, the memory protection flag is set, and control goes to an executive program with the computer in the executive mode.

RL (Relocation Register) An 8-bit register used by the executive system in the user mode. If the address in the MA does not exceed the largest address specified by the MP, and is greater than 17_8, the contents of the RL register are added to the high order 8-bits of the MA before the memory cycle takes place.

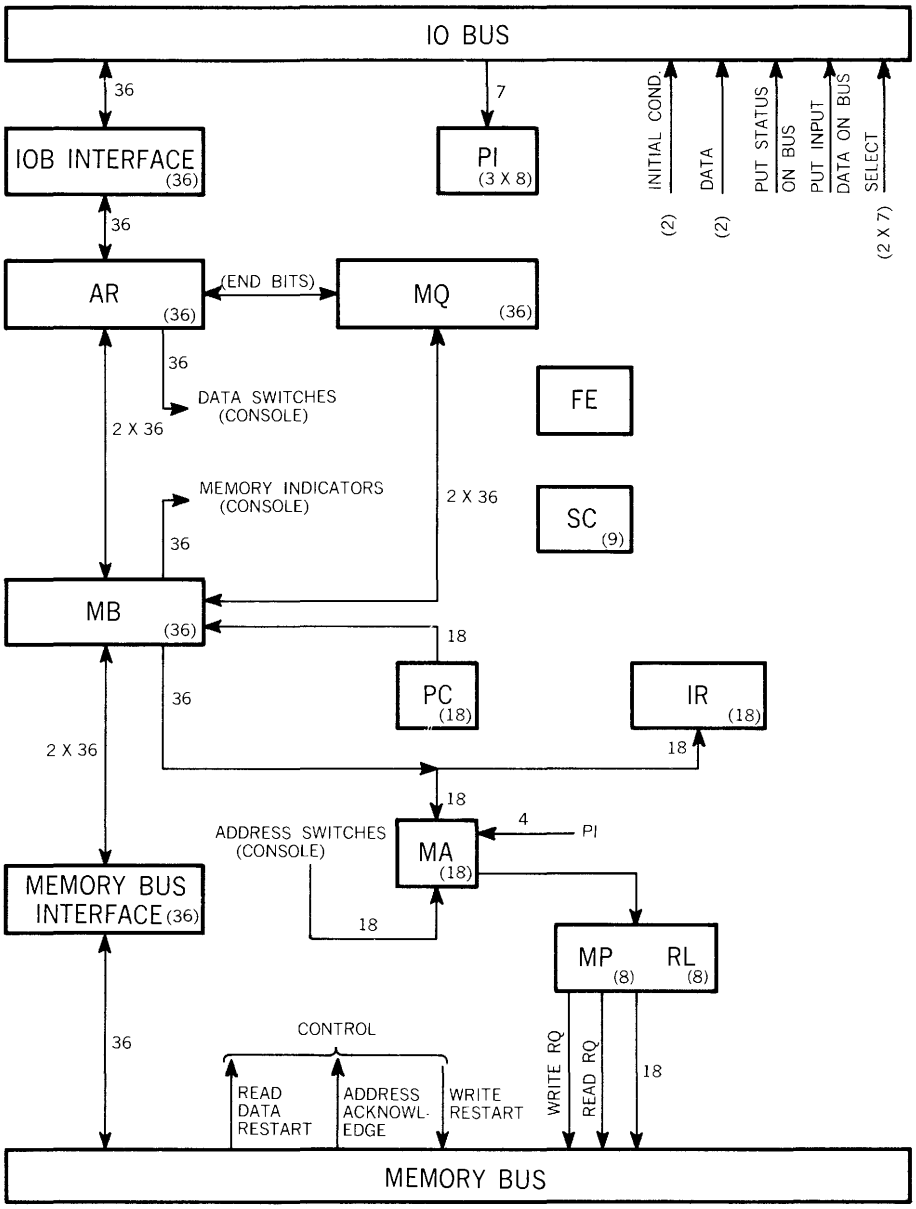


Figure 4 Data Flow Diagram

In addition to the registers shown in Figure 4, there are several flags important to programming.

AR CRY0	Arithmetic register Carry 0. This flag is set when there is a carry from bit 0 of the arithmetic register.
AR CRY1	Arithmetic register Carry 1. This flag is set when there is a carry from bit 1 of the arithmetic register.
AR OV	Arithmetic register Overflow flag. This flag is set when the sign of a result is not consistent with the signs of the operands.
PC CHANGE	Program Counter Change. This flag is set whenever control is transferred by a skip or jump instruction, except for a JFCL instruction with bit 12 set to 1.
PDL OV	Push Down List Overflow. A designated accumulator is incremented or decremented by 1000001_8 during the execution of the push down instructions. This flag is set if a carry from bit 0 occurs.
BIS	Byte Increment Suppression. This flag, when set to 1, suppresses incrementing during a byte instruction that calls for incrementing. This flag is necessary since a byte instruction may be interrupted between incrementing and the byte operation. In such a case, the incrementing must be suppressed during the first execution of the instruction following the interrupt. The flag is automatically set if such an interrupt occurs. It is cleared before the byte instruction is completed.
COUNT CLOCK ENABLE	This flag is set by a CONO instruction. If set, it enables the Clock flag to cause an interrupt. (See Processor I/O instructions.)
CLOCK FLAG	This flag is set periodically by a clock. If the Enable flag is set, and if a priority channel is assigned to the processor flags, it causes an interrupt.
PC CHANGE ENABLE	This flag is set by a CONO instruction. If set, and a priority channel has been assigned to the processor flags, it enables the PC Change flag to cause an interrupt.
OV FLAG ENABLE	This flag is set by a CONO instruction. If set, and if a priority channel has been assigned to the processor flags, it enables the AR OV flag to cause an interrupt.
EXEC MODE	This mode flip-flop is set when the executive program is running. In the executive mode, the program has access to all of memory. In the user mode (executive mode flip-flop is reset) the protection and relocation registers specify the area of core in which programs may be run. In the user mode illegal memory references and instructions which use the I/O equipment or attempt to halt the computer set the executive mode flip-flop.

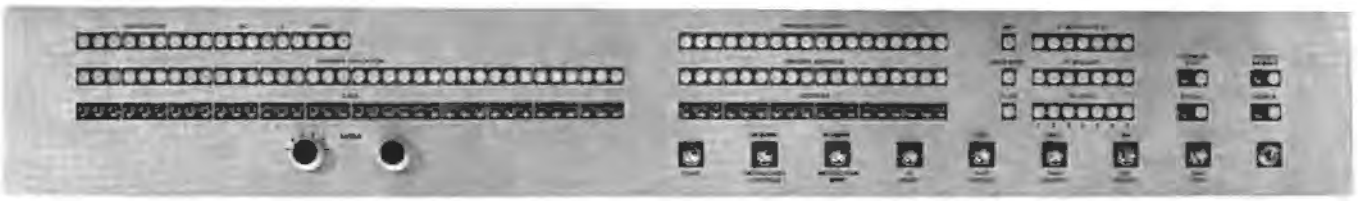


Figure 5 The Control Panel

CONSOLE CONTROLS AND INDICATORS

Lever Switches	Function
START	Depressing this switch resets the program counter to the address contained in the ADDRESS switches and starts the processor. Raising the switch does the same as the above except the machine is started in shadow mode. In the shadow mode, all references to memory locations $0–17_8$ go to the core memory rather than to fast memory. The shadow mode is terminated whenever an instruction located above 17_8 is executed. Therefore, if the address in the ADDRESS switches is greater than 17_8, the effect of the switch is the same whether it is raised or depressed. The switch has no effect if the machine is running.
STOP	Stops the processor at the end of the memory or instruction cycle. When the switch is in MEMORY STOP position, the read-pause-write mode of operation is inhibited; the machine stops at the end of both the read cycle and write cycle.
CONTINUE	If the processor has stopped at the end of an instruction or memory cycle, the program may be resumed by depressing this key. The combined use of the STOP and CONTINUE keys has the effect of a single step key.
I/O RESET	Clears the I/O System.
DEPOSIT	In the THIS position, the contents of the 36 DATA switches are deposited in the contents of the memory location specified by the ADDRESS switches. The machine will not be stopped if it is running. In the NEXT position, the memory address register is incremented by one, and the contents of the 36 DATA switches are deposited in the memory location specified by the contents of the memory address register. This key position has no effect if the machine is running.
EXAMINE	In the THIS position, the contents of the memory register addressed by the ADDRESS switches are displayed in the memory indicator lights. If the machine is running, it is not stopped. In the NEXT position, the memory address register is incremented by one, and the contents of the memory location specified by the memory address register are displayed in the memory indicator lights. This key position has no effect if the machine is running.
READER ON-OFF	Turns the power to the perforated tape reader on or off.
TAPE FEED	Feeds blank tape through the tape punch.

Toggle Switches	Function
CONSOLE DISABLE	This is a key-locked switch that disables all control switches that can interfere with the operation of the machine.

ADDRESS STOP	Stops the processor each time the contents of the memory address register are equal to the contents of the ADDRESS switches. An adjacent indicator light is on when the switch is on. The existence of an address stop condition inhibits the read-pause-write mode of memory operation, and the machine stops at the end of both the read cycle and write cycle.
REPEAT	If on while a lever switch is held down, the effect is as if the lever switch were being pressed repeatedly with a frequency determined by the setting of the speed knob and vernier.
SPEED	Two controls that vary the repeat interval when the REPEAT switch is on. The left knob is a six-position coarse control giving speed ranges from 3.4 microseconds and increasing by a factor of ten up to 340 milliseconds. The right knob is a continuously variable fine control.
POWER	Turns on power to the processor and all equipment connected to it.
MEMORY DISABLE	This switch inhibits the non-existent memory continue pulse. If a non-existent area of memory is addressed, the non-existent memory flag is set; and if the MEMORY DISABLE switch is on, the machine stops.
DATA	Up to 36 bits of information can be inserted for transfer to memory when the DEPOSIT switch is pressed. (May be examined under program control. See I/O Programming for Type 166 Processor.)
ADDRESS	There are 18 address switches which are used with various keys and switches.

Indicator Lights	Function
INSTRUCTION	Displays bits 0–8 of the instruction register.
AC	Displays bits 9–12 of the instruction register.
I	Displays bit 13 of the instruction register.
INDEX	Displays bits 14–17 of the instruction register.
MEMORY INDICATION	Displays the contents of the memory register addressed by the ADDRESS switches. The MI is set whenever the EXAMINE key is operated or the program references the register.
PROGRAM COUNTER	Displays the contents of the program counter. When the processor stops, the program counter will be pointing to the next instruction.
MEMORY ADDRESS	Display the last location referenced in memory before the machine stopped. If the machine stopped on a halt instruction (JRST 4), the MA contains the address of the halt instruction.
RUN	Lights when the machine is running.
MEM STOP	Lights and machine stops when end of memory cycle is reached if STOP switch is on or if an ADDRESS STOP occurs.
PI ON	Lights when priority interrupt system is enabled.

Indicator Lights	Function
PI ACTIVE	Display which PI channels are enabled.
PI REQUEST	Display which channels are requesting an interrupt.
PI IN PROGRESS	Display which channel is actually causing an interrupt.

PROGRAMMING FOR TYPE 166 PROCESSOR

For the Type 166 the first 16 locations of memory ($0–17_8$) function as accumulators, index registers, and ordinary memory locations. The state of the system resides entirely in memory except for the program counter and the AR flags. There are, therefore, no instructions which refer explicitly to the arithmetic elements of the processor and no need to protect partial results in the event of an I/O operation.

The function to be performed by one of these 16 locations is determined by the instruction word field in which the address appears. The processor always addresses memory in exactly the same way regardless of how it intends to use the data. Consequently, from the point of view of the memory system there is nothing unusual about the first 16 locations except that they may reside in a fast memory module, a desirable though not necessary option. Depending on the field in which the address appears, the processor uses the data received as an accumulator operand, as an index register for the effective address calculation, or as an ordinary memory operand. Thus, there is no need for separate groups of instructions to perform accumulator, index register, and memory manipulations. It should be noted that only the right halves of the 15 memory locations $1–17_8$ are used as index registers.

Some instructions use two accumulators: the accumulator specified by the accumulator field of the instruction word, and the one following. The accumulator following 17_8 is taken to be accumulator 0.

Some areas of memory are reserved for special functions and must be used with caution. They are:

$20_8–37_8$	Reserved for loader if fast memory is not available for the shadow mode.
$40_8–41_8$	Reserved for unused operation code traps.
$42_8–57_8$	Program interrupt system.

It is advisable that memory locations up to 100_8 be left unused to allow room for future expansion.

There are five major categories of Type 166 Processor instructions: data transmission, arithmetic and logical operations, executive instructions, I/O instructions, and push down instructions (a hybrid group consisting of both data transmission and control instructions).

The data transmission group may be divided into four subgroups: full word transmission; half word transmission (particularly useful for address and index manipulations); byte manipulation (particularly useful for storing and retrieving information in segments of less than one memory word); and two miscellaneous instructions, an exchange instruction and a block transmission instruction.

The arithmetic and logical operations may also be divided into four subgroups: fixed point arithmetic, floating point arithmetic, Boolean instruction, and shift instruction.

There are five subclasses of executive instructions: memory or accumulator modification and testing, a group of instructions that will increment or decrement an accumulator or memory word by one and/or test to determine a jump or skip; arithmetic compare, a group that will compare a number in an accumulator with a number in memory (fixed or floating point) and skip as specified; logical and modify compare, a group that will test and/or modify masked bits of an accumulator and skip as specified; jump instructions; and some miscellaneous instructions.

The push down and I/O groups are both very small groups with four and eight instructions, respectively.

MODES — Some instruction classes consist of basic instructions and modes. Groups of instructions which move and/or combine information require a transmission mode. Transmission modes specify the direction in which the information is to be moved, e.g., memory to accumulator or accumulator to memory; they may specify where to leave the results of an operation, e.g., in memory, in an accumulator, or both; or they may specify what parts of the result from an operation are to be stored, e.g., low order bits and/or remainders in floating point operations. Finally, there is an immediate mode which directs the instruction to take its effective address as the memory operand.

Groups of instructions which compare and skip or jump require modes to specify the conditions under which the skip or jump is to occur.

UNUSED OPERATION CODES — There are 26 unused operation codes reserved for future expansion. They have no effect on the machine and may be thought of as no-operation instructions.

NO OPERATION INSTRUCTIONS — Some combinations of instructions and their modes have no effect on the state of the machine. These may be considered as no-operation instructions.

PROGRAMMED OPERATORS — The 64 instruction codes with zeros in the first three bits have been reserved for programmed operators. When used, bits 0–17 of the instruction are stored in bits 0–17 of location 40. The result of the effective address calculation is stored in bits 18–35 of location 40. The instruction in location 41 is then executed. Presumably, this instruction would be a jump to a routine that would interpret the programmed operator. It should be a jump which stores the program counter in order to return to the main program.

Instruction Word Format

There are two formats for the Type 166 Processor instructions, the basic instruction format and the I/O instruction format. The basic format is shown in Figure 6.

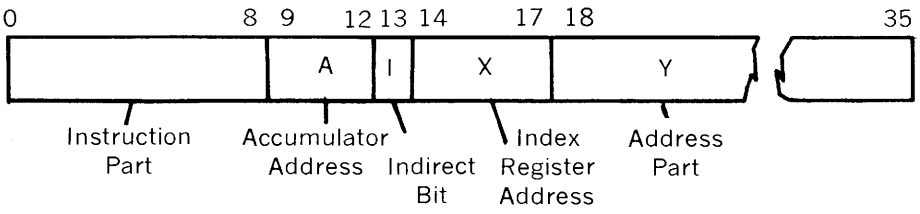


Figure 6 — Basic Instruction Format

All instructions, with the single exception of the I/O, use this basic format. The functions of I, X, and Y, which are common to the basic and I/O formats, are described below under Effective Address Calculation. Bits 0-8 specify the instruction code, including the mode. Bits 9-12 specify the accumulator to be used by the instruction. The format for the I/O instruction is shown in Figure 7.

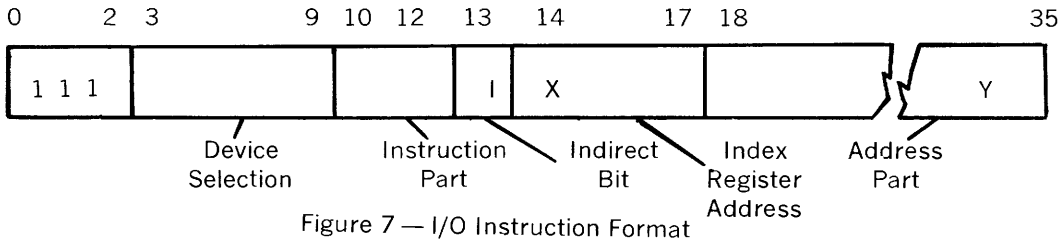


Figure 7 — I/O Instruction Format

Bits 3-9 are used to specify any one of the 128 possible I/O devices, where the processor itself and the program-interrupt system are considered to be the first of the devices (device #000 000 0 and #000 000 1 respectively for use with the conditions-out and conditions-in instructions). Bits 10-12 are used to specify any one of the eight possible I/O instructions.

EFFECTIVE ADDRESS CALCULATION

All instructions in the PDP-6, without exception, calculate an effective address using bits 13-35 in exactly the same way. The steps are:

1. Obtain the number in the address field Y, bits 18–35. Any one of 262,144 locations can be specified.
2. If the index field X, bits 14–17, is non-zero, then add the contents of the specified index register to the number obtained in step 1.
3. Obtain the indirect bit, I, bit 13. If it is 0, the calculation is done and the result of steps 1 and 2 is the effective address. If it is 1, then go to step 4.
4. Use the address calculated by steps 1 and 2 to obtain a new word from memory, and go back to step 1.

The effective address calculation continues until a word is encountered with a 0 in bit 13. At that point the result of steps 1 and 2 is taken as the effective address for the instruction.

This calculation is carried out for all instructions. When the immediate mode is specified, if one or more indirect addresses are involved, the effective address is not the number appearing in the address field of the word. Rather, it is the first result of the effective address calculation as it is for all instructions.

Timing

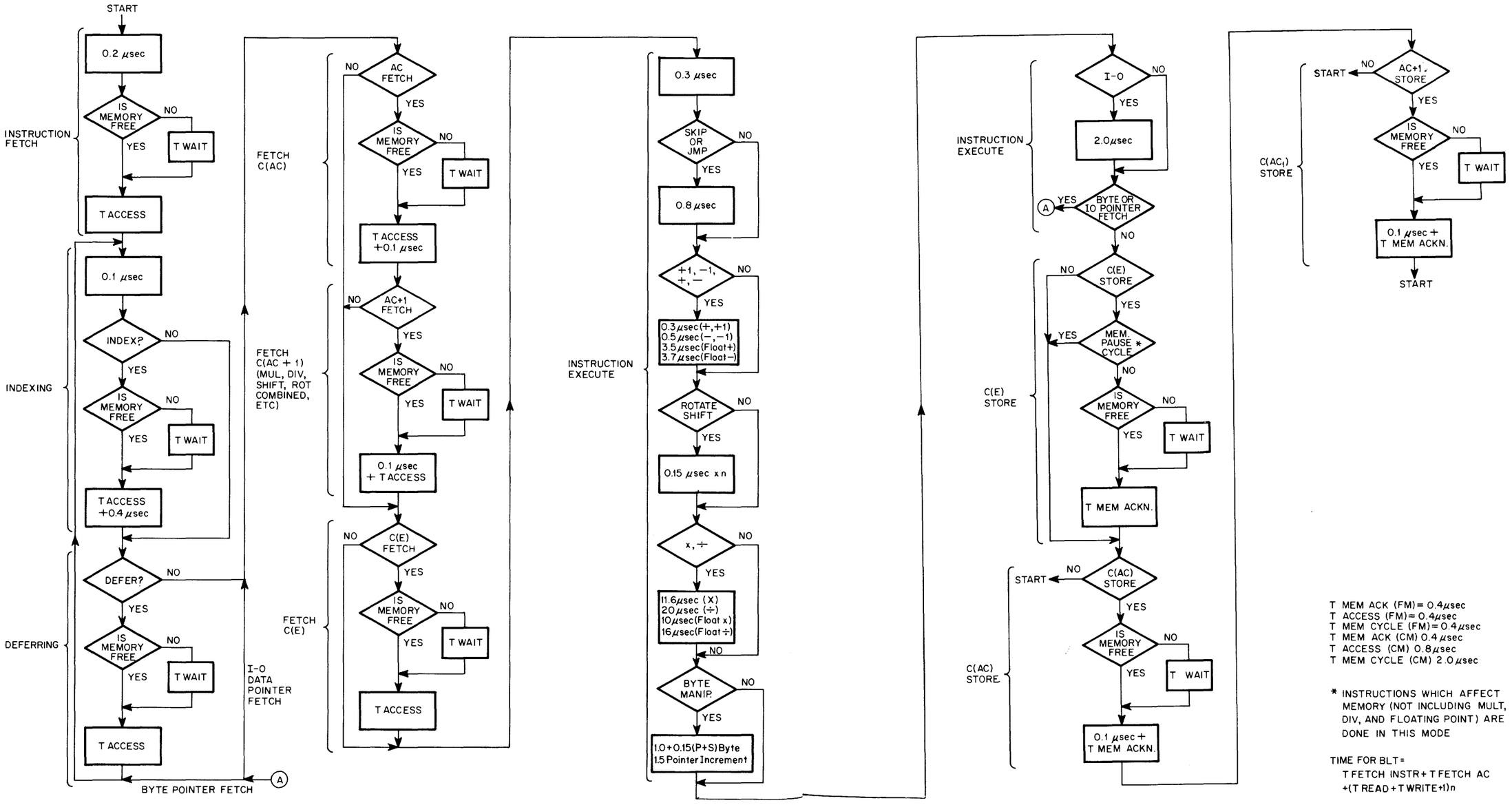
PDP-6 processors are not synchronized by a clock. Therefore, PDP-6 systems run asynchronously with as high a duty factor as possible for each unit. Instruction times depend on several variables, such as the speed of the memory module in which the desired information is stored, whether the module is busy when the information is requested, and where the results are to be stored. Speed of execution also depends on the number of iterations involved in the effective address calculation, particularly since a new memory reference is required each time an indirect hit is encountered.

Table 2 summarizes instruction times for each group of instructions. Two times are given, a fast time and a slow time. The fast times are based on starting with instruction and data in fast memory. The slow times are based on starting with both instruction and data in the same core memory and allow for one index reference. The fast times are not necessarily minimum, since instructions in the immediate mode may run faster. Nor are the slow times maximum times, since an instruction may take considerably longer if there are several levels of indirect addressing. Exact times depend on the program context in which the instructions occur and on other factors. Therefore the figures should not be used to calculate program running time.

TABLE 2 SUMMARY OF INSTRUCTION TIMES

Instructions	Fast	Slow
Full and half word moves	1.9	4.0
Full and half word immediate	1.5	2.7
Byte manipulation	5.7	8.0
Byte manipulation and increment	7.2	8.0
Block transfer	1.5+0.8n	2.4+1.2n
Exchange	2.8	4.0
Fixed point add	2.7	4.3
Fixed point subtract	2.9	4.5
Fixed point multiply	14.5	16.1
Fixed point divide	23.4	25.0
Floating point add	5.8	8.0
Floating point subtract	6.0	8.2
Floating point multiply	12.4	14.5
Floating point divide	18.4	20.5
Boolean	2.7	4.3
Shifting (18 bits)	4.7	5.9
Memory, AC modification and testing	2.6	3.9
Arithmetic compare	2.7	4.4
Logical compare	2.7	4.4
Jumping	1.8	3.0
I/O		
basic	3.0	6.2
augmented	3.8	7.0
Push down	3.1	6.4

The flow diagram, Figure 8, can be used to calculate the approximate execution time of any instruction under any set of conditions.



T MEM ACK (FM) = 0.4 μsec
T ACCESS (FM) = 0.4 μsec
T MEM CYCLE (FM) = 0.4 μsec
T MEM ACK (CM) 0.4 μsec
T ACCESS (CM) 0.8 μsec
T MEM CYCLE (CM) 2.0 μsec

* INSTRUCTIONS WHICH AFFECT MEMORY (NOT INCLUDING MULT, DIV, AND FLOATING POINT) ARE DONE IN THIS MODE

TIME FOR BLT =
T FETCH INSTR + T FETCH AC
+ (T READ + T WRITE + 1) n

Figure 8 Instruction Time Flow Diagram

INSTRUCTION CLASSES

Data Transmission

For all transmission instructions, the word whose contents are being moved is referred to as the source word. The word to which the data is being moved is referred to as the destination word.

FULL WORD TRANSMISSION

MOVE	Move full word.
MOVS	Move full word with left and right halves exchanged.
MOVN	Move negative (twos complement) of full word.
MOVM	Move magnitude of full word.

MODES

The source words are unaffected except in the S mode where the source word and the destination word are identical.

	Move contents of effective address to accumulator.
I	Move effective address to accumulator.
M	Move contents of accumulator to effective address.
S	Move contents of effective address, swapped, negated, or set to magnitude, to effective address.

HALF WORD TRANSMISSION

There are 16 half word move operations derived from all possible combinations of left to left, right to right, left to right, and right to left, with four possible operations on the other half of the destination word. The operations are: do nothing to the other half; set the other half to all zeros; set the other half to all ones; and set all bits of the other half equal to the high order bit of the half being moved.

HLL	Move left half to left half, no effect on right half of destination word.
HRR	Move right half to right half, no effect on left half of destination word.
HRL	Move right half to left half, no effect on right half of destination word.
HLR	Move left half to right half, no effect on left half of destination word.
HLLZ	Move left half to left half, set right half of destination word to all zeros.
HRRZ	Move right half to right half, set left half of destination word to all zeros.
HRLZ	Move right half to left half, set right half of destination word to all zeros.
HLRZ	Move left half to right half, set left half of destination word to all zeros.

HLLO	Move left half to left half, set right half of destination word to all ones.
HRRO	Move right half to right half, set left half of destination word to all ones.
HRLO	Move right half to left half, set right half of destination word to all ones.
HLRO	Move left half to right half, set left half of destination word to all ones.
HLLE	Move left half to left half, set all bits of the right half to the sign (bit 0) of the left half.
HRRE	Move right half to right half, set all bits of the left half to the sign (bit 18) of the right half.
HRLE	Move right half to left half, set all bits of the right half to the sign (bit 0) of the left half.
HLRE	Move left half to right half, set all bits of the left half to the sign (bit 18) of the right half.

MODES

The source words are unaffected except in the S mode where the source word and the destination word are identical.

	Move contents of effective address to accumulator.
I	Move effective address to accumulator. (If source is left half, a 0 is transferred.)
M	Move contents of accumulator to effective address.
S	Move contents of effective address to effective address.

BYTE MANIPULATION

Byte manipulation instructions permit easy access to any number of contiguous bits located anywhere in a single, 36-bit memory word. To specify the size and location of the byte, a pointer word is located by the effective address of the byte manipulation instructions. The I, X, and Y fields of the pointer word (see Figure 9) are used in the usual manner to compute an effective address which will be the location of the memory word containing the byte. The P field specifies the number of bits between the right end of the word and the farthest right bit of the byte. The S field specifies the size of the byte, up to 36 bits. Therefore, the byte is located in bits 36-P-S through 35-P.

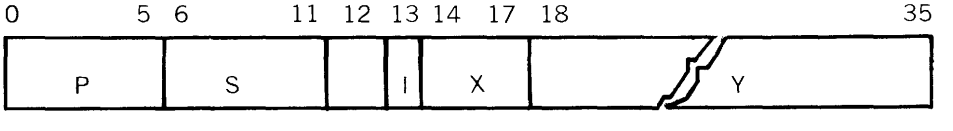


Figure 9 — Byte Pointer Format

To store successive bytes in memory and retrieve them with a minimum of instructions, the processor must be able to increment the pointer word. To increment, the size of the byte is subtracted from P, moving the position to the right by one byte. If there is insufficient room in the memory word for the next byte (P-S $<$ 0), then the Y field of the pointer word is incremented by one and P is reset to 36-S.

LDB — Load bits 35 to 36-S of the accumulator with the specified byte. Bits 0 through 35-S of the accumulator are cleared.

DPB — Deposit byte, starting in bit 36-S through bit 35 of the accumulator, in the portion of the memory word as specified by the pointer word.

IBP — Increment byte pointer by one byte position.

ILDB — Same as LDB, but increments pointer word before loading.

IDPB — Same as DPB, but increments pointer word before depositing.

See the BIS flag for its effect on the increment instruction.

MISCELLANEOUS

EXCH — The contents of the effective address and the contents of the specified accumulator are exchanged.

BLT — Moves a block of storage (source block), beginning at the location specified by the left half of the designated accumulator, to a block (destination block) beginning at the location specified by the right half of the accumulator. The size of the block is determined by the effective address; that is, the effective address is equal to the last location of the destination block. The source block is unaffected unless it overlaps the destination block. The accumulator is only referenced at the beginning of the instruction; therefore, it is not changed unless a program interrupt occurs. If a program interrupt occurs, the left half of the accumulator (source block) is set to the address of the next word to be moved, and the right half is updated to the next word in the destination block. When the break is dismissed, the BLT instruction then resumes where it left off.

Arithmetic and Logical

Two types of arithmetic are used by the Type 166 Processor: fixed point and floating point. Integer arithmetic, such as address and index arithmetic, is only a special case of fixed point arithmetic. For consistency and to make possible the use of fixed point arithmetic compare commands with floating point number, all arithmetic in the 166 is done in twos complement binary. That is, carries from the highest order bit are dropped, and negative numbers are the twos complement of the positive numbers with the same magnitude.

To form the negative (twos complement) of a number, change all zeros to ones and all ones to zeros, then add one to the result. In this scheme, there are two points to be carefully noted. Bit zero will always function as a sign bit; if zero, the number is positive, if one, the number is negative. (However, this must not lead to identification with sign-magnitude conventions.) Secondly, forming the negative by twos complement applies to floating point numbers as well as fixed point numbers. Therefore, the exponent part of floating point numbers, as well as the fractional part, is complemented.

Fixed point numbers are represented in binary form so that the value of a number in memory is:

$$\sum b_i 2^{(35+n-i)}$$

where b_i designates the contents of bit i, and n depends on the position of the binary point. However, since the position of the binary point is of no consequence to the machine, two vastly different numbers may have the same representation in memory. It is left to the programmer to establish consistent point conventions for any sequence of operations

Floating point numbers use 1 bit for the sign, 8 bits for the exponent, and 27 bits for the fraction. The binary point is therefore to the left of bit 9.

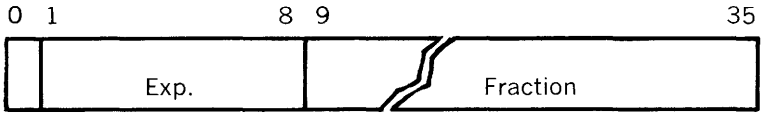


Figure 10– Floating Point Format

For positive numbers, the exponent is excess 200_8. Since in Type 166 negative numbers are always the twos complement of the positive number with the same magnitude, exponents for negative numbers are the ones complement of the excess 200_8 exponent. Floating point zeros are represented by all zeros. Floating point numbers are considered to be normalized if the sign bit does not equal bit 9, or if bits 9–35 contain 400000000_8.

FIXED POINT ARITHMETIC

Fixed point arithmetic is single precision. However, to facilitate programming double precision, a carry from bits 0 and 1 will set the CRY0 and CRY1 flags, respectively. Also, the fixed point instructions will set the OV flag if the sign of the answer is not correct with respect to the signs of the operands. For additional conditions under which the OV flag will be set, see the IMUL, DIV, and IDIV instructions. Note that the CRY0 and OV flags are not set under equivalent conditions.

ADD — The memory operand is added to the contents of the accumulator. This instruction may set the CRY0, CRY1, and OV flags.

SUB — The memory operand is subtracted from the contents of the accumulator. The subtraction is carried out by adding the twos complement of the contents of the accumulator to the memory operand, followed by forming the twos complement of the result. This instruction may set the CRY0 ,CRY1, and OV flags.

MUL — Multiply the contents of the accumulator by the memory operand. The contents of the designated AC are placed in the MB. The contents of the effective address are placed in the MQ. The AR is cleared.

For multiplication the MQ has an extra low order bit, bit 36, which is cleared before multiplication begins. Bits shifted out of MQ_{35} enter MQ_{36}.

The multiplication algorithm is:

1. If MQ_{35} and MQ_{36} are equal, then shift AR/MQ right one place.
2. If MQ_{35} is a 1, and MQ_{36} is 0, then subtract C(MB) from the AR. Shift AR/MQ right one place.
3. If MQ_{35} is a 0 and MQ_{36} is a 1, then add C(MB) to the AR and shift AR/MQ right one place.

AR/MQ is treated as one 72-bit word. Bits shifted out of AR_{35} enter MQ_0. The algorithm is repeated until the bit originally in MQ_0 reaches MQ_{35}. The algorithm is then repeated once more with shift MQ right one place replacing shift AR/MQ. That is, the final shift is for the MQ only with AR_0 being copied into MQ_0 so the signs of the AR and the MQ agree.

IMUL — Is the same as MUL, except that the low order part of the product is stored as specified by the mode control. The high order part is lost. This instruction will set the OV flag on the further condition that there are significant bits in the high order part of the product.

DIV — Divide the dividend by the contents of the memory operand. The dividend is a 72-bit word with the high order part set equal to the contents of the designated accumulator n, and the low order part set equal to the contents of accumulator n + 1. If one or both of the operands is negative, the effect of this instruction is as if division were performed on the magnitudes of the operands. The answer is then formed by twos complementing where necessary. The remainder has the sign of the dividend, and the sign of the quotient is the exclusive OR of the sign of the dividend and the sign of the divisor. The quotient is stored as specified by the mode control. The remainder, except in the M mode, goes to accumulator n + 1. The instruction sets the OV flag if the high order part of the dividend is greater than or equal to the divisor.

IDIV — Same as DIV, except that the low order part of the dividend is set to the contents of the designated accumulator. The high order part of the dividend is set to the sign bit of the low order part. OV is possible only if the divisor is zero.

MODES

Combine contents of the effective address with the contents of the accumulator; results to accumulator.

I — Combine the effective address with the contents of the accumulator; results to accumulator.

M — Combine contents of effective address with the contents of the accumulator; results to the effective address.

B — Combine contents of the accumulator with the contents of the effective address; results to both accumulator and effective address. If the effective address equals the accumulator number plus one, the effective address is left with either the low order part of the result or with the remainder, since the accumulators are stored after the memory.

FLOATING POINT ARITHMETIC

The floating point instructions may set the OV flag if exponent overflow or underflow occurs. No attempt is made to distinguish between them.

CAUTION

The floating point instructions may not give the correct results for unnormalized operands.

FAD — Floating add the memory operand to the contents of the accumulator. The number with the smaller exponent is shifted right by the difference in the magnitude of the exponents. The fractional parts are then added as fixed point numbers. The result is normalized, and the answer given in terms of the higher exponent. This instruction may set the OV flag.

FSB — Floating subtract the memory operand from the contents of the accumulator. The subtraction is done by adding, as above, the twos complement of the contents of the effective address. May set OV flag.

FMP — Floating multiply the contents of the accumulator by the memory operand. The multiplication is done by adding the exponents and multiplying the fractional parts as fixed point numbers. May set the OV flag.

FDV — Floating divide the contents of the accumulator by the memory operand. The division is done by subtracting the exponent parts and dividing the fractional parts as fixed point numbers. May set the OV flag.

There are four instructions which include rounding. The rounding is done as follows: If bit 1 in the low order part of the result is a 1, and the sign of the high order part is positive, then add one to the lowest order bit in the high order part. Otherwise, do nothing.

FADR — Same as floating add, followed by rounding.

FSBR — Same as floating subtract, followed by rounding.

FMPR — Same as floating multiply, followed by rounding.

FDVR — Same as floating divide, followed by rounding.

MODES

Combine contents of effective address with contents of accumulator; results to accumulator.

L — Combine contents of effective address with contents of accumulator; results to accumulator, remainder or low order bits to AC + 1, left adjusted with no exponent part.

M — Combine contents of effective address with contents of accumulator; results to memory.

B — Combine contents of effective address with contents of accumulator; results to accumulator and to effective address.

There is one floating point instruction which has no mode control:

FSC — (Floating Scale) Multiply contents of accumulator by 2^E, where E is the effective address.

BOOLEAN

The Type 166 Processor can perform all 16 Boolean operations of two variables. The contents of the designated accumulator and the memory operand are matched on a bit for bit basis. The result is a 0 or 1 in the corresponding bit of the resultant word. The Boolean instructions are listed below. The result of each instruction is shown at the right under the four possible memory-accumulator configurations.

Mnemonic	Meaning	Memory Bit / Accumulator Bit	0 0	0 1	1 0	1 1
SETZ	Set to all zeros		0	0	0	0
AND	And		0	0	0	1
ANDCA	AND with the contents of the accumulator complemented		0	0	1	0
SETM	Set the result equal to the contents of the effective address		0	0	1	1
ANDCM	AND with the contents of the effective address complemented		0	1	0	0
SETA	Set the result equal to the contents of the accumulator		0	1	0	1
OR	Exclusive OR		0	1	1	0
IOR	Inclusive OR		0	1	1	1
ANDCB	AND with the contents of both the accumulator and the effective address complemented		1	0	0	0
EQV	Equivalent		1	0	0	1
SETCA	Set the resultant to the complement of the contents of the accumulator		1	0	1	0
ORCA	Inclusive OR with the contents of the accumulator complemented		1	0	1	1
SETCM	Set the resultant equal to the complement of the contents of the effective address		1	1	0	0
ORCM	Inclusive OR with the contents of the effective address complemented		1	1	0	1
ORCB	Inclusive OR with the contents of both the accumulator and the effective address complemented		1	1	1	0
SETO	Set resultant to all ones		1	1	1	1

MODES

	Combine contents of effective address with contents of accumulator; results to accumulator.
I	Combine effective address with contents of accumulator; results to accumulator.
M	Combine contents of effective address with contents of accumulator; results to effective address.
B	Combine contents of effective address with contents of accumulator; results to both accumulator and effective address.

SHIFTING

There are three kinds of shifting:

1) arithmetic shift — performs twos complement multiplication by powers of 2. The sign is unchanged. When going to the right, the sign is shifted into bit 1. Ones or zeros leaving bit 35 are lost. The Overflow flag is set if, during shifting, the exclusive OR of $C(AR)_0$ and $C(AR)_1$ becomes a 1.
2) rotate — bits leaving one end enter at the other end.
3) logical shift — bits leaving one end are lost, and zeros enter the other end.

The direction and number of places to be shifted are determined by bits 18 and 28–35 of the effective address, which together form a 9-bit twos complement number. If this number is positive, the shift is to the left; if negative, the shift is to the right. The number of bits shifted is equal to the magnitude of the number.

Shifts may be of the contents of an accumulator A or of a combined word, consisting of the contents of the accumulator A and the contents of the accumulator A + 1, in that order. If a combined shift instruction, and the specified accumulator is 17_8, then location 20_8 will be rotated unless locations 20_8–37_8 are read-only memory. Consequently, it is best to avoid this combination.

ASH	Arithmetic shift
ROT	Rotate
LSH	Logical shift
ASHC	Same as ASH, except that the contents of accumulator n and accumulator n + 1 are treated as one 72-bit word with accumulator n on the left or higher order position. When shifting either left or right, the sign bit of accumulator n + 1 is set equal to the sign of accumulator n. When shifting right, bits moving through bit 35 of accumulator n enter bit 1 of accumulator n + 1. When shifting left, ones or zeros leaving bit 1 of accumulator n + 1 enter bit 35 of accumulator n. Bits leaving bit 1 of accumulator n are lost. Zeros enter bit 35 of accumulator n + 1 from the right.
ROTC	Rotate combined accumulator
LSHC	Logical shift of combined accumulator

Executive

MEMORY AND ACCUMULATOR MODIFICATION AND TESTING

There are two sets of instructions from this group: accumulator jumps and memory skips. The jump instructions test and/or modify contents of the accumulator. If the conditions specified by the mode control are met, program control jumps to the effective address. Skip instructions test and/or modify the contents of the effective address. If the conditions specified by the mode control are met, the next instruction is skipped.

Memory Skips	Accumulator Jumps	
SKIP	JUMP	No modification, test only
AOS	AOJ	Add one and test
SOS	SOJ	Subtract one and test

MODES

Skip or jump if the contents of the word tested are:

—	Never skip
L	Less than zero
E	Equal to zero
LE	Less than or equal to zero
A	Always skip
GE	Greater than or equal to zero
N	Not equal to zero
G	Greater than zero

ARITHMETIC COMPARE

CAM	Compares algebraically the contents of the effective address with the contents of the accumulator. If the condition specified by the mode is met, the next instruction is skipped.
CAI	Same as CAM, except that it compares the effective address with the contents of the accumulator.

MODES

—	Never skip
L	Skip if contents of accumulator are less than contents of data word.
E	Skip if contents of accumulator are equal to contents of data word.
LE	Skip if contents of accumulator are less than or equal to contents of data word.
A	Always skip.
GE	Skip if contents of accumulator are greater than or equal to contents of data word.
N	Skip if contents of accumulator are not equal to contents of data word.
G	Skip if contents of accumulator are greater than contents of data word.

LOGICAL COMPARE AND MODIFY

Bits of an accumulator word that are masked by bits of a memory word, that is, those bit positions of the accumulator corresponding to bit positions in the memory word containing ones, can be modified and/or tested to determine a skip. The masked bits may be left unchanged, set to zeros, set to ones, or complemented. The masking may be done by the contents of the memory word, specified by the effective address taken directly; by the contents of the memory word with its left and right halves swapped; or by the effective address itself. Since the effective address is only 18 bits, it must be specified whether it is to mask the left or right half of the accumulator. The unused half of the accumulator is masked by zeros.

The conditions under which a skip will occur are specified by the mode control. The test is made before any modification occurs. Instructions set the PC Change flag if a skip is executed.

TDN	Test, masking by direct contents of memory word, no modification.
TSN	Test, masking by swapped contents of memory word, no modification.
TLN	Test, masking left half by effective address, no modification.
TRN	Test, masking right half by effective address, no modification.
TDZ	Test, masking by direct contents of memory word, set masked bits to zero.
TSZ	Test, masking by swapped contents of memory word, set masked bits to zero.
TLZ	Test, masking left half by effective address, set masked bits to zero.
TRZ	Test, masking right half by effective address, set masked bits to zero.
TDO	Test, masking by direct contents of memory word, set masked bits to ones.
TSO	Test, masking by swapped contents of memory word, set masked bits to ones.
TLO	Test, masking left half by effective address, set masked bits to ones.
TRO	Test, masking right half by effective address, set masked bits to ones.
TDC	Test, masking by direct contents of memory word, complement masked bits.
TSC	Test, masking by swapped contents of memory word, complement masked bits.
TLC	Test, masking left half by effective address, complement masked bits.
TRC	Test, masking right half by effective address, complement masked bits.

MODES

Skip if:	
—	Never skip
E	All masked bits are zero
A	Always skip
N	Not all masked bits are equal to zero

JUMP INSTRUCTIONS

JSR — Jump to subroutine. The AR OV, AR CRY0, AR CRY1, PC Change, BIS, and User Mode flags are all stored in the contents of the effective address, bits 0 through 5 respectively, and the BIS flag is cleared. The program counter, which has been incremented to point to the next instruction in sequence, is stored in C(E), bits 18–35. Program control then jumps to the effective address plus one. (See JRST for the return instruction.)

JSP — Jump and save program counter. The contents of the program counter, which will point to the next instruction, are stored in the designated accumulator. Program control jumps to the effective address. (The return from this jump can be a single jump instruction using the accumulator as an index register.)

JSA — Jump and save accumulator. The accumulator is stored in the effective address. The PC, which points to the next instruction is stored in the right half of the AC, and the effective address is stored in the left half of the accumulator. Program control jumps to the effective address plus one.

JRA — Jump and restore accumulator. This is the return instruction to match JSA. The contents of the accumulator are replaced by the word addressed in its left half. Control is transferred to the effective address.

There are three reasons for the JSA-JRA pair: to provide for subroutines with multiple entries; to provide an easily accessible reference for getting data; and to prevent loss of information, making it possible to nest subroutines.

AOBJP — Increment both halves of accumulator by one, and jump if the result is positive. (See the Miscellaneous Instructions.)

AOBJN — Increment both halves of accumulator by one, and jump if the result is negative. (See the Miscellaneous Instructions.)

JRST — Jump and restore. (See the Miscellaneous Instructions.) Bits 9–12 (accumulator field) determine the effect of this instruction as follows:

If bit 9 = 1, (AC8), reset the current priority interrupt channel.

If bit 10 = 1, (AC4), halt.

If bit 11 = 1, (AC2), restore flags (see note below).

If bit 12 = 1, (AC1), set User Mode flag to a 1 (enter user mode).

This instruction, with bit 11 equal to 1, may be used as a return instruction to match the JSR. In this case the indirect bit, bit 13, must also be set to 1. Otherwise, the instruction is undefined.

The AR OV, AR CRY0, AR CRY1, PC Change, and BIS flags are set to the C(E), bits 0–4 respectively. If C(E), bit 5, is a 1, the User Mode flag is set; otherwise, the User Mode flag is unchanged.

JFCL — If designated flags are set, jump and clear flags. (See the Miscellaneous Instructions.) The flags are designated as follows:

Bit 9, (AC8), designates the OV flag.

Bit 10, (AC4), designates the CRY0 flag.

Bit 11, (AC2), designates the CRY1 flag.

Bit 12, (AC1), designates the PC Change flag. This flag remains cleared.

The PC Change flag is set whenever the PC is incremented by more than one to enable program flow tracing.

MISCELLANEOUS

XCT — Execute instruction at effective address. The instruction at the effective address may be another XCT instruction. Thus, chains of XCT instructions are permitted. (See the Miscellaneous Instructions.)

Push Down Instructions

In the following group of instructions, note that if a carry from bit 0 occurs when incrementing (decrementing) the accumulator, the PDL OV flag is set.

PUSH — The contents of the effective address are moved to the next location on the push down list, that is, the location to which the address part of the accumulator is pointing + 1. The accumulator is then incremented by 1000001_8.

PUSHJ — The accumulator is incremented by 1000001_8. The contents of the program counter, which is pointing to the next instruction, are then moved to the next free location on the push down list, that is, the location to which the address part of the accumulator is pointing. Control then jumps to the effective address. This instruction sets the PC Change flag.

POP — The last word placed on the push down list, that is, the contents on the location to which the address part of the accumulator is pointing, is moved to the effective address. The accumulator is then decremented by 1000001_8.

POPJ — The last word placed on push down list, that is, the contents of the location to which the address part of the accumulator is pointing, is moved to the program counter. The accumulator is then decremented by 1000001_8. This instruction sets the PC Change flag.

Input-Output Instructions

CONO — The effective address is transferred to the device control register. For the significance of the bit configuration see the specific device write-up.

CONI — The contents of the device control register are transferred to the contents of the effective address.

DATAO The contents of the effective address are placed on the I/O bus for output by the device. This instruction will also cause certain control actions in the device; see the particular device for details.

DATAI Data read in by the device and available in the device buffer register is transferred to the effective address. This instruction will also cause certain control actions in the device; see the particular device for details.

CONSZ This instruction is equivalent to a CONI followed by a test and skip. If the logical AND of the contents of the device control register and the contents of the effective address equals zero, then skip. Otherwise, execute the instruction following the CONSZ.

CONSO Equivalent to a CONI followed by a test and skip. If the logical AND of the contents of the device control register and the contents of effective address is not equal to zero, then skip. Otherwise, execute the instruction following the CONSO.

BLKI This instruction is used to read in a block of data. It uses the effective address to locate a pointer word, then increments both halves of the pointer word by one. The right half of the pointer word is used to determine the address of the memory location which is to receive the input data.

The instruction reads in the data word from the appropriate I/O device in exactly the same way as a DATAI instruction. The left half of the pointer word is then tested for a zero (end of block condition). If zero, control goes to the next instruction in sequence. If not zero, then

a) if in priority interrupt, dismiss channel.

b) if not in priority interrupt, skip the next instruction.

BLKO This instruction is exactly like the BLKI, except that it is an output instruction.

INPUT-OUTPUT PROGRAMMING FOR TYPE 166 PROCESSOR

This section is concerned with the use of the Type 166 Processor as an I/O processor. For details of the I/O system see Chapter 4. Here we will be concerned only with the use of the 166 I/O commands along with the use and control of the 166 Priority Interrupt System.

There are four basic 166 instructions for controlling the I/O devices; that is, for generating the six command pulses described in Chapter 4. They are:

CONO Generates the conditions out sequence.

CONI Generates the conditions in sequence.

DATAO Generates the data out sequence.

DATAI Generates the data in sequence.

Of the remaining four commands two are equivalent to CONI followed by an instruction to the processor to test and skip. They are CONSZ and CONSO. The remaining two, BLKO and BLKI, are instructions to the processor to control the repeated execution of DATAO and DATAI commands to the I/O system.

The use of these instructions will be illustrated by programming examples for a generalized device. For their use with actual devices, a device number must appear in bits 3–9 of the instruction word. (See the I/O Instruction Format, Figure 7.) This number may be generated by a device mnemonic.

Further, the significance of the bits in the effective address used with a CONO instruction and the significance of the bits read into the contents of the effective address by a CONI instruction must be known for each device. Finally, due consideration must be given to the number of bits which can be transmitted to the data register since information to be transmitted from memory to the device must appear in the n low order bits of the memory word, where n is the length of the data register. Similarly, due consideration must be given to the number of bits which the device will transmit to memory through the processor. All of this required information may be found by consulting the individual device write-up in Chapter 4.

Up to 126 I/O devices may be attached to the 166 through an I/O bus. The reason for the restriction to 126 devices, rather than 128, is that the processor and the processor's priority interrupt system are considered to be two I/O devices as follows:

Processor

Device Number 000 000 0
Mnemonic APR

The following I/O instructions may be used to refer to the processor:

DATAO Used to set the memory protection and relocation registers. The contents of the effective address, bits 18–25, go to the relocation register; bits 27–34 go to the protection register.

DATAI Reads the 36 data switches.

CONO Bits 18–35 of the effective address have been assigned the following significance:

- 18 If a 1, reset the PDL OV flag.
- 19 I/O reset.
- 22 If a 1, reset the Memory Protection flag.
- 23 If a 1, reset the nonexistent Memory flag
- 24 If a 1, turn the Clock Count Enable flag off.
- 25 If a 1, turn the Clock Count Enable flag on.
- 26 If a 1, turn the Clock flag off.
- 27 If a 1, turn the PC Change Enable flag off.
- 28 If a 1, turn the PC Change Enable flag on.

29 If a 1, turn the PC Change flag off.

30 If a 1, turn the OV flag enable off.

31 If a 1, turn the OV flag enable on.

32 If a 1, turn the OV flag off.

33–35 Assign a priority channel to the above processor flags.

If a priority channel is assigned to the processor flags, an interrupt will occur on that channel if any one, or more, of the following conditions hold:

1. If the Illegal Instruction flag is a 1.
2. If the non-existent Memory flag is a 1.
3. If the Clock Count Enable flag is a 1, and the Clock Count flag is a 1.
4. If the PC Change flag enable is a 1, and the PC Change flag is a 1.
5. If the OV flag enable is a 1 and the OV flag is a 1.

The real time clock runs at the power line frequency. It sets the Clock flag every 1/60 of a second for 60-cycle current whether or not the Clock Count Enable flag is on.

CONI Bits 18–35 of the location specified by the effective address are set as follows:

18 To a 1 if the PDL OV flag is set.

22 To a 1 if the Illegal Instruction flag is set.

23 To a 1 if the non-existent Memory flag is set.

25 To a 1 if the Clock Count enable is on.

26 To a 1 if the Clock Count flag is set.

28 To a 1 if the PC Change enable is on.

29 To a 1 if the PC Change flag is set.

31 To a 1 if the OV enable is on.

32 To a 1 if the OV flag is set.

33–35 Set to the priority channel assignment.

Priority Interrupt System

Device Number 004_8 (000 000 1)

Mnemonic PRS

The 166 Processor will honor interrupt requests on a priority basis at any time between the beginning of an instruction cycle and the end of the effective address calculation. However, if the instruction being executed is the result of an interrupt and is at one of the locations 40 + 2J, it can not be interrupted at any time during the instruction cycle. The processor honors the request by executing the instruction in location 40 + 2J, where J is the channel number. The program counter is not affected by the interrupt unless the trapped instruction is a jump instruction.

If a jump, it is left to the programmer to save the program counter if he has any intention of returning to the interrupted program sequence.

It is intended that the instructions executed by an interrupt, that is, the instructions located in locations 40 + 2J, should be either BLKI, BLKO, JSR, or XCT which executes one of the preceding three. While other instructions are not illegal, their results could be unfortunate. For example, the contents of the program counter could be lost, making it impossible to return to the main program.

The priority interrupt system is controlled by the CONO and CONI instructions by referring to device number 000 000 1 and setting bits in the effective address as follows:

CONO

23	If 1, clear the priority interrupt system.
24	Activate an interrupt on the channels selected by bits 29–35.
25	If 1, turn on the channels selected by bits 29–35.
26	If 1, turn off the channels selected by bits 29–35.
27	If 1, turn off the priority interrupt system.
28	If 1, turn on the priority interrupt system.
29–35	If a bit is 1, it selects the corresponding channel, with channel 1 in bit 29 to channel 7 in bit 35.

CONI

28	If 1, the priority interrupt system is on.
29–35	If a bit is 1, the corresponding channel is on.

Once an interrupt has occurred on a given channel, the channel must be dismissed by a BLKI or BLKO instruction, or a JRST or CONO instruction. (See the instruction descriptions for details.) If the break is not dismissed, there will be no further interrupts on lower priority channels. While servicing a device, the channel may be interrupted by a request on a higher priority channel. If so, the service to the lower priority device can be resumed by the instructions which dismissed the channel.

Input-Output Programming

Since the purpose of this section is merely to discuss the use of the I/O operations of the Type 166 Processor, particular I/O devices are not considered. Rather, the discussion and programming examples are presented for a generalized device. All that is required is that the device be defined sufficiently like the devices found in a PDP-6 system.

For some degree of definiteness, consider a generalized input device. The device is like those for the PDP-6 system in that it has a control register and a data register. The operation of the device is completely specified by bits of the control register which may be set by a CONO instruction. Further, the device control module can set some of the bits to reflect the status of the device, and the device may be requested to place the contents of the control register on the data lines by a CONI instruction. Finally, a DATAI command will cause the device to place data from its data register on the data lines for transmission to the processor.

The device may be turned on to read one character by setting the Busy flag (a designated bit in the control register) to 1. When the character is ready in the data register, the device clears the Busy flag to 0 and sets the Done flag (also a designated bit in the control register) to 1. A DATAI instruction will result in the contents of the data register being transferred through the data lines to the processor followed by resetting the Busy flag to read the next character and clearing the Done flag.

If a priority channel has been assigned by a CONO instruction, which places a channel number in the low order three bits of the control register, setting the Done flag to 1 will cause an interrupt on the assigned channel.

When using I/O instructions, we must include a device number or a device mnemonic. In the examples which follow for the generalized device, we will use the mnemonic MNE.

Such a device could be programmed without the priority interrupt system as follows:

```
CONO, MNE,  Y
CONSO, MNE, X
JRST, .—1          .—1 means this location minus one.
DATAI, MNE,  E
```

The CONO instruction transmits the effective address Y to the device control register to set the Busy flag and to specify the operation. CONSO matches the effective address X against the contents of the control register to detect when the Done flag is set to 1. JRST is an unconditional jump to CONSO. When the Done flag is set, the program skips to the DATAI instruction which reads the data in and moves it to location E in memory.

In the priority interrupt mode, the same operation can be carried out as follows. The priority interrupt system and channels are turned on by a CONO instruction. Then, using channel 1 we could write the following program:

```
CONO, MNE, Y + 1, + 1 to assign PI channel at location 42,
and beginning at INPUT
INPUT, EMPTY REGISTER FOR PROGRAM COUNTER AND FLAGS
DATAI, MNE, E
JRST, 12 @ INPUT
```

Where @ is an assembler symbol meaning jump indirectly to the location specified by the right half of location INPUT; that is, resume the interrupted instruction.

In this way the program could continue without waiting for the device. As soon as possible after setting the Done flag, the JSR at location 42 is executed. The control jumps to INPUT +1 with the flags and C(PC), still pointing to the location of the interrupted instruction, stored in INPUT. The DATAI is executed followed by the JRST. The number 10 in the accumulator field sets bits 9 and 11 to dismiss the priority channel and to restore the flags and program counter. It is important to note that, unless bit 9 is set, the channel will not be dismissed and it will be impossible for an interrupt to occur on all lower priority channels. Further, it may be desirable to add an additional CONO instruction to disconnect or disable the priority channel. Otherwise, additional interrupts will occur each time the device finishes reading in a new character.

To read in a block of information, the BLKI instruction may be used. Assuming the pointer for the BLKI has been properly set and is at location E, that is, the left half

of location E contains the negative of the size of the block and the right half contains the location of the first word of the block minus one, there are two options. Without the priority interrupt:

```
CONO, MNE,  Y
CONSO, MNE, X
JRST, .—1          .—1 means this location minus one.
BLKI, MNE,  E
JRST, MNE, .+2     .+2 means this location plus two.
JRST, .—4          .—4 means this location minus four.
```

When the entire block has been read in, BLKI causes the next instruction to be executed. Until then, it skips.

Using the priority interrupt system:

```
CONO, MNE, Y + 1, + 1 to assign channels at locations
42 and 43:
42, BLKI, MNE, E
    JSR, MNE, OFF
```

The BLKI instruction will automatically dismiss the priority channel after the word of data has been read in unless it is the last word of the block. If the last word, the JSR following the BLKI is executed. The sequence of instructions to which the JSR jumps must then dismiss or turn off the interrupt.

At location OFF a blank register could be followed by a sequence of one or more CONO instructions to turn off the device and/or priority channel and/or a JRST instruction.

Several devices can be assigned to the same channel, such as several Teletype stations in a time-sharing system. For this situation the instruction, in location $40 + 2j$, executed by the interrupt is a JSR to a subroutine. The subroutine consists of a sequence of condition skip instructions which examine the control register of each device assigned to the channel and jump to instructions to service the device whose Done flag is set. If only one device is serviced, and more than one is calling for an interrupt, then the other devices will immediately call for an interrupt when the channel is dismissed.

So far I/O programming has been discussed for a generalized input device. Programming for output is sufficiently similar to be given no separate detailed explanation. However, there are some minor differences. For example, rather than turning on the device with the same CONO instruction that sets the mode of operation, priority channel, etc. and waiting for the Done flag to give a data transfer command, with an output device we set the mode of operation with CONO and let the first DATAO instruction turn on the device and set the Busy flag. Further, the meaning of an interrupt is different for an input and output device. To be specific, consider the transfer of a block of data using the BLKI and BLKO instructions. When the JSR instruction following a BLKI is executed, the device has finished its present assignment. If, for some reason, it is desired that it not get the next word of data, a CONO instruction may be given immediately to clear the Busy flag. In contrast, when the JSR following a BLKO instruction is executed, the DATAO command has just been given, and the device will probably still be busy. If some other action is desired, such as writing an end of file on tape, we must wait for the Done flag.

CHAPTER 3
MEMORY

PDP-6 memory system configurations can be of almost any type, depending on system need. One processor can address several memory modules, and up to four processors can address a single module. Each module can therefore be a component in up to four memory systems. The core memory controls have a priority system used to determine which processor gets service. One processor has the highest priority; a second one, the second highest; and the third and fourth processors alternately share the third priority. For the Type 162 Fast Memory, a manual switch selects one of the connected processors.

The basic characteristic of any configuration is that each processor is connected to its memory modules through a bus. A function of the module is to recognize its address, to read and write information, and to put appropriate control information and data on the bus for the processor. Once it has received its instructions, the module performs these functions independent of the processor. In reading or writing, once the data has been taken off of the bus, the bus is free to carry requests from the processor to other memory modules.

To a processor, the memory system appears as one homogeneous unit. The processor simply places the appropriate data and/or commands on its bus and awaits the response of the memory. Should another processor have a memory request in progress in the same module, the later request is delayed until the module is free. Up to 262,144 words of memory can be directly addressed by one processor.

PDP-6 memory modules are of two types: Type 163 Core Memory (8,192 or 16,384 words, 2-microsecond cycle time) and Type 162 Fast Memory (16 words, 0.4-microsecond cycle time). In all memory configurations connected to the Type 166 Processor bus, the first 16 locations in memory are used as accumulators, index registers, and ordinary memory locations. Since the processor makes constant reference to these 16 locations, the use of fast memory here causes a significant increase in system speed. Figure 11 shows a typical memory system.

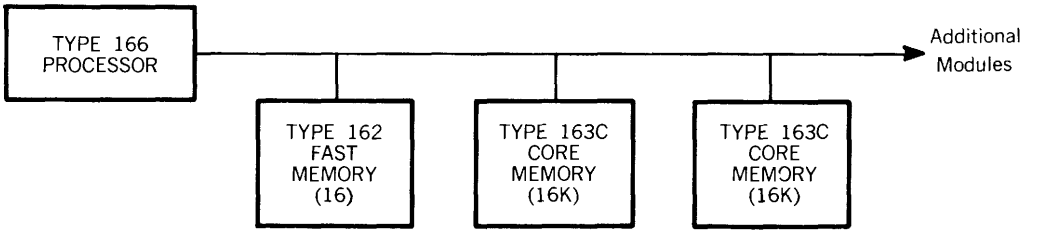


Figure 11 Typical Memory System

MEMORY FUNCTIONS

The processor can make three kinds of requests to memory: read, write, or read-pause-write. In each case the processor places the address of the module on the five module address lines and a request cycle signal on the request cycle line. It also specifies one of the functions to be performed, as described below. If the module is free, it returns an address acknowledge signal within 0.4 microseconds. If the module is busy, the processor waits until the module is free to acknowledge.

If the processor wants information from memory, it gives a read request, and when the data is available in the memory buffer, the memory module control places it on the bus. At the same time the memory module control sends a read restart signal to the processor, causing the processor to take the data off the bus. At this point the processor is free to continue its operations, which may be a new memory request, while the memory completes its cycle independently.

To write information into memory, the processor places the data on the bus along with the write request. As soon as the processor receives the acknowledge signal from the memory module, it is free to continue its operations. Again, this may include a new memory request.

The read-pause-write function can be used to advantage when the processor is to take data out of memory, perform a short operation (anything except multiply, divide, or floating point arithmetic), and store the results back in the same location. Read-pause-write is specified by requesting both read and write. When the processor receives the data and the read-restart signal, it performs the operation while the memory module remains connected. When the operation is completed, the processor places the results on the bus along with a write-restart command. It is then free to continue while the memory module completes its cycle.

ADDRESS SELECTION

Address selection consists of dividing the 18-bit address in the memory address register into a module address and a word address. The 5-bit module address is placed on the 5 module selection lines, and the word address is placed on the 14 address lines. As can be seen, the 5-bit module address and the 14-bit word address share 1 bit.

The exact method of determining the 5-bit module address depends on the system configuration. In general, four of the module address lines are connected to the four high order bits of the memory address register, while the fifth line is connected to any one of the fourteen low order bits. The fifth line is used in memory overlapping, as explained below.

Each module may be set to respond to one or more designated addresses by jumper connections within the module interface. The addresses are determined at each interface separately. Therefore, during the course of multiple processor system operation, different processors may address a given module by different sets of addresses.

A fast memory module added to the memory system replaces the first 16 locations in core memory for all processors connected to the fast memory module. When the fast memory module is addressed, bits 18 through 21, containing the module selection bits, are 0, and bits 22 through 31, the high order bits of the address selection, are

also 0. Sensing this condition, the processor sends a fast block signal to all memory modules, which enables the fast memory and inhibits the core memory. The first 16 locations in the first core memory module may, of course, be used by another processor not connected to a fast memory module.

MEMORY OVERLAPPING

In an asynchronous system it is highly desirable to interleave memory addresses in modules so that consecutive memory references are made to different modules.

For example, all even addresses can be assigned to one module and all odd addresses to another. The processor can then address a request to the other module without waiting for the previously addressed module to finish its cycle.

Since it is impossible, in general, to know where the next memory reference will be made in relation to the present one, there is no way to guarantee that the next reference will be to a new module. However, statistically there is a gain in performance if interleaved address ranges are assigned to different modules.

In PDP-6 memory systems the size of the interleaved blocks of memory locations can be any power of two from 1,024 to 8,192 depending on which of the 14 low order bits is connected to the fifth selection line. The module interfaces are set to respond to zero or one in the fifth position, depending upon the addressing scheme. When the fifth selection line is used in this manner, the module gets the word location part of the address from 13 of the 14 low order address bits and from 1 of the 4 remaining high-order selection bits.

MEMORY PROTECTION

A PDP-6 system can be concurrently shared by two or more users. In this case, the users' programs are assigned blocks of memory storage which may be any length from 512 to 262,144 locations in powers of two. Since an erroneous address in one user's program might destroy part of another's program in memory, automatic memory protection is provided.

The Type 166 processor contains two 9-bit registers, the protection register and the relocation register. When the system is in the protection mode, the contents of the protection register are compared with the high order nine bits of each memory address requested by the program. If any bits have ones in both the protection register and the memory address, an error is signaled and the memory cycle does not take place. If no error occurs, the address sent to the memory system consists of the original nine low order bits and the bit by bit inclusive OR of the contents of the relocation register and the high order nine bits of the requested address.

Before running a program in the protection mode, the protection register and relocation register are loaded by the executive program.

MULTIPLE PROCESSOR SYSTEMS

There are many possible systems in which more than one processor uses the same memory module or modules. Two examples of multiple processor systems are described below. In all such cases, the memory module bus interfaces are assigned priority levels: one interface has the highest priority, another the second highest, and the remaining two interfaces are alternately assigned the third highest priority.

Figure 12 illustrates a system in which two arithmetic processors share one memory module. A typical use of this configuration would be where incoming data was to be processed by A, using information stored in modules 1 and 2, then transferred to B for final processing in connection with information stored in modules 4 and 5.

Since a PDP-6 processor can be any device that has the proper interface to its memory bus, a drum processor can be combined in the system with an arithmetic processor. This combination is especially effective in time-sharing applications where core memory capacity may be limited. In Figure 13, one user's program is operating in the arithmetic processor and memory modules 1 and 3, while another user's program is being loaded into module 2.

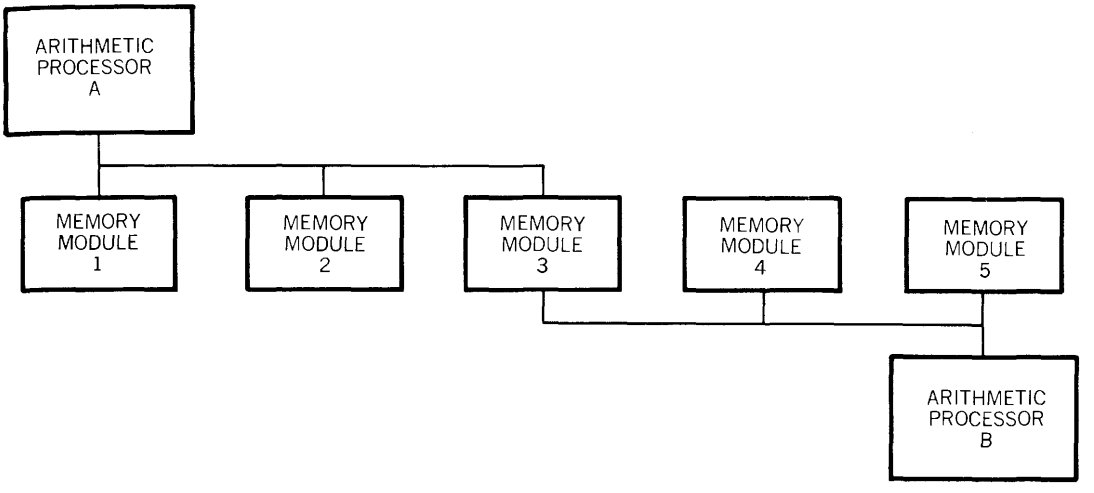


Figure 12 Partial Memory Sharing by Two Arithmetic Processors

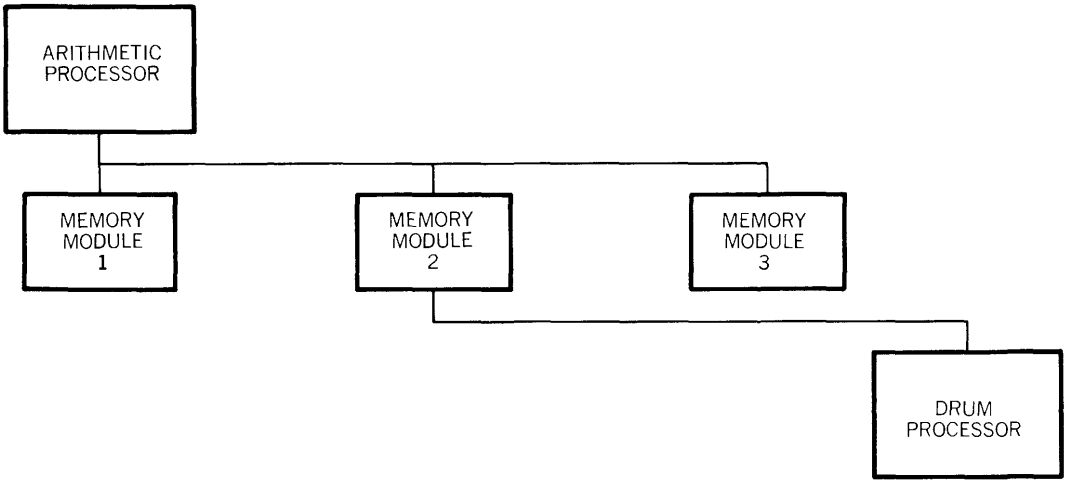


Figure 13 Use of Backup Storage in Time-Sharing Operations

The address selection schemes used by the processors may be entirely independent. Both processors may address the memory modules in the same way; one processor may address module 1 as the low memory field and module 2 as the high field, while the other processor addresses them in reverse order. One processor may have address overlapping while the other does not. In short, each memory-processor system functions as if there were no other system sharing its components. However, in programming the two systems to function simultaneously due regard must be given to the relationship between the address selection schemes.

It is impossible to illustrate all of the memory-processor systems that could be put together. A multiple processor system could conceivably consist of any number of processors and any number of words of directly addressable memory less than 262,144 times the number of processors, with memory modules interconnected in a network-like arrangement. The only restrictions would be that a single processor could not address more than 262,144 words, and that a single module could not be addressed by more than four processors.

Memory Timing

MEMORY MODULE TYPE 163

The Type 163B Core Memory Module contains 8,192 36-bit words, the Type 163C contains 16,384 words. The timing sequence for both is shown in Figure 14[1].

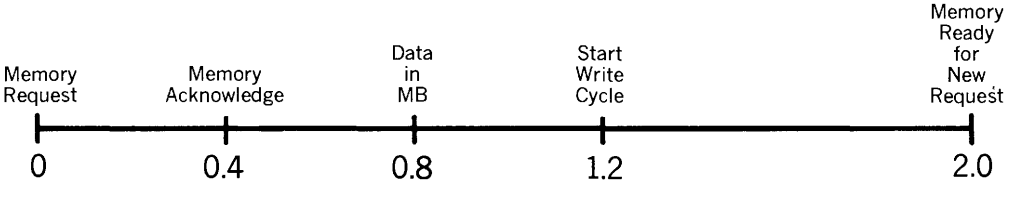


Figure 14 Type 163 Core Memory Internal Timing

Note that the data resulting from a read request is available in the memory buffer at 0.8 microseconds. At that time memory will signal the processor to take the data and proceed with its operation. During the write cycle, the data which has been read out of the specified memory location is written back into the location.

The write only mode of operation is exactly the same as the above except that the processor places data into the buffer register and is dismissed as soon as it receives the acknowledge signal, that is, 0.4μsec. While the memory goes through a read cycle, the contents of the buffer are not changed so that the information placed in it by the processor is written into the memory location during the write cycle.

In the pause mode of operation, the write cycle, shown above as beginning simultaneously with the end of the read cycle, will not begin until a signal is received from the processor.

1. Nominal times; actual times may be shorter.

MEMORY MODULE TYPE 162

The Type 162 Memory Module contains 16 36-bit words of flip-flop memory with a cycle time of 0.4 microseconds. Acknowledgement and data are available at the end of this time.

EXECUTION TIMES

The time required for the processor to execute a given instruction must include the time required for all necessary memory references. Consequently, the speed of a PDP-6 system depends on the memory configuration and on how it is utilized. Chapter 2 contains a flow diagram for calculating all instruction times and a table summarizing instruction times assuming various locations of the instruction and data. The following diagram illustrates the reduction of total instruction time that results when instruction, accumulator, and operand are in different memory modules. Assume that the instruction is in core memory, the accumulator in fast memory, the memory operand in a different core memory module than the instruction, and that the result is stored in the accumulator.

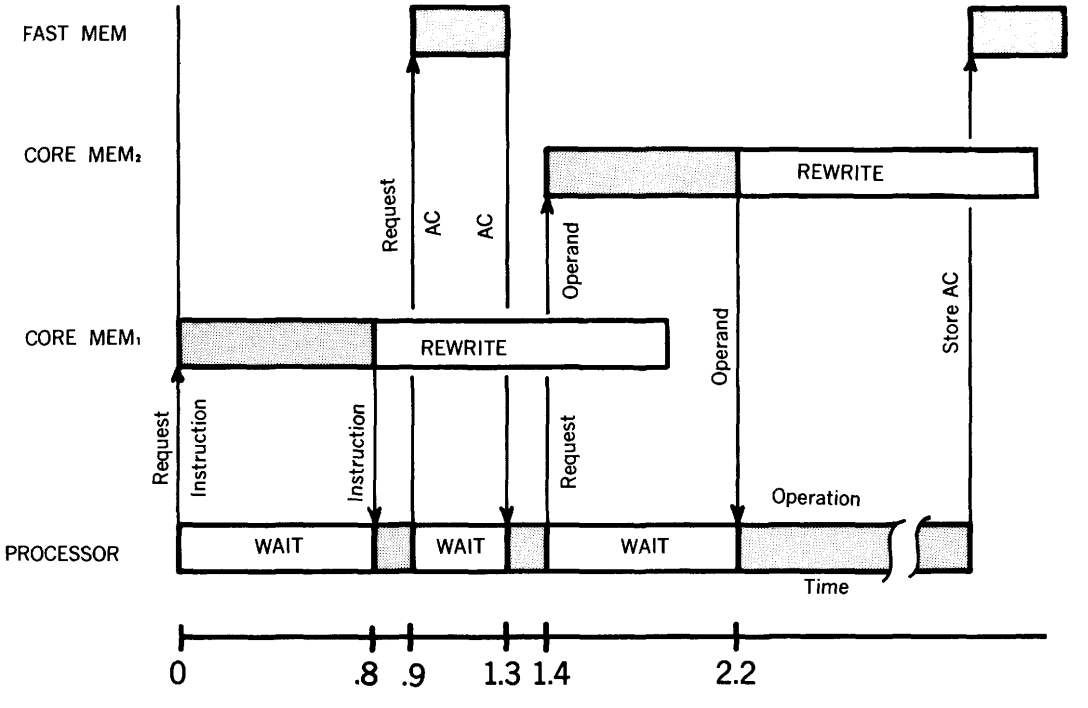


Figure 15 Typical Memory Timing Using Three Memory Modules in the Execution of One Instruction

CRT Display 346

Microtape 555

Line Printer 646

Tape Transport 570

CHAPTER 4
INPUT-OUTPUT

PDP-6 input-output systems consist of up to 128 devices, which may include different types of processors, connected in parallel through a common bus to a single processor. The processor, in addressing any I/O unit, sees only the interface of the single bus. Unit selection is done by placing a device number on separate lines of the bus. Each I/O device control has its own built in logic circuitry to decode its own device number and respond to an I/O command only when its own number is on the lines.

There are only four functions performed by the I/O control units: take initial conditions from the data lines, place status bits on the data lines, take data from the data lines, and place data on the data lines. To carry out these functions, the I/O controls are provided with separate data registers, control registers, and status registers.

The data registers vary in size from 6 to 36 bits. In the case where the data registers are less than 36 bits they are connected to the low order lines; that is, data enters and leaves the processor accumulator in the low order bit position.

The control and status registers vary in size up to 18 bits, and for most units are the same register. The processor is able to specify the function it wishes the device to perform by loading initial conditions into the control register, or it can determine the current state of any device by requesting that it place the contents of the control and/or status register on the lines. Whenever a device has a separate status register in addition to the control register, the status register is assigned an additional device number.

In the PDP-6 there are only four basic I/O instructions corresponding to the above four functions:

Conditions Out (CONO) Set the control register to specify the operation of the device.

Conditions In (CONI) Transmit the contents of the control and/or status register to the processor.

Data In (DATAI) Transmit the contents of the data register of the device to the processor.

Data Out (DATAO) Transmit the contents of the I/O register in the processor to the data register of the device.

The complete I/O instruction consists of bits to specify one of the above commands along with a 7-bit device number. Four additional instructions apply only to the Type 166 processor, such as skip on given conditions from the status register. They do not issue any new commands to the I/O system.

Priority Interrupt System

Since a device can only take data from the bus, or place data on the bus, upon simultaneously receiving its own device number and a command from the processor, there is no danger of conflicting usage. However, it does raise the question of how a device can signal the processor if it needs immediate attention. For this purpose a seven channel priority interrupt system has been provided. Any number of I/O

devices can be connected to any one of the seven priority interrupt lines. The connection is made whenever a conditions out instruction places the three bit binary number for the channel in the low order three bits of the device control register, zero being equivalent to no connection.

When any of the devices connected to a priority line needs service, indicated by predetermined bits in the control register being set to one, and the priority line is free, the priority line begins requesting a program interrupt. The processor will honor the request, with commands to the devices known to be on the line, if there are no higher priority channels (channels with a lower number) in an interrupt, and if it is not committed to finish its current instruction.

Input-Output Devices

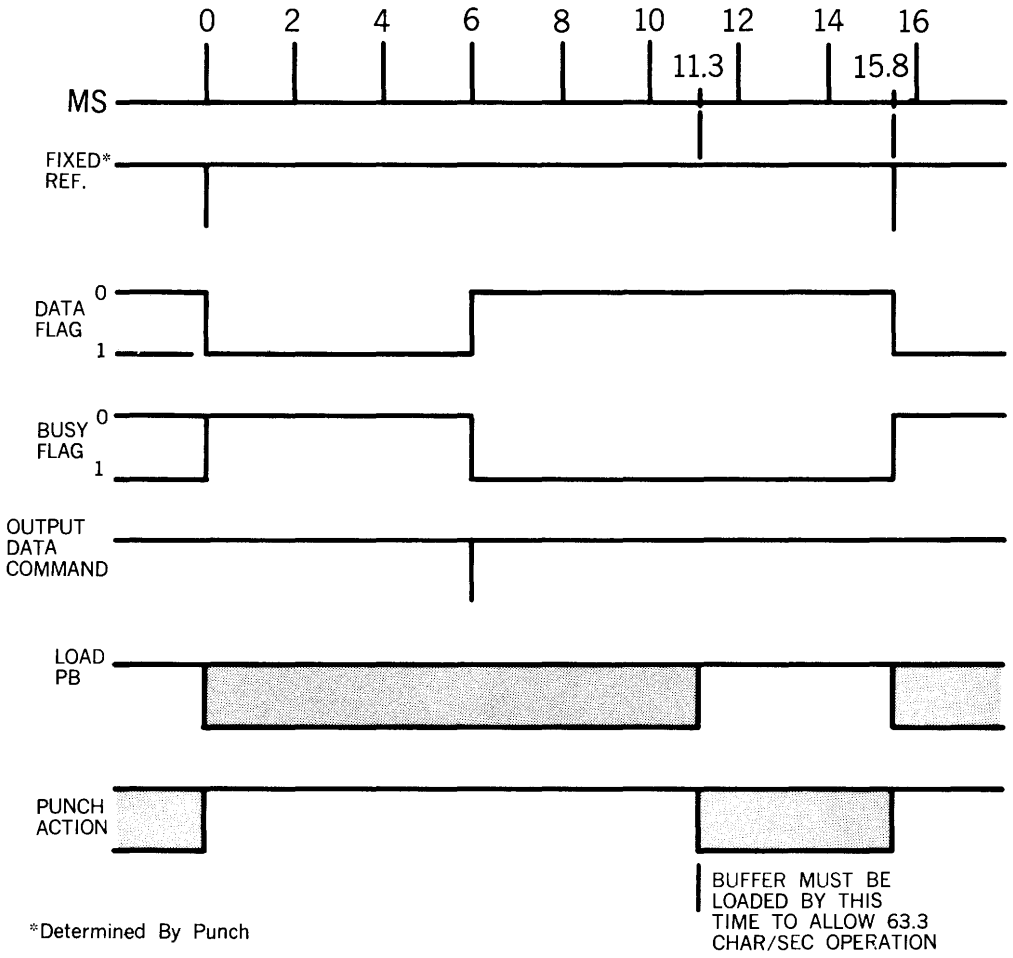
The input-output devices are listed with their device numbers in numerical order. Included under each device are a brief description and control word configuration. The essential information, such as the control registers, status registers, and operation codes, is summarized in a table at the end of this chapter.

PERFORATED TAPE PUNCH TYPE 761

Device Number: 100_8 (001 000 0)

Mnemonic: PTP

The Type 761 is a Teletype BRPE punch which perforates 8-hole tape at 63.3 characters (lines) a second. The following figure shows the timing for the punch.

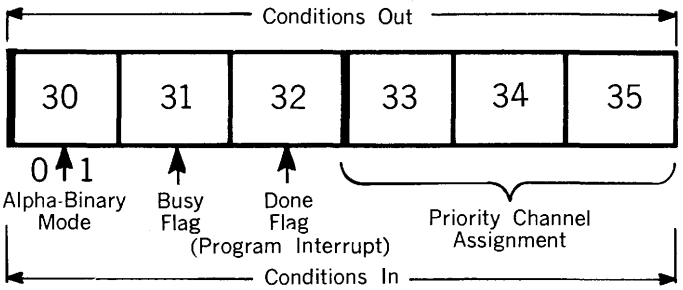


The punch can operate in one of two modes, alpha or binary. In the alpha mode, the eight bits of the data register are punched in all eight holes with ones only punching holes. In the binary mode, the eighth hole is automatically punched and the seventh hole punch is suppressed. The low order six bits in the data register are punched in the remaining holes.

Since the punch data register is only eight bits, information to be punched should be located in the low order eight bits of the I/O registers of the processor.

An automatic timing switch prevents punching for one second after the first setting of the Busy flag to allow the motor to reach punching speed. If no data-out commands appear for five seconds, the motor is turned off.

The control register configuration for the perforated-tape punch is given below.



If bit 30 is 0, the punch is in the alpha mode; if a 1, the punch is in the binary mode.

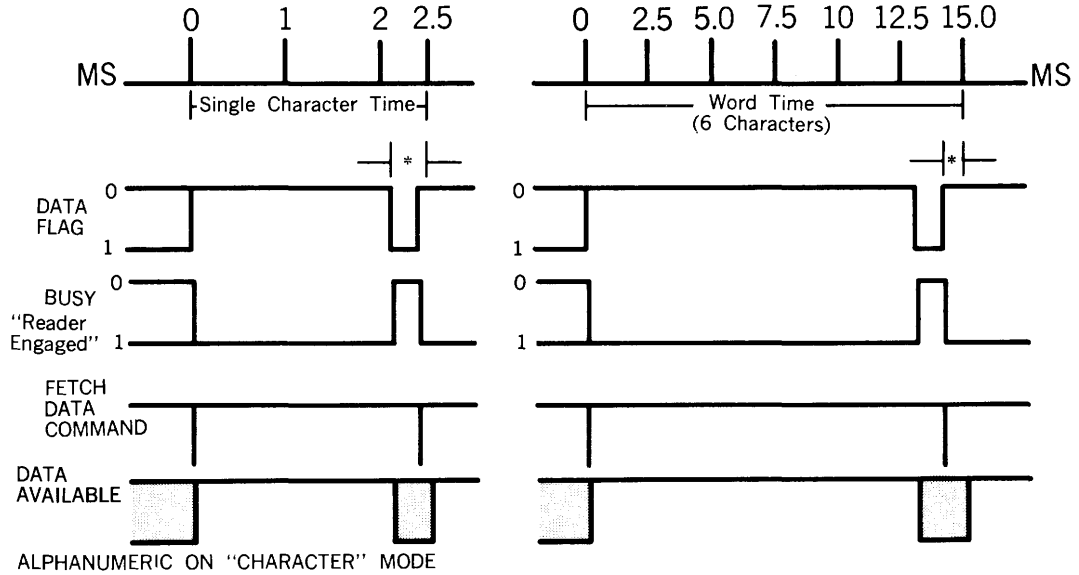
To program, use a conditions-out instruction to set the mode bit and the Busy flag to start the motor, followed by a data-out command. Additional data-out commands may be given when the Done flag is raised. The data-out commands will clear the Done flag and set the Busy flag.

PERFORATED TAPE READER TYPE 760

Device Number: 104_8 (001 000 1)

Mnemonic: PTR

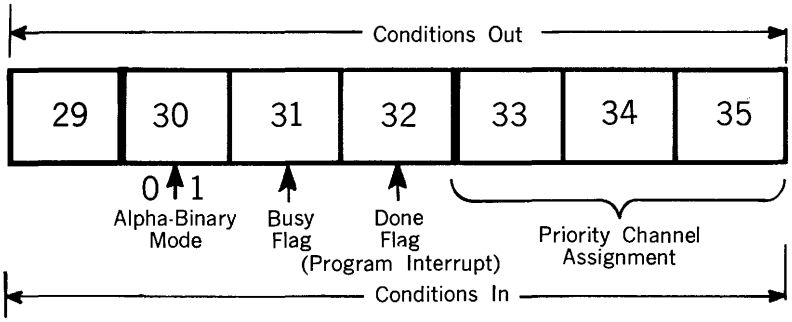
Depending on the guide setting, the perforated-tape reader is capable of reading photoelectrically 5-, 7-, or 8-hole perforated-paper tape at 400 lines a second. The figure below shows the timing for the reader.



*The next "fetch data" command must be given during this 400 microsecond interval to keep the Reader running at maximum rate

The reader can operate in one of two modes, alpha or binary. In the alpha mode, all eight holes are read into the data register, and the device clears the Busy flag and sets the Done flag. In the binary mode, a line of tape is read only if the eighth hole is punched. The seventh hole is ignored and the remaining six are read. Six rows are read for each read command, and the results are assembled into a 36-bit word in the data register. The first line goes into the high order bits, and each succeeding line goes into the next lower order position. When the word is complete, the device again clears the Busy flag and sets the Done flag.

The control register configuration for the perforated tape reader is given below.



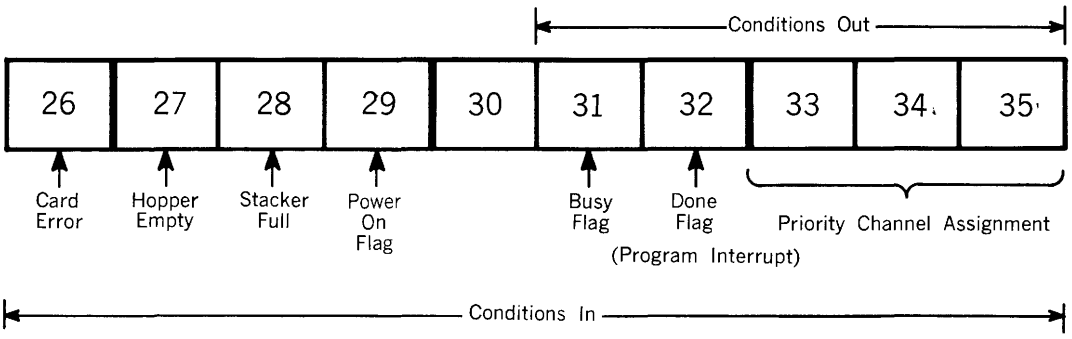
If bit 30 is 0, the alpha mode is specified; if a 1, the binary mode is specified. To instruct the reader to read one data word, set bit 30 to 0 or 1 and bit 31 to 1 with a conditions-out instruction. When the data-in command is given. the Busy flag is set to 1, and the Done flag is cleared causing the reader to get the next character.

CARD PUNCH CONTROL TYPE 460

Device Number: 110_8 (001 001 0)

Mnemonic: CP

The Card Punch Control Type 460 permits punching standard IBM cards at 100 cards a minute. Following is the control register configuration for the card punch control.



To program the card punch, 80 words are transmitted 12 times a card. The first time the words are transmitted, bit 24 of each word is loaded in sequence into an 80-bit buffer by the card reader control module. Then row 12 is punched. The next time the 80 words are transmitted, bit 25 of each word is loaded into the buffer to control punching for row 11. This process is repeated with bits 26-35 until rows 0-9 have been punched. A 1 in any bit means punch a hole.

To begin punching a card, a conditions-out instruction should be used to clear the Done flag and set the Busy flag. Setting the Busy flag will move a card into punch position. The Busy flag will then remain set until the entire card has been punched.

When the card is in position, the Done flag is set to request one word of information. The data-out command transmits a word of information and clears the Done flag. When the control module has finished reading the appropriate bit into the buffer, the Done flag is set to read the next word. When all 80 words have been transmitted, the row is punched before the Done flag is set to request again the first word of the sequence. Finally, when all 12 rows have been punched, the Busy flag is cleared by the device.

CARD READER TYPE 461

Device Number: 114 (001 001 1)

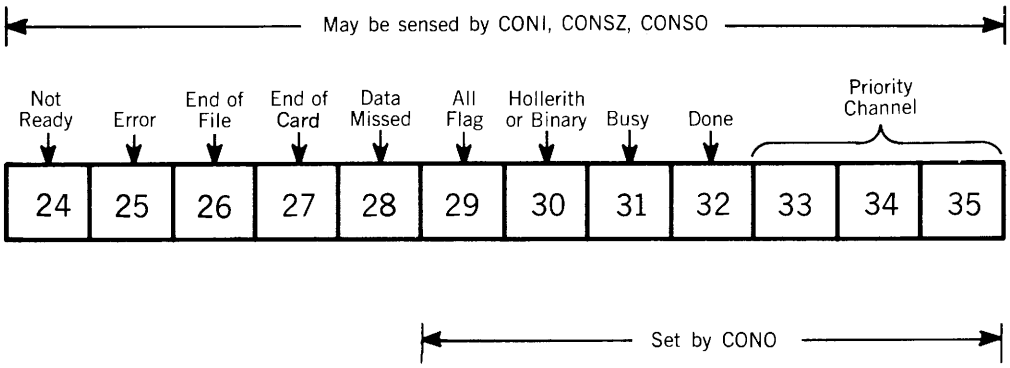
Mnemonic: CR

The Type 461 Card Reader reads standard 80 column cards at 200 cards per minute. Cards are read on a column by column basis.

The card reader reads in one of two modes: Hollerith or column binary. In the Hollerith mode the 12 bits of a column are converted to a 6-bit code. The column binary mode transmits the 12 bits of a column in unaltered form. In the Hollerith mode six 6-bit characters are packed in a word, and in the binary mode three 12-bit characters are packed in a word.

In the Hollerith mode the first word is composed of columns 1 through 6, the second of 7 through 12, the third of 13 through 18, etc. for a total of 13 words. Fig. 16 shows the reader card code. In the column binary mode the first word is composed of columns 1 through 3, the second of 4 through 6, the third of 7 through 9, etc. for a total of 26⅔ words.

The control register for the card reader has the following layout:



The significance of the bits of the status register is as follows:

24 Not Ready flag: Set to 1 if the power is off, validity check error (validity checking must be on), a card jam, card stacker is full, card hopper is empty, covers not in place, START button not depressed, or a read circuit failure.

25 Error flag: Is set to 1 if there is a failure in the card reader.

26 End of File: Is set to 1 when the END OF FILE button on the card reader is depressed to indicate that all cards have been read.

27 End of Card: Set to 1 when all 80 columns have been read. This bit sets the Done flag and clears the Busy flag.

28 Data Missed: Set to 1 if information in the buffer register is not read before the arrival of new data.

29 All flag: If set to a 1, the Done flag is raised after reading every column.

30 Mode: If a 0, the reader is in Hollerith mode;
If a 1, the reader is in column binary mode.

31 Busy flag: Set to 1 if the reader is in the process of reading a card.

32 Done flag: Set to 1 whenever a full word has been assembled unless the All flag is set, or is at the end of the card.

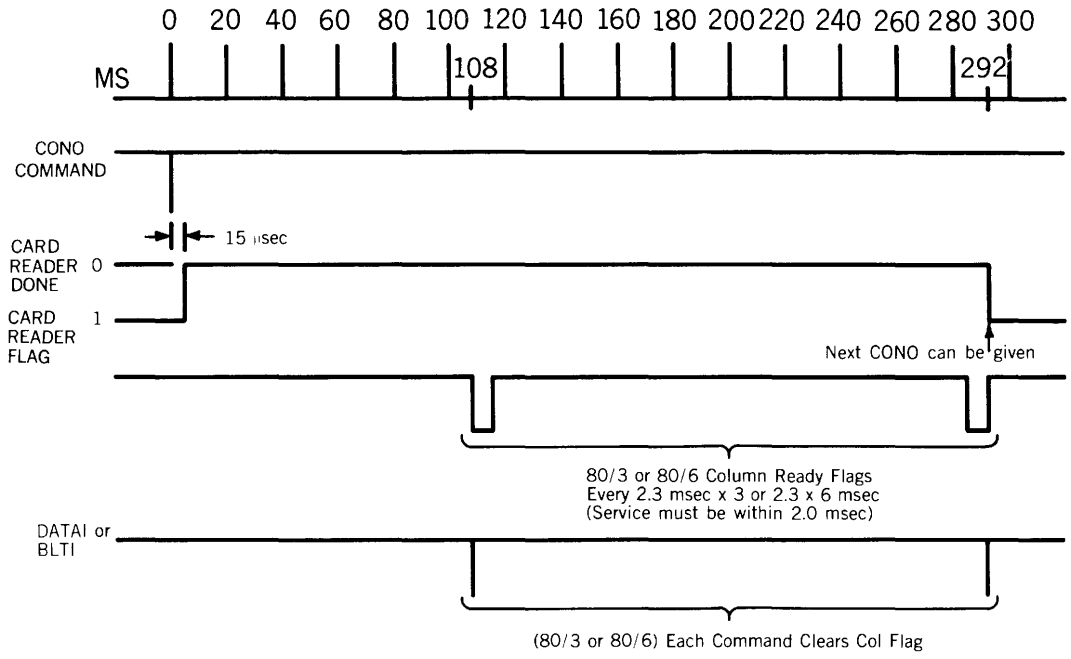
33-35 Priority channel assignment.

Card Code		Internal Code	Character
Zone	Num.		
—	—	01 0000	Blank
12	8-3	11 1011	.
12	8-4	11 1100	)
12	8-5	11 1101	]
12	8-6	11 1110	<
12	8-7	11 1111	←
12	—	11 0000	+
11	8-3	10 1011	$
11	8-4	10 1100	*
11	8-5	10 1101	|
11	8-6	10 1110	>
11	8-7	10 1111	&
11	—	10 0000	—
0	1	01 0001	/
0	8-3	01 1011	,
0	8-4	01 1100	(
0	8-5	01 1101	'
0	8-6	01 1110	#
0	8-7	01 1111	%
—	8-3	00 1011	=
—	8-4	00 1100	@
—	8-5	00 1101	↑
—	8-6	00 1110	'
—	8-7	00 1111	\
12	0	11 1010	?
12	1	11 0001	A
12	2	11 0010	B
12	3	11 0011	C
12	4	11 0100	D
12	5	11 0101	E
12	6	11 0110	F
12	7	11 0111	G
12	8	11 1000	H
12	9	11 1001	I

Card Code		Internal Code	Character
Zone	Num.		
11	0	10 1010	↑
11	1	10 0001	J
11	2	10 0010	K
11	3	10 0011	L
11	4	10 0100	M
11	5	10 0101	N
11	6	10 0110	O
11	7	10 0111	P
11	8	10 1000	Q
11	9	10 1001	R
0	8-2	01 1010	;
0	2	01 0010	S
0	3	01 0011	T
0	4	01 0100	U
0	5	01 0101	V
0	6	01 0110	W
0	7	01 0111	X
0	8	01 1000	Y
0	9	01 1001	Z
—	0	00 1010	0
—	1	00 0001	1
—	2	00 0010	2
—	3	00 0011	3
—	4	00 0100	4
—	5	00 0101	5
—	6	00 0110	6
—	7	00 0111	7
—	8	00 1000	8
—	9	00 1001	9
All other codes		00 0000	←

Fig. 16. Card Reader Codes

To read a card, the "card reader busy" bit is set to 1, the mode is specified, and the priority interrupt channel assignment is made. Once the CONO command is given, the reader will read all 80 columns in order. The end-of-card bit becomes a 1 after the last column is read. The end-of-card bit in turn, sets the Done flag and clears the Busy flag. At this time, columns 79 and 80 are read, and are in the buffer right-adjusted.



Card Reader Timing

If it is desirable to read a deck with both Hollerith and column binary cards, this may be done by setting the All flag to a 1 for the first column and the mode to column binary. The presence of a 7 and 9 punch indicates a binary card. Otherwise, the card is in Hollerith. A new CONO instruction may then be given. The instruction will clear the buffer register, and if given within 2 milliseconds, the first column of the card will be reread.

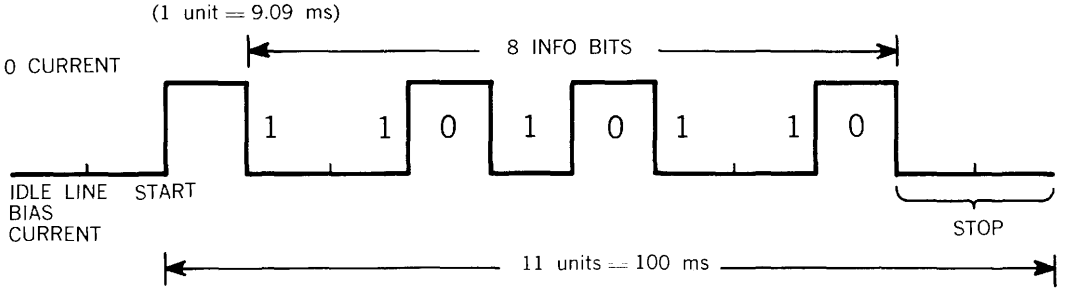
TELEPRINTER TYPE 626

Device Number: 120_8 (001 010 0)

Mnemonic: TTY

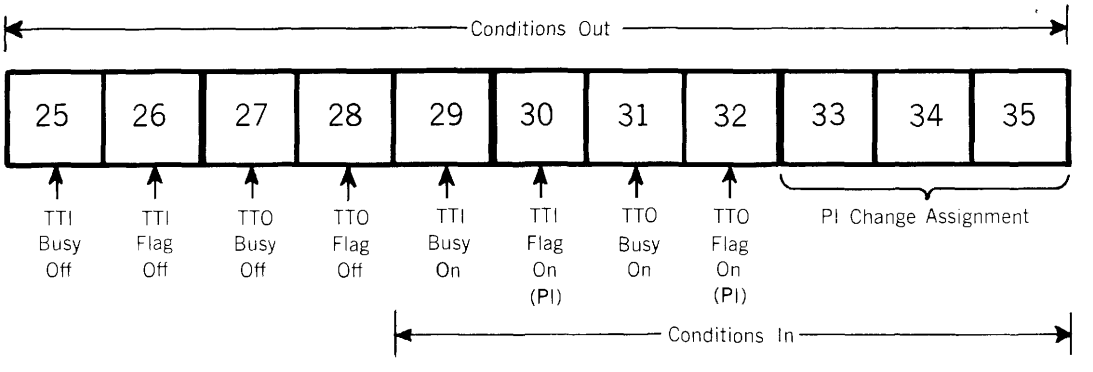
The Printer-Keyboard is a console Teletype Model KSR 33 which can print or receive ten characters a second as represented in Figure 17 by the Teletype eight-level code.

The signals to and from the KSR to the control unit are standard serial 11 unit-code Teletype signals. The signals are: start (1 unit), eight signals dependent upon the eight information bits (1 unit each), and stop (2 units). The figure below indicates the current pattern produced by the binary code 11010110.



TELETYPE TIMING OF INFORMATION CODE 110 10 110

The control register configuration for the printer-keyboard is given below.



Since the teleprinter consists of two independent devices, a keyboard and a printer, the control register must provide independent setting of TTI and TTO bits. Consequently, this control register functions differently from those of other devices. Only bits 33–35 can be cleared by a conditions-out instruction transmitting zeros. A conditions-out instruction can only set the remaining bits by transmitting ones; zeros are ignored. If, for example, the TTI flag is 1, the only way it can be cleared is by a data-in instruction or by a conditions-out instruction setting bit 26 to a 1.

Striking a key clears the TTI flag and sets the TTI Busy flag. When all eight bits of the code are available in the data register, which is eight bits long, the TTI Busy is cleared and the TTI flag is set. Execution of a data-in command clears the TTI flag. Note that since the keyboard and printer function as independent devices, striking a key does not cause printing. Printing follows typing only if a program transmits the input character back to the printer.

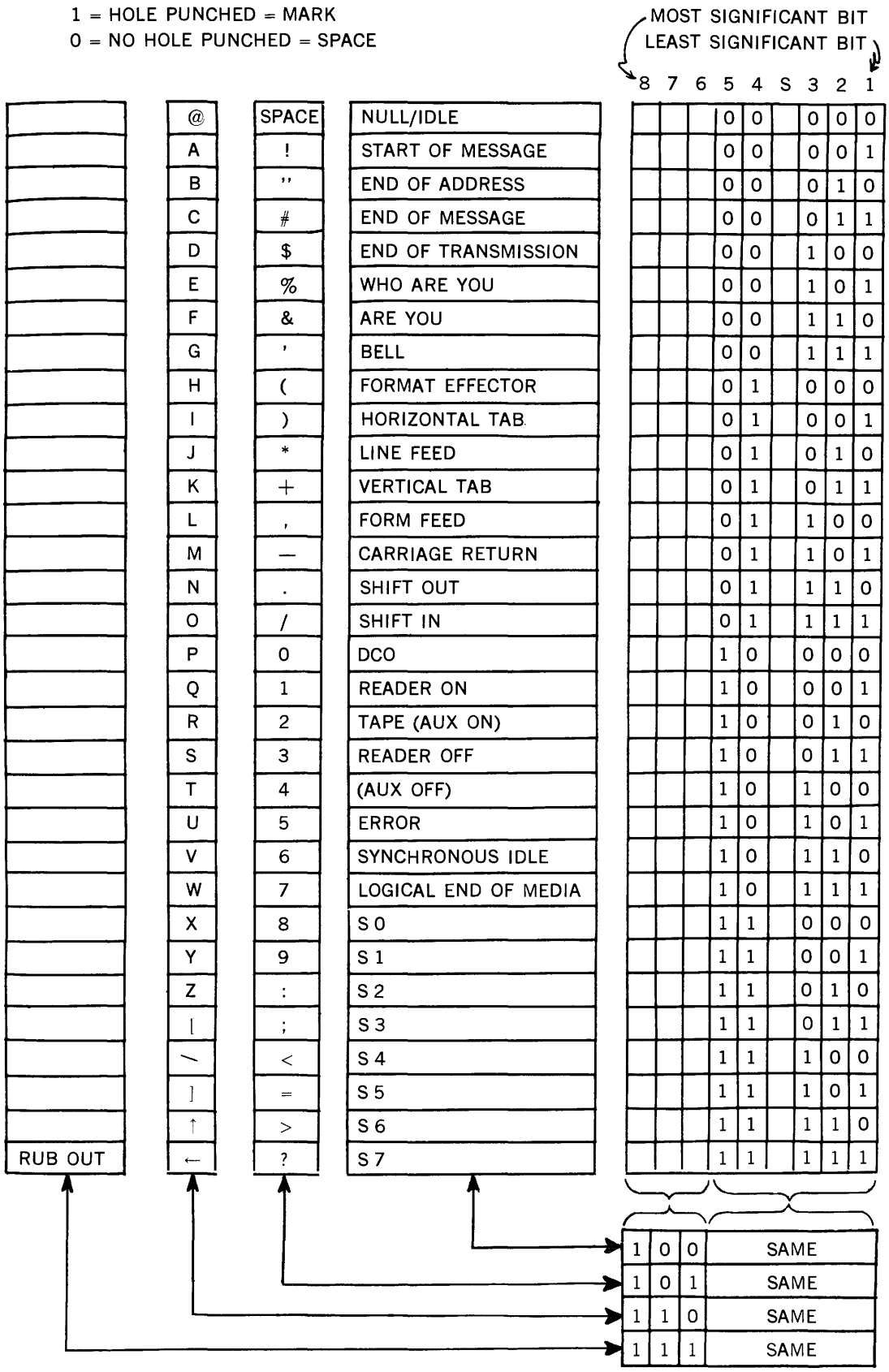
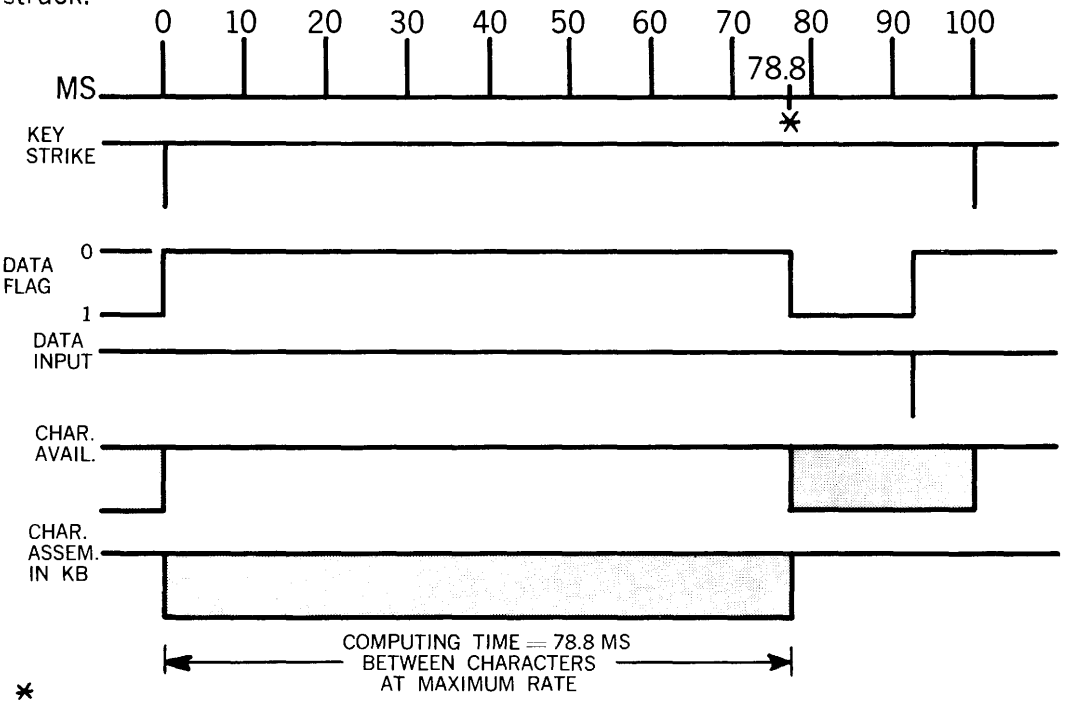


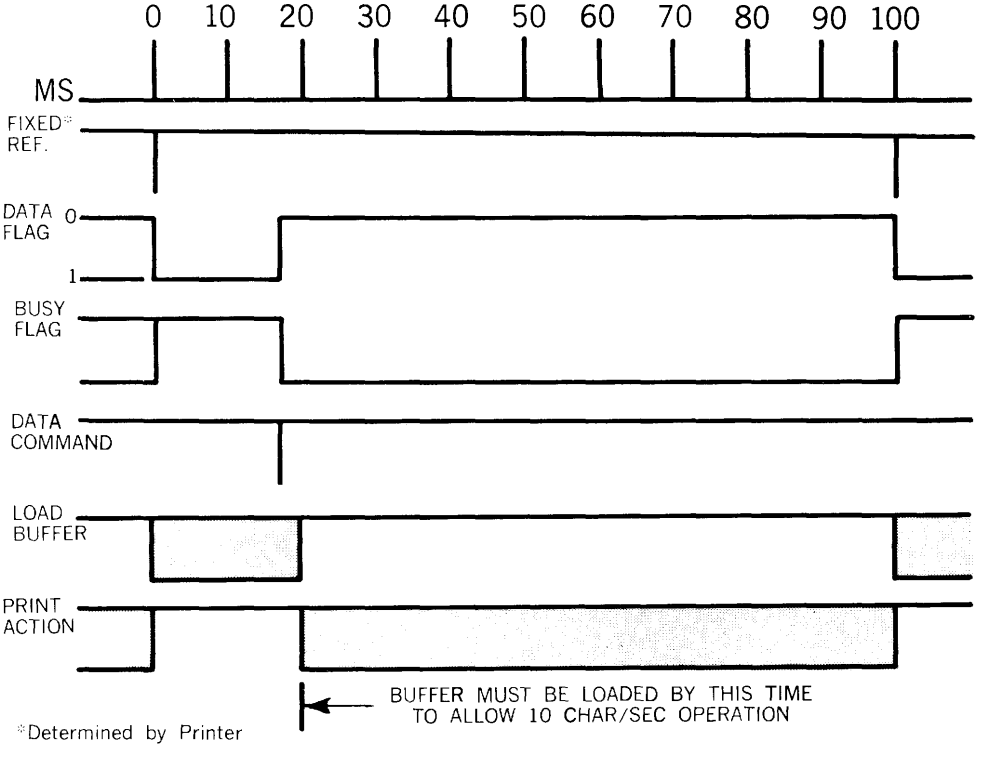
FIGURE 17. Teletype Eight Level Code

The figure below illustrates the keyboard timing; that is, the time required for the input to be assembled and available in the data register from the time the key is struck.



* Information Available at Middle of 8th Information Bit

For printing, a data-out command places an 8-bit code in the data register, clears the TTO flag, and sets the TTO Busy flag. When the character has been printed, the device clears the TTO busy bit and sets the TTO flag. The following figure shows the timing for the printer.



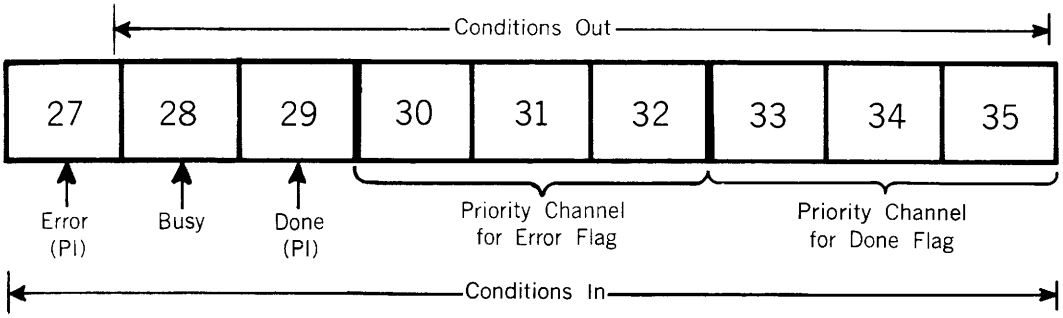
*Determined by Printer

LINE PRINTERS TYPE 646

Device Number: 124_8 (001 010 1)

Mnemonic: LPT

Three on-line printers are available: Type 646A operates at a maximum speed of 300 lines per minute; Type 646B at 600 lpm; and Type 646C at 1000 lpm. All use the same control. The control register configuration is:



The printer system contains a core buffer capable of holding up to 120 characters. Execution of a conditions-out command clears the data register, clears the control register and sets the priority channel assignments, and initiates clearing of the printer buffer. The conditions-out instruction automatically sets the Busy flag to a 1 and clears the Done flag. When the buffer is clear, the Busy flag is reset to 0, and the Done flag is set to 1. The Error flag is set if any condition exists which would make the printer inoperable.

When the Done flag is on (1), five 7-bit ASCII coded characters in bits 0–34 may be transferred to the printer data register by a data-out command. The data-out command automatically clears the Done flag and sets the Busy flag. The five characters are processed from left to right and sent to the printer buffer. When all five characters have been sent to the buffer, the Done flag is reset and the Busy flag is cleared. No more than 120 characters should be transferred without giving a print control character, as defined below.

The line printer control characters are a subset of the Type 626 Teleprinter control characters. All other teleprinter control characters are ignored by the printer; that is, an illegal code is simply shifted out of the data register and the next character, if legal, is sent to the printer. The legal control characters are:

HORIZONTAL TAB (011) — A horizontal tab of 8, 10, 12, 16, or 20 columns can be selected by means of a rotary switch mounted in the printer buffer. Upon receipt of this character, space characters are loaded into the printer buffer until the tab stop position in memory is reached. The next character sent to the buffer is printed in the column corresponding to the tab stop.

FORM FEED (014) — prints and spaces paper to top of the next page. The line of characters previously stored in the buffer is printed and paper motion started. Track 1 of the vertical format tape is selected (see below). The column counter is not reset; so the first character printed on the next page appears in the column to the right of the last character before the form feed.

CARRIAGE RETURN (015) — Prints out the line of characters stored in the buffer and resets column counter. Paper is not spaced.

EOT (004) — This character is used when there is no more printing to be done. It clears the Busy flag without setting the Done flag.

VERTICAL FORMAT TAPE — Whenever paper spacing occurs, the printer spaces until a hole is seen in the specified tape channel. Any carriage control tape may be placed in the printer, but a tape in the following format is supplied as standard:

Control Character	Track	No. of Lines to Next Hole on Standard Carriage Control Tape
Form Feed (014)	1	Skips to top of page
DC_0 (020)	2	1
DC_1 (021)	3	2
DC_2 (022)	4	3
DC_3 (023)	5	6
DC_4 (024)	6	11
Vertical Tab (013)	7	22
Line Feed (012)	8	33

In addition four blank lines are on the control tape before the line which has a hole in channel 1 (top of form). This insures that the bottom two and top two lines of each page are always left blank. This leaves 62 printing lines per page.

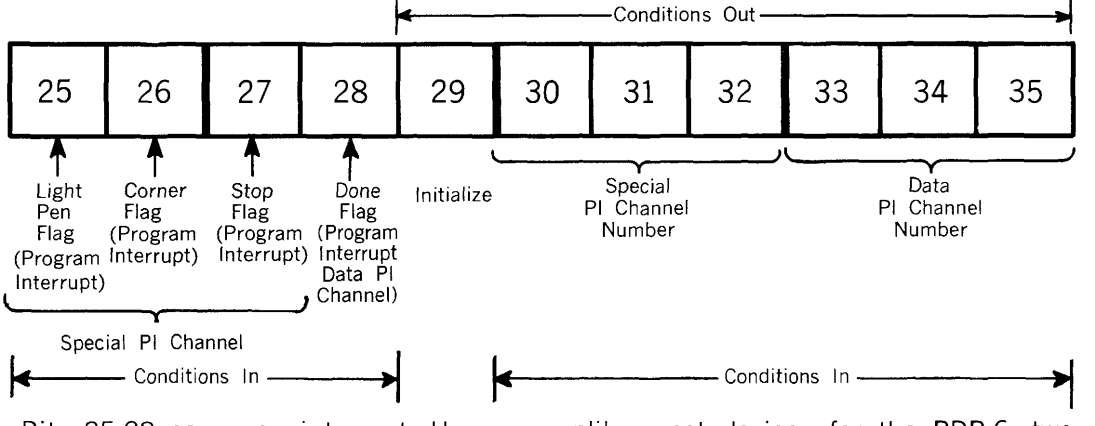
INCREMENTAL CRT DISPLAY TYPE 346

Device Number: 130_8 (001 011 0)

Mnemonic: DIS

The Type 346 is an incremental display with point plotting, line generator, vector, and character generator modes. The display must be used with the interface for PDP-6. Display Monitor Type 343, a slave display, can be used in addition to the Type 346 to provide additional cathode ray tube displays for multiple consoles.

The control register configuration is shown below.

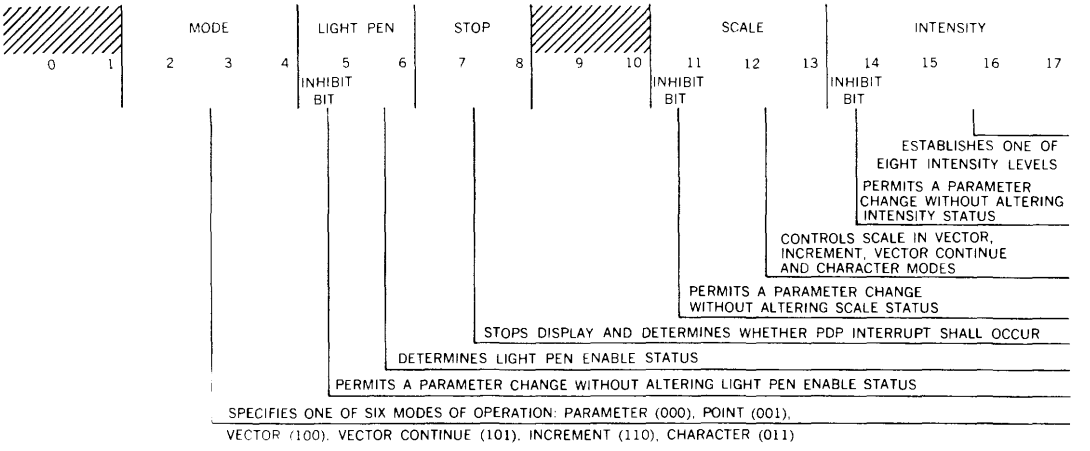


Bits 25-28 cause an interrupt. However, unlike most devices for the PDP-6, two priority channels may be assigned. The Done flag, bit 28, requests an interrupt on the data PI channel; the remaining interrupt flags request an interrupt on the special PI channel.

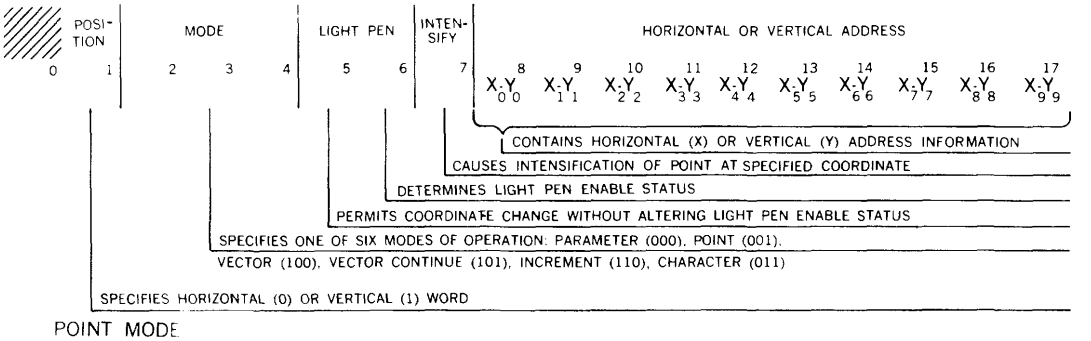
The Light Pen flag is set by the pen whenever it is in use and detects a spot on the scope. The use of the Edge flag and the Stop flag will be explained in the discussion that follows.

Whenever any interrupt occurs on the special PI channel, the programmer has the option of resuming operation with the existing conditions, or he may initialize the display by setting bit 29 to a 1 with a conditions-out instruction.

The Type 346 display is controlled by 18-bit control words, which are packed two to a PDP-6 memory word. The parameter and X-Y formats are shown below.



PARAMETER MODE



POINT MODE

In the X-Y mode the horizontal and vertical coordinates are identified by a 0 or a 1 respectively in bit one. All data is transmitted to the display as 36-bit words; the display control examines the left 18 bits first, then the right 18 bits. The order in which the X-Y words appear is not critical; however, when both horizontal and vertical words must be used to set up a single plot, faster operation is achieved by having the vertical word in the left half.

The bits in the X-Y mode have the following significance:

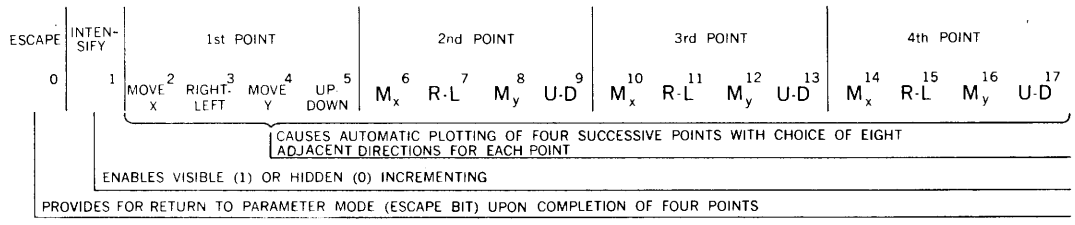
Bit 0	Presently unused.
Bit 1	Axis index. Must be a 0 to indicate a horizontal coordinate, a 1 to indicate a vertical coordinate.
Bits 2, 3, 4	These specify the mode of operation for the subsequent data. These bits are stored so that all data words following a single control word are interpreted in the mode specified by the previous control word. An escape mechanism is provided in each mode in order that the display can be forced to return to the control word mode for a change in position, parameter, or mode.
Bit 5	This bit enables a change in the light pen enable status, bit 6.
Bit 6	Subject to bit 5 being a one, this bit enables the light pen when a 1, disables the light pen when a 0.
Bit 7	Intensify; if a 1, a single point will be plotted at the specified coordinates.
Bits 8-17	These bits specify the horizontal or vertical coordinate, subject to the axis index, bit 1.

The bits of the parameter mode format are interpreted by the display as follows:

Bits 0	Presently unused
Bit 1	Presently unused
Bits 2-4	Mode bits which have the same significance as bits 2-4 of the X-Y mode.
Bits 5, 6	Light pen, with the same significance as bits 5, 6 of X-Y mode above.
Bit 7	Stop bit; if a one, the data request is inhibited and the display halts.
Bit 8	Interrupt on stop. Only if this bit is a 1 will the display interrupt the programming device on a stop.
Bits 9, 10	Presently unused.
Bit 11	If a zero this bit inhibits the setting of the scale bits, 12 and 13.
Bits 12, 13	These two bits control the size of the character in the character mode or the scale in the vector and increment modes.
Bit 14	If a zero this bit inhibits the setting of a new intensity level.
Bits 15, 16, 17	Used to select one of the possible intensity levels.

POINT PLOTTING MODE — This mode can be used to plot individual points located at random on the tube face. Whenever a new point is specified by the control word, there is a set-up delay of 30 microseconds.

THE INCREMENT MODE — In the increment mode an 18-bit data word, whose format is shown below, will cause the plotting of four successive points. These words are packed two to a PDP-6 word and are transmitted as 36 bits to the display. The display interprets the left 18 bits first followed by the second 18 bits.



INCREMENT MODE

Each point is specified with four bits which describe the direction of motion about the current spot location. With four bits, one of eight possible points immediately adjacent to the present spot location can be plotted. Each point requires 1.5-μsec plotting time. Four zeros in an increment character inhibit motion in either direction, and no point will be plotted. To increment the spot without intensifying, the intensify bit, bit 1, is made a 0.

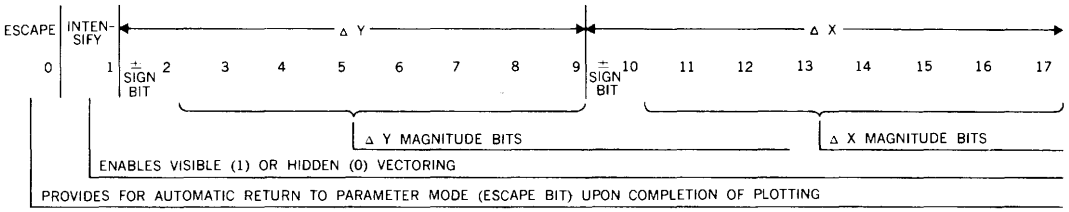
Once the display has been programmed to the increment mode, all incoming data is interpreted as increment mode data. Escape from the increment mode is accomplished by making bit 0 a 1. When the escape bit has been enabled, the four specified points are plotted, and the display returns to the parameter mode prior to the next incoming data cycle.

The vector continue mode enables the drawing of straight lines from any given point within the matrix to the edge of the screen with a single instruction. Any direction subject to the maximum (or minimum) delta-Y delta-X ratio can be achieved. An escape to the parameter mode and a request for data automatically result from a vector continue mode cycle.

When the scale bit is enabled, the X and Y registers are incremented by 2^n ($n = C(SC_0, SC_1)$) whenever a point is plotted. This has the effect of double spacing the points.

An Edge Flag signal is provided whenever the edge of the screen has been violated. An illegal Edge flag will stop the display while inhibiting the data request; a legal Edge flag, for example, at the end of a continue cycle, will not inhibit the data request. In either case, escape to the parameter word mode automatically occurs. The flag is reset when new data arrives.

THE VECTOR GENERATOR — The vector generator used in the vector mode provides a means for displaying straight lines between two points without specifying any in-between points. The vector mode data word, whose format is shown below, consists of eight bits of delta-X information, eight bits of delta-Y information, an intensity bit, and an escape bit. These words are packed two to a PDP-6 word and are transmitted as 36 bits to the display. The display interprets the left 18 bits first followed by the second 18 bits.



VECTOR MODE

Delta-X and delta-Y each comprise seven magnitude bits and one direction bit. Since the display area consists of a 1024 x 1024 point matrix, the maximum length vector which can be drawn with a single instruction is ± one-eighth of the display width. There is no limitation for a minimum length vector. Each point requires 1.5-microseconds plotting time.

The escape and intensity bits are used the same in the vector mode as in the increment mode.

THE CHARACTER GENERATOR TYPE 342 — The character generator provides a display of characters. Each character or symbol will be encoded with 6 bits, packed 6 to a 36-bit word. Of a total of 128 characters, the alphabet contains 123. Two characters are used in a pair of case-shift codes to achieve the required encoding capability. Upon specific character-like commands, the character generator also is capable of performing the space and carriage return functions which utilize two more characters. Escape from the character mode is accomplished with a character-like code and accounts for the 128th character.

DATA CONTROL TYPE 136

Device Number: 200

Mnemonic: DC

The Type 136 Data Control is an intermediate device between the processor and other external devices such as tape controls. The 136 serves as a data buffer stage and as a control device relieving the processor of exercising detailed control over the transfer of each individual transfer of data to and from the external device.

As a data buffer, the 136 assembles units of data from the external device, where the units may be 6, 12, 18, or 36 bits, into single 36-bit words for transfer to the processor. Similarly, it accepts 36-bit words from the processor and transfers them to the external device in groups of 6, 12, 18, or 36 bits.

As a control device, the 136 is capable of controlling up to 6 devices, one at a time. The 136 sends and receives all signals to and from the device necessary to control and synchronize the flow of data between the device and the buffer register. Also, it signals the processor when the buffer register is ready to send or receive a full 36-bit word. Figure 18 shows the logical flow of information through the 136.

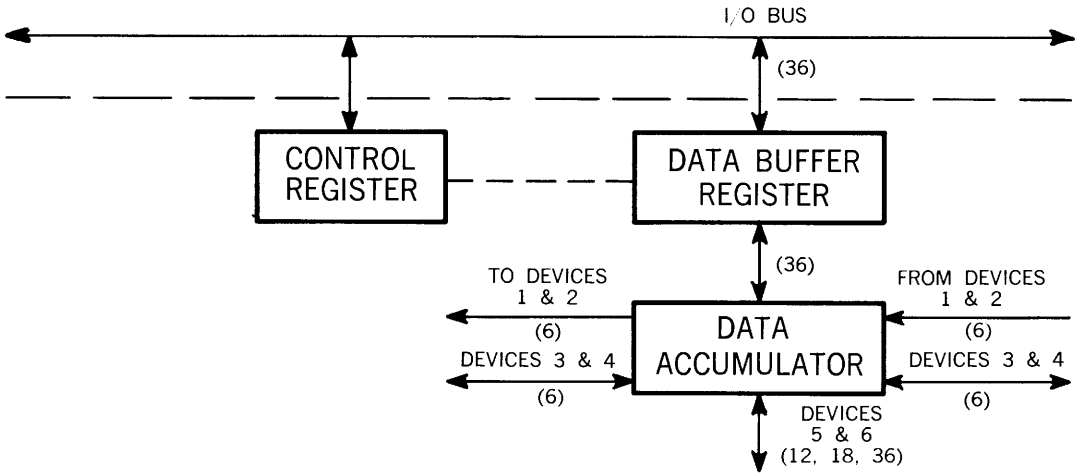


Figure 18 Data Control Type 136

For output, information is transferred from the I/O bus to the data buffer register where it is held until the data accumulator is ready for a new word. The information is then automatically transferred to the data accumulator and the data buffer Request flag is set to get more data from the processor. The data in the data accumulator is then sent to the desired device.

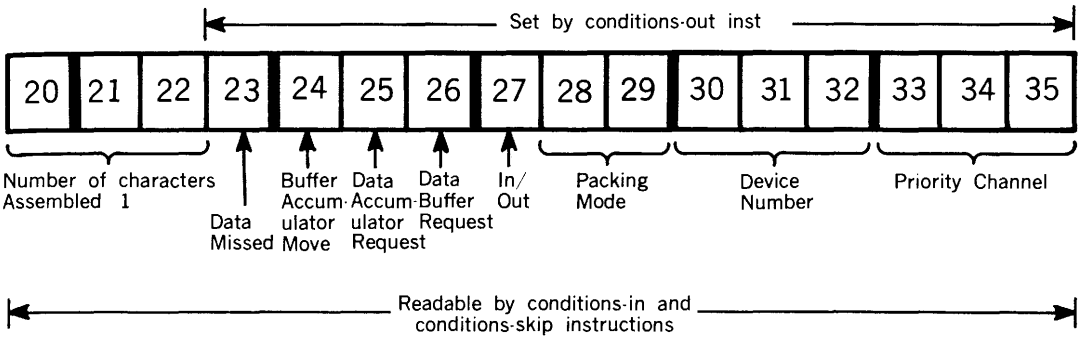
For input, information is transferred from the device to the data accumulator where it is assembled into a complete 36-bit word. It is then transferred to the data buffer register for transfer to the processor via the I/O Bus.

There are three ways in which data can leave the accumulator for the device, or enter the accumulator from the device. Which way is chosen depends on the device being used.

Devices 1 and 2 are 6-bit character oriented devices which can be run in two directions (microtape). Upon receipt of a Take/Give Character Left signal from the device, either six bits of data leave bits 0-5 of the accumulator for the device, if output, or six bits of data from the device enter bits 30-35 of the accumulator, if input, and the accumulator is shifted left six places. If the device were running in the opposite direction, it would issue a Take/Give Character Right signal in which case a similar sequence of events would follow. However, data would leave the accumulator from bits 30-35 and enter bits 0-5 followed by shifting the accumulator right. When all six characters have been shifted into, or out of, the accumulator, the accumulator Request flag is set to request a transfer of data between the accumulator and the data buffer register.

Devices 3 and 4 are 6-bit character oriented devices which can be run in one direction only (magnetic tape). They are exactly like devices 1 and 2 except for being unidirectional; they issue Take/Give Character Left signals only.

Devices 5 and 6 are unidirectional devices as are 3 and 4. However, they may be 12-, 18-, or 36-bit devices. Following is the control register configuration for the data control.



Bit Assignments:

20-22 The number of characters assembled or remaining in the data accumulator minus one.

23 Missed Data. This flag is set to one if the external device attempts to initiate a transfer of data out of or into the data accumulator before the Data Accumulator Request flag is cleared.

24 Buffer-Accumulator Move flag is a synchronization flag for the movement of data between the data accumulator and the data buffer. Data is moved when both the Move flag and the Data Accumulator Request flag are equal to one.

25 Data Accumulator Request flag is set to one when the correct number of characters have been assembled into the accumulator or shifted out of the accumulator.

26 Data Buffer Request flag is set to one when data is available in the buffer to be moved to the processor or when the buffer is ready to accept data from the processor.

27 In/Out If 0, data is to be transferred from the external device to the processor. If 1, data is to be transferred from the processor to the device.

28-29 Packing Mode
- 00 For six 6-bit units of data
- 01 For one 36-bit unit of data
- 10 For three 12-bit units of data
- 11 For two 18-bit units of data

30-32 Device Numbers 1-6.

33-35 Priority Channel Assignment

For output, the Buffer Request and the Data Accumulator Request flags should be set to ones and the Move flag to zero. A 1 in the Buffer Request flag will request the transfer of data from the processor, either by causing a priority interrupt or by being tested by a conditions-in instruction. When the data is set in the buffer, the Request flag is cleared resulting in the setting of the Move flag.

Since the Accumulator Request flag and the Move flag are both 1, the data is moved from the buffer register to the accumulator, the Data Accumulator Request flag and the Move flags are then cleared and the Buffer flag is set to 1, loading the buffer with a second word of data and again setting the Move flag.

The above sequence of events can be occurring while the external device is being made ready to receive data, for example, a tape unit being brought up to speed. As soon as the external unit has requested all the information from the data accumulator, the Accumulator Request flag is set and the above cycle is repeated.

For input, a conditions-out instruction is used to set the Move flag to 1 and the two Request flags to 0. As soon as the external device has requested the transfer of sufficient data to fill the accumulator, the Data Accumulator flag is set to 1. Since the Move flag was previously set to a 1, data is then moved to the Buffer Register, the Accumulator Request flag is cleared to accept the next unit of data from the external device, and the Move flag is cancelled. When the contents of the Buffer Register have been transferred to the processor, the Buffer Request flag is cleared and the Move flag is reset.

The data control may be used in or out of the priority interrupt. In the latter case, data flow is synchronized by examining the status bits with a conditions-skip instruction.

Microtape

Microtape is a bidirectional tape system in which data is written on, and read from, the tape in accordance with a prerecorded block format. Five tracks of information are recorded redundantly making a total of ten tracks on the tape. Two of the tracks are used as a timing track and a mark track. The timing track is used to synchronize reading and writing with each line of information on the tape appearing adjacent to a timing mark. The mark track is used to identify the type of information to be found in the three data tracks.

Figure 19 shows the sequence and types of data on the tapes.

Only the data words and block marks are transmitted to the PDP-6. The remaining words on the tape are used as control and error detection information by the Microtape control.

The blocks may contain any number of data words depending on the prerecorded formats. Indeed, each block may contain a different number of data words. However, block lengths of 128_{10} are considered to be standard format for the PDP-6.

Before Microtape is used for data storage it must be prerecorded. Prerecording is done in two passes. During the first pass, the timing and mark tracks are placed on the tape. During the second pass, forward and reverse block mark numbers, the standard data pattern, and the automatic check sums are written. Since part of the data word must be reserved to produce the mark track, it is impossible to write intelligent data in the information channels at the same time. For this reason, the two passes are required. Once these two passes have been made, the tape is said to be prerecorded and is ready for normal use.

The Microtape system includes a programmed mode of operation called write timing and mark track and a manual switch which permits wiring on the timing and mark tracks and also activates a clock which produces the timing track and flags for program control. Unless both the mode and the switch are used simultaneously, it is physically impossible to write on the mark or timing tracks. A red indicator lights on all transports connected to the appropriate control when the manual switch is in the "on" position. In this mode only, information channel 1 (bits 0, 3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33) is also connected to the mark track channel.

The actual mark track which is written on the tape (see Figure 19) was selected after careful consideration and provides many functions, such as:

a) Program synchronization
b) End of block detection
c) Error checking and prevention
d) Protection of control information
e) Block and word addressability
f) Automatic bidirectional compatibility

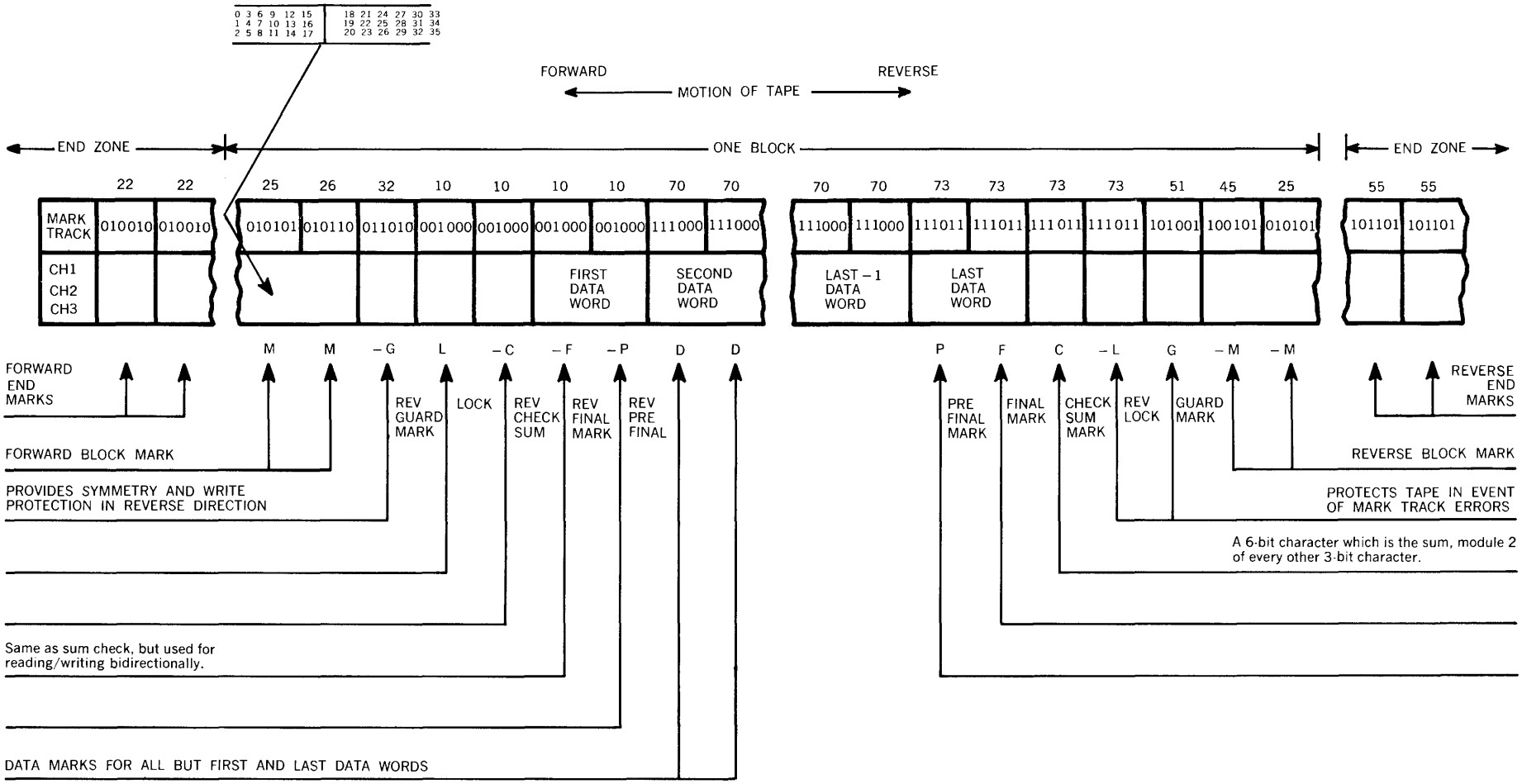


FIGURE 19 MARK AND INFORMATION TRACK FORMATS

g) End of tape detection
h) Variable block format
i) Inclusion of marks to allow expansion for various word lengths.

One other unique feature of the mark track is that the six control marks before the data marks are "complement obverses" of the six control marks after the data marks.* The data mark is the complement obverse of itself. Thus, when reading in the reverse direction the mark track window **sees exactly the same thing in both directions** since the flux reversals on the tape are opposite to those when reading forward and the bits are read in the reverse order. No special logic is required to distinguish the format of the tape in either direction.

The end marks on either end of the tape illustrate this bidirectional ability even better. As the end marks are complement obverses of each other, only that end of tape will be recognized at which the tape will physically come off the reel if further movement continues. Thus, no special hardware is needed for opposite ends of the tape, and there is no harm in coasting into or turning around in the end zones. Errors will be indicated only if attempting to go further into the end zone. The particular bit structure of the end marks is a repetitive one, so that any shift of three bits in the window will appear as another end mark. This makes it virtually impossible to pull the tape off the reel in any of the normal modes.

DUAL MICROTAPE TRANSPORT TYPE 555

The Type 555 Transport consists of two logically independent bidirectional tape drives capable of handling 260 foot reels of ¾ inch, 1.0 mil Mylar tape. The bits are recorded at a density of 375 (±60) bits per track inch. Since the tape moves at a speed of 80 inches per second, the effective information transfer rate is 90,000 bits per second, or one 36-bit word every 400 microseconds. Traverse time for a reel of tape is approximately 40 seconds.

The 3½ inch reels are loaded simply by pressing onto the hub, bringing the loose end of the tape across the tape head, attaching it to the take up reel and spinning a few times. Individual controls on the transport enable the user to manipulate the tape in either direction manually. The units can be "dialed" into a particular selection address.

There is no capstan or pinch-roller arrangement on the transport, and movement of the tape is accomplished by increasing the voltage (and thereby the torque) on one motor, while decreasing it on the other. Braking is accomplished by a torque pulse applied to the trailing motor. Start and stop time average 0.15-0.2 seconds and turn around time takes approximately 0.3 seconds.

RECORDING TECHNIQUE

The Microtape system uses the Manchester type polarity sensed (or phase modulated) recording technique. This differs from other standard types of tape recording, where for example, a flux reversal might be placed on the tape every time a one is desired. In the polarity sensed scheme a flux reversal of a particular direction indicates a zero while a flux reversal in the opposite direction indicates a one. A timing track, recorded separately in quadrature phase, is used to strobe the data tracks. Thus, the **polarity** of the signal at strobe time indicates the presence of a zero or one. Using the timing track on the tape as the strobe also negates the problems caused by variations in the speed of the tape.

*The complement obverse of a word is defined as the complement of a word with the bits read in the reverse direction, e.g.: 010110 (26) and 100101 (45), 001000 (10), and 111011 (73).

With this type of recording only the polarity, not the amplitude of the signal, need be considered, thus removing some of the signal to noise problems and allowing the use of read amplifiers with high uncontrolled gain. This recording also allows the changing of individual bits on the tape without changing the adjacent bits.

Reliability is further increased by redundantly recording all five of the information tracks on the tape. Figure 20 shows the placement of this track. This is accomplished by simply wiring the two heads for each information track in series. On reading, the analog sum of the two heads is used to detect the correct value of the bit. Therefore, a bit cannot be misread until the noise on the tape is sufficient to **change the polarity** of the **sum** of the signals being read. Noise which reduces the amplitude would have no effect.

MICROTAPE CONTROL TYPE 551

Device Numbers:
210 for Control Register
214 for Status Register

Mnemonics:
UTC for Control Register
UTS for Status Register

The Type 551 Microtape control contains all of the read and write circuitry as well as the circuitry for block detection logic and motion control logic. The 551 is capable of controlling four dual microtape transports or eight tapes. Unlike most devices which take their commands and data from the I/O bus directly, the Microtape controls work in conjunction with the Type 136 Data Control. Therefore, the 551 is controlled by both the control register and its status with respect to the data control.

Data transfers between the 551 and the 136 are on a 6-bit character basis. The data is assembled in the Data Control to form a full 36 bit word before being transferred to the PDP-6.

There are four modes of operation which require that the user either provide information to the microtape system or accept information from the microtape system. These are governed by the Microtape Control Type 551 and are:

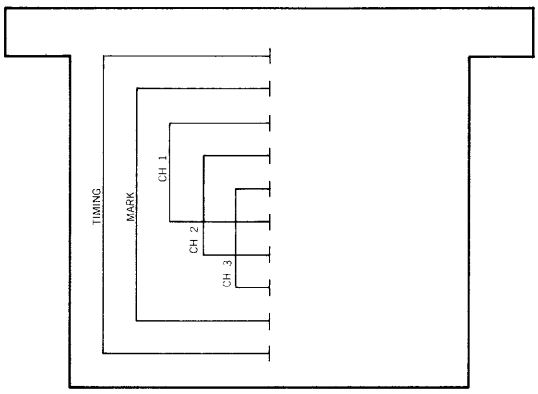


Figure 20 Placement of Microtape Tracks

1. Read or write data mode (Sum check is automatically read and written).
2. Read or write block mark number.
3. Read or write as a continuous record beginning at next reverse block mark.
4. Move tape and do not transfer data.

Figure 21 illustrates the use of the mark track for synchronization and the times at which commands can be given to the data control relative to mark track information.

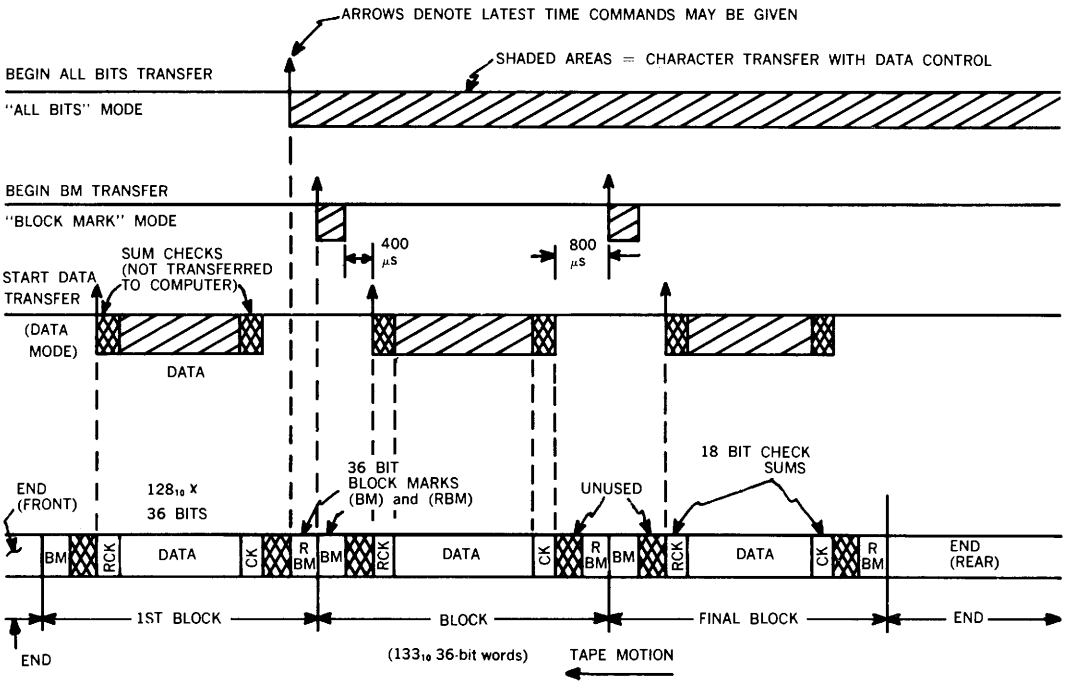
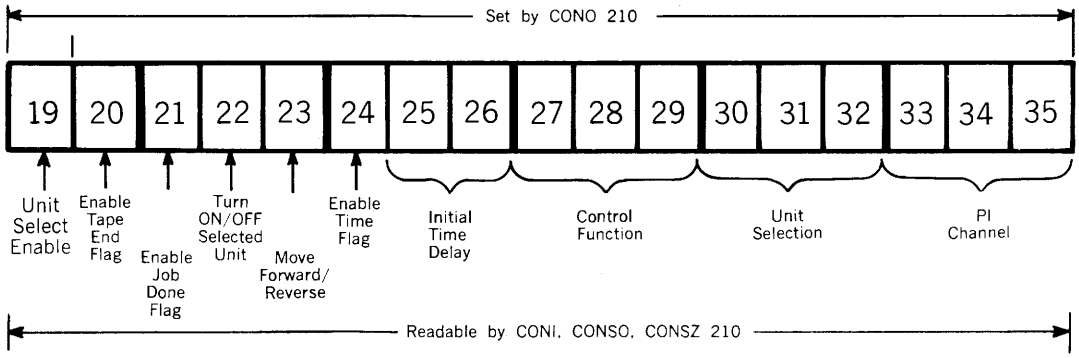


Figure 21 Microtape Format

In order to synchronize the control, the mark track is read by passing the bits through a 6-bit 'moving window' which shifts bit by bit as the tape moves. A decoder associated with the window interprets the pattern present, and takes appropriate action within the control.

The control register configuration is shown below. This register is set by a CONO with device #210 or read by a conditions-in instruction with device #210.



The bit assignments are as follows:

33-35 Sets the priority interrupt channel (1-7; 0 = no interrupt).

30-32 Unit selection (units 1-7 selected by 1-7, unit 8 selected by 0).

27-29 Tape Control function encoded:

27-29

000 Select unit and motion. No data transmission.

001 Read all tape as a single record, beginning with the next reverse block mark found.

010 Read block number. Gives a 36-bit block mark number and moves to job done status.

011 Read data and move to job done status if the data control is no longer accepting data or is deselected from the Microtape control.

100 Write timing and mark track as a single, continuous block. (The mark track is a copy of channel 0 on tape.) The WTM ENABLE switch must be "on" within the control.

101 Write all tapes as a single record beginning with the next reverse block mark found.

110 Write Block Number (36 bits) and move to job done status.

111 Write data and move to job done status at end of block if there is no more data to transmit from the data control.

25-26 Specifies 0, 20, 160, or 300 msec. Wait before transmitting any data, sets the Time flag at the end of 20, 160, or 300 msec.

25	26	
0	0	0 msec
0	1	20 msec
1	0	160 msec
1	1	300 msec

24 Interrupt enable the Time flag.

23 Move forward/move reverse. 0 = forward, 1 = reverse.

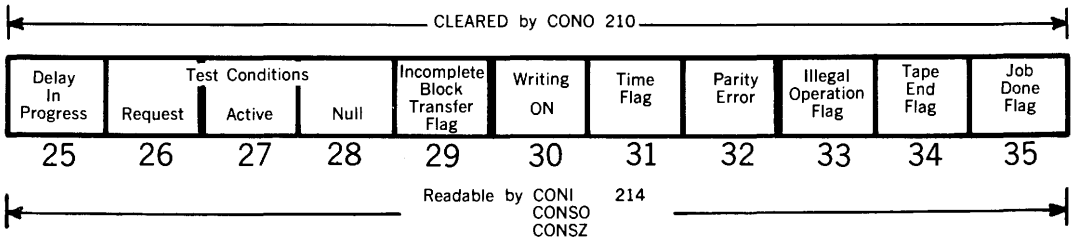
22 STOP/START the selected unit. 0 = stop, 1 = start.

21 Interrupt enable the Job Done flag.

20 Interrupt the Tape End flag.

19 Unit select enable. This bit must be a 1 to select a unit; if 0, all units are deselected.

Below is the configuration for the status register. This register is cleared whenever a CONO, 210, is given. It may be read by a conditions-in instruction with device number 214.



35 Job Done flag — signifies a task has been completed, and may be used to interrupt the program if Job Done Enable is a 1.

34 Tape End flag — Set to a 1 when the tape moves into its end zone and may be used to interrupt if the Tape End Enable is a 1. When the Tape End flag is set, the unit control flip-flop is turned "off".

33 Illegal Operation flag — Set if a write command is given, and a reel is write protected. Also set if no units or more than one unit is selected. The flag causes an interrupt.

32 Parity Error — Set if an error occurs.

31 Time flag — Set by the completion of the specified time delay and may be used to interrupt if the Time Enable is a 1.

30 Writing ON — A test condition which is a 1 during writing operations.

29 Incomplete Block Transfer flag — Set if a complete block has not been transferred, either by deselection of data control from Micro Tape control or data control is receiving or transmitting no more data.

28-27-26 Test conditions which specify state of the control. The control is in the null, active, or request state.

25 Delay in progress (test condition).

Figure 22 shows the operation of the Microtape control.

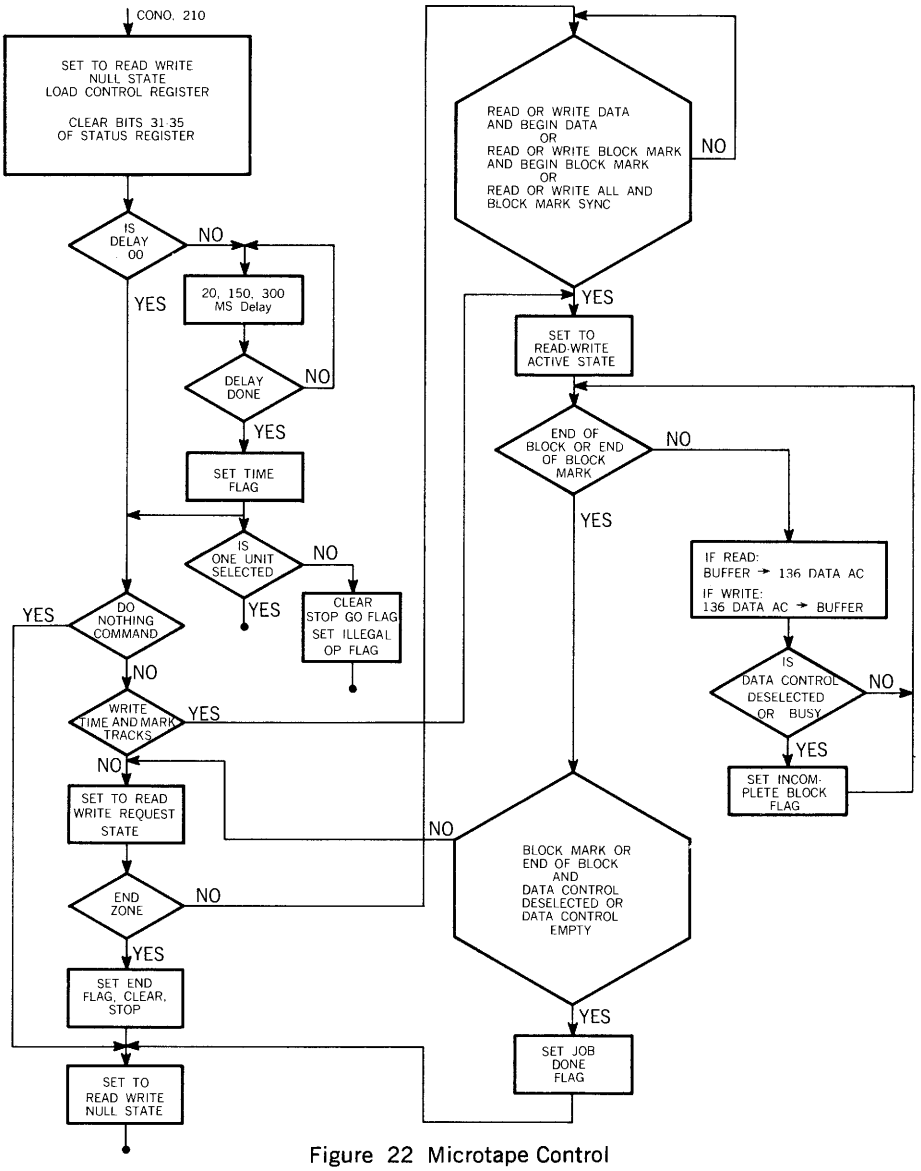


Figure 22 Microtape Control

MAGNETIC TAPE CONTROL TYPE 516

Device Numbers:
220 For Control Register
224, 230 For Status Registers

Mnemonics:
MT For Control Registers
MTS For Status Registers 224
MTM For Status Registers 230

The 516 operates with the 136 Data Control and 520, 521, or 522 transport interfaces. Operating through the 520 series interfaces, the 516 will control a maximum of eight transports to provide IBM compatible magnetic tape input or output.

Tapes may be written or read at 200, 556, or 800 densities depending on the transport connected. Tape speeds are 75 inches a second or 112.5 inches a second. At these speeds and densities the time between characters is: 66 μsecs, 24 μsecs, and 16.6 μsecs for the 75 ips speed; 44 μsecs, 16 μsecs, and 11 μsecs, for the 112 ips speed.

All tape functions are transmitted via the PDP-6 I/O Bus to the 516 Control by a conditions-out instruction setting the control register. Functions transmitted to the control are placed in a hold register until the previous function has been completed. If the same function is given before the previous one has been completed, the control acts as if it had been placed in a continue or proceed mode. This mode of operation terminates as soon as a different function is given or the previous command completes before a new function is transmitted.

However, the sequence of commands, read, space, read compare, constitutes an exception. Since these are all read commands in the forward direction, the transport would stay in the continuous mode of operation as long as the command hold register is filled.

If an illegal command is given, it will not be accepted. If an illegal condition arises during the execution of a command, the operation terminates, the illegal Command flag is set, and no more commands of that type are accepted in the continuous mode which is then terminated. The control functions defined are:

REWIND

The selected transport rewinds tape to the load point and stops.

REWIND/UNLOAD

Tape is wound off the take up reel of the selected transport.

WRITE

N characters may be written from consecutive or nonconsecutive locations of memory into one record. Binary (odd) or BCD (even) parity may be selected. When the write function is given, all BCD characters 00_8 will automatically be changed to 12_8. Otherwise, BCD 00_8 would be written as blank tape and could not be detected within a record except for the fact that a Status flag would indicate a missed character read while writing. All characters are read and checked for parity while writing. The LPCC (Longitudinal Parity Check Character) that is written is compared when it is read. Separate Status flags are provided for both types of possible parity errors.

WRITE END OF FILE

Write EOF is an automatic instruction and does not require the use of the 136 Data Control. The end-of-file mark is written 17_8 BCD. It is automatically detected during a read or space instruction.

WRITE BLANK TAPE

Writes a full reel of blank tape to end point. To write 3 inches of blank tape, the programmer gives a write EOF command and then a space back command. In either case, the 136 Data Control is not needed.

READ

N characters may be read in either parity mode. When the read function is given, all BCD characters 12_8 will automatically be changed to a 00_8. An additional read buffer is incorporated into the 516 design for generating a longitudinal check sum that is compared with the check character when it is read.

Whenever the specified word count is exceeded in the PDP-6, the 136 Control is disconnected from the 516, but the 516 may not finish its function until ERF is seen and the proper shut down delays have completed. If the word count has not overflowed before the ERF is seen, the transport is shut down unless the command hold register contains another command. If so, the word count may overflow within the next record. This implies gather reading or read continuous.

If a record ends containing a character count that is not a factor of six, the characters appear right-justified in the 136 Data Control. The program must test and read the 136 for the remaining characters.

READ BACKWARD

Identical to read except for the direction of tape motion. Tape recorded at 800 bpi cannot be read backward.

READ COMPARE

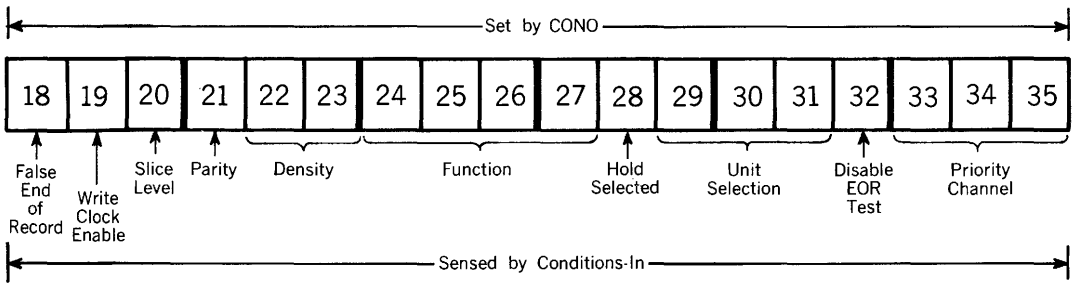
N characters are read and compared character by character with words stored in the PDP-6 memory. The 136 Data Control sends characters to the 516 where the comparison takes place. The comparison is performed in the LPCC read buffer. In this case, the LPCC is not read or compared. This function combines the capabilities of the read and write functions. An inequality sets the Read Compare Error Status flag.

SPACE FORWARD
OR BACKWARD

All spacing is done without the use of the 136 Data Control. Spacing forward or backward one record is automatic. If a bit is specified, tape is spaced until a file mark is read. Spacing continuously in one direction is achieved by loading the command hold register before the previous space command has been completed.

Regardless of the function given, the programmer can examine the status register whenever he pleases. A flag called ERF (End of Record) is supplied to the programmer through the 7 channel priority interrupt system. The flag may cause an interrupt to any one of the 7 channels. The flag is also available in the status register.

The control register is set or sensed by an I/O instruction with device number 220_8.



The significance of the control register bits is as follows:

33-35 Priority channel assignment

32 If a 1, the EOR test is disabled for maintenance purposes. An inconsistent format can be read. Every character on the tape is read.

29-31 Unit selection (0-7)

28 If a 1, do not return transport (Type 570) to pool.

24, 25-27 Tape control functions:

24	25-27	
0	000	No operation
0	001	Rewind selected transport
1	001	Rewind/unload selected transport
0	010	Write at selected transport
0	011	Write end-of-file mark
1	011	Write blank reel
0	100	Read-compare
0	101	Read
1	101	Read backward
0	110	Space forward one record
1	110	Space forward until file mark
0	111	Space backward one record
1	111	Space backward until file mark

22, 23 Density select

22	23	
0	0	200 bpi
0	1	556 bpi
1	0	800 bpi
1	1	556 bpi

21 Parity
If 0, BCD (even)
If 1, binary (odd)

20 Slice
If 0, high sense level
If 1, low sense level

19 Write clock enable

18 False end of record

Bit 28 of the control register requires some additional explanation. The Type 570 Transport may be operated in a pool whereby it can be selected by either one of two controls. Figure 23 illustrates such a pooled arrangement. When a control is finished with the transport, the transport is returned to the pool unless bit 28 is set to 1. In that case the transport can no longer be selected by the other control until some future time when it is returned to the pool.

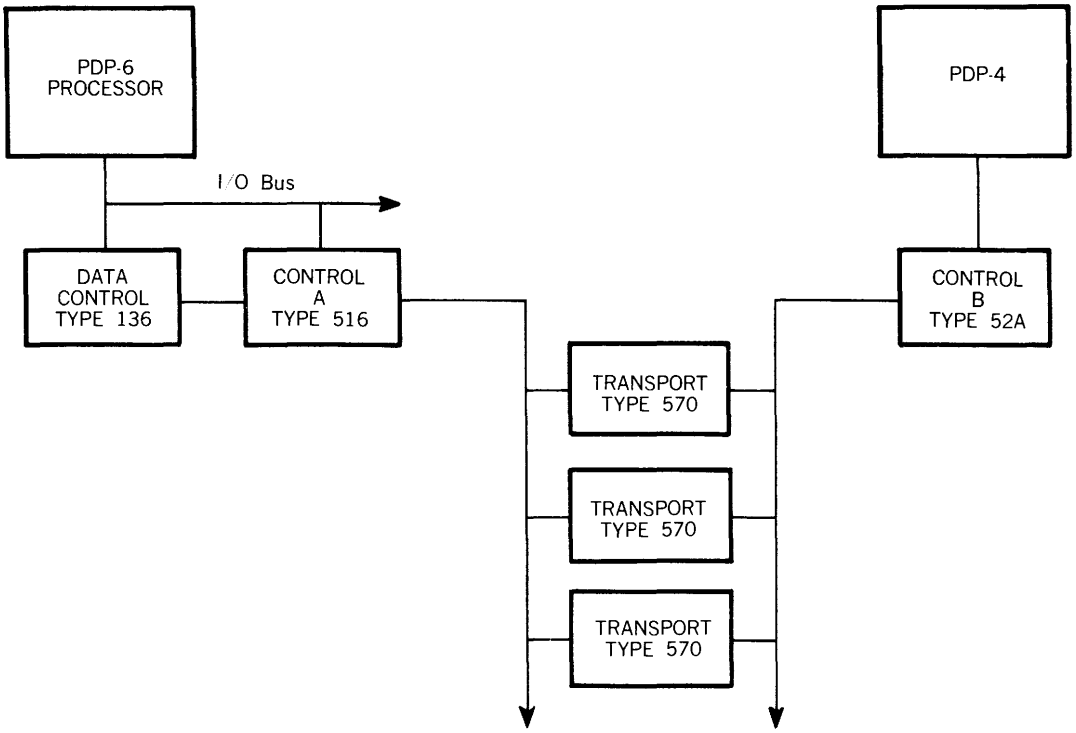
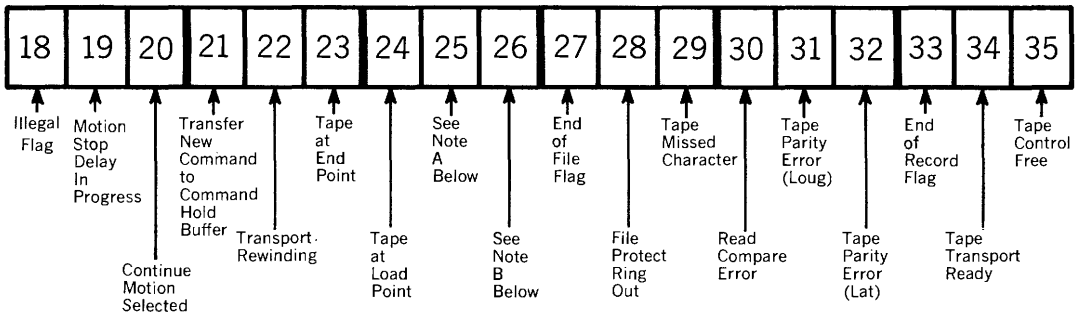


Figure 23 Pooled Transports

The 516 Control contains two status registers in addition to the control register. Neither of these can be set by a CONO instruction; both may be sensed by a conditions in instruction.

The configuration for the status register sensed with device No. 224_8 follows:



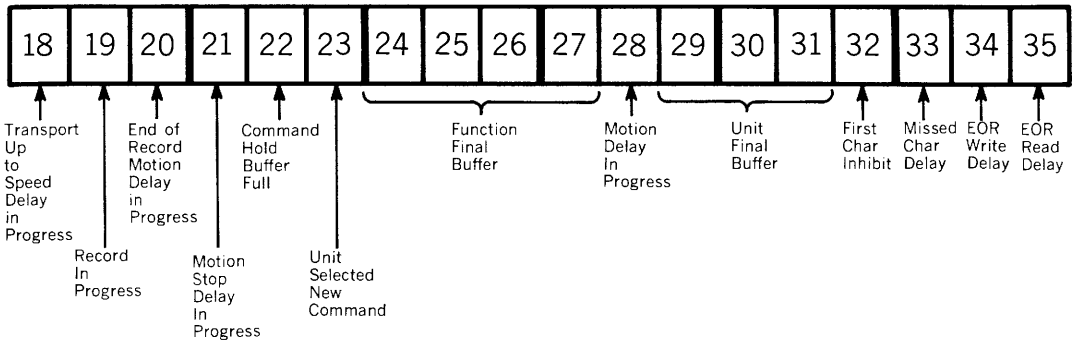
Note A

Tape near end point. (520 interface)
First operation write (521 or 522 interface)

Note B

Tape near load point if 520 interface
B Control using transport (521 interface) See Poo
Write Echo Not OK (522 interface)

Below is the configuration for the status register sensed with device No. 230_8.



DATA COMMUNICATION SYSTEM TYPE 630

The 630 Data Communication System (DCS) is a real time interface between Teletype stations and PDP-6. It is used for multiuser time sharing systems, message switching systems, and data collection-processing systems. Its basic function is to receive and transmit characters. When receiving, characters of different data rates and unit codes arrive from the Teletype stations in serial form. The DCS converts the signals to Digital voltage levels; the characters are converted from serial to parallel form and are forwarded to the computer. When transmitting, characters in parallel form are presented to the DCS by the computer. The characters are converted to serial Teletype form of the correct data rate and unit code, they are converted from Digital voltage levels to Teletype station signal levels, and they are sent to the Teletype stations.

The modularity and plugability of the 630 DCS simplify the expansion of the system from one station to the maximum number of stations. Various combinations of data rates, unit codes, station types, and station signal level can be accommodated in one 630 DCS.

The 630 System consists of 631 Data Line Interfaces, 632 Send/Receive Groups, and a 633 Flag Scanner. It has a maximum capacity of 8 groups (8 stations per group) or 64 stations (128 pairs of wires for full duplex operation). Figure 24 is a simplified block diagram of the system.

The Type 631 Data Line Interface converts Teletype station signal levels to Digital voltage levels, and converts Digital voltage levels to Teletype station signal levels. The extent of modularity of the 631 is dependent upon the type of station signals to be converted. The 631 is plug connected to the 632.

The Type 632 Send/Receive Group converts parallel characters to serial Teletype characters, or converts serial Teletype characters to parallel characters. It mixes the received characters of the eight Teletype stations onto a bus for presentation to the 633, and notifies the 633 when service is required.

When a character has been received or transmitted, a flag (indicator) is activated. The flag in turn notifies the 633 that service is required for that particular station. The manual OFF-ON switch mounted on the handle of the receiver and transmitter modules may be turned off to inhibit the flag from requesting service.

The Type 632 can accommodate a maximum of eight receiver modules and eight transmitter modules. The quantity required is dependent upon the number of Teletype stations. (If four half duplex stations are to be interfaced, only four receiver and four transmitter modules are required.) The type of each module required is de-

pendent upon the data rate, unit code, and the number of data bits. Teletype stations requiring different data rates, unit codes, and data bits can be intermixed in the Type 632. The receiver module disregards hits (noise) less than one-half of a unit in length on an idle line. The 632 is completely pluggable.

The Type 633 Flag Scanner decodes and interprets computer instructions, forwards received characters to the computer upon request from the computer, sends characters to the transmitter modules when instructed by the computer, scans each 632 in search of activated flags, notifies the computer when an activated flag has been found, and forwards the station number requiring service to the computer upon request from the computer.

The Type 633 contains a precision crystal-controlled clock that generates highly accurate timing pulses. The transmitter and receiver modules use the pulses to sample the serial Teletype signals. An additional crystal clock can be added to accommodate multiple Teletype speeds. A crystal clock is also used to generate timing pulses that control the search logic of the scanner. The scanning mechanism of the 633 is modular. Each expansion permits eight additional stations (1 group) to be scanned.

A rotating priority scanner notifies the computer when an active flag has been found. The computer program requests the station number and then handles the character. Programmed priority of the stations is permitted.

The flag scanner operates at the following speeds: the maximum total time required to examine 64 inactive stations, 32 microseconds; the maximum total time to search, notify the computer and continue to search for 64 simultaneously active stations, 544 microseconds (exclusive of computer interrupt and programming cycles); the minimum time required to find the next active station upon being released by the computer, 6 microseconds; and the maximum time required to find the next active station (station being serviced minus one) upon being released by the computer, 92 microseconds.

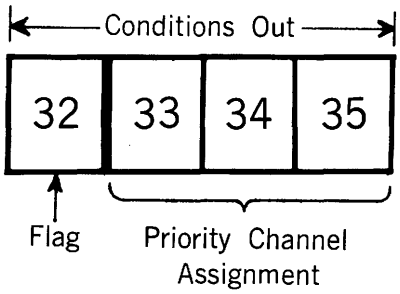
PROGRAMMING FOR THE 633
HALF DUPLEX

Device Numbers:
300_8 (011 000 000)

304_8 (011 000 100)

Mnemonics:
DCSA, DCSB

The control register is assigned to device number 300. Following is the control register configuration.



The line number counter and send buffer (used to select a transmitter on an idle line) are assigned to device number 304.

The effect of an I/O instruction depends upon the device number (mnemonic). The various effects may be summarized as follows:

Symbolic Operation	Function
CONO DCSA, E	The priority interrupt channel defined by bits 33-25 of E is assigned to the data communication system. The scanner is released if bit 32 is 1. The counter will be reset to 0 and the scanner will be released if bit 31 is a 1 (programmed priority).
CONI DCSA, E	The previously assigned priority interrupt channel is read into the C(E) bits 33-35. The state of the Scanner flag is read into C(E) bit 32 (1 = on or active). The C(E) bits 0-31 are cleared.
DATAI DCSA, E	The character in the receiver (as specified by the receiver counter) is read into bits 28-35 of C(E). The C(E) bits 0-27 are cleared; the Receiver flag is turned off. The scanner and/or Scanner flag are not affected by this instruction.
DATAO DCSA, E	The transmitter (as specified by the send buffer) is loaded from bits 28-35 of C(E). The Receiver flag is turned off.
CONO DCSB, E	The send buffer is loaded from bits 30-35 of E (used to select a transmitter on an idle line).
CONI DCSB, E	The line number in the receiver counter is read into bits 30-35 of C(E). The C(E) bits 0-29 are cleared.
DATAI DCSB, E	The character in the receiver (as specified by the receiver counter) is read into bits 28-35 of C(E). The C(E) bits 0-27 are cleared. The Receiver flag is turned off, the Scanner flag is turned off and the scanner is released.
DATAO DCSB, E	The transmitter (as specified by the receiver counter) is loaded from bits 28-35 of C(E). The Receiver flag is turned off, the Scanner flag is turned off and the scanner is released.

The method of transmitting is as follows:

1. A send buffer is provided to initiate transmission to an idle station. The program loads the send buffer with a specific station number. The program then instructs the 633 to send the character to the station (using the send buffer to select the desired station). The associated Receiver flag is cleared at this time. Note that the scanner is not released. Thus neither the position of the scanner counter nor the state of the Scanner flag need be considered when initiating transmission on a line. If transmission is initiated in multiple priority levels, it is suggested that the interrupt system be disabled during the setting of the send buffer and sending the character.

2. At the completion of sending a character the Receiver flag is turned on.

3. Steps 1, 2, 3, and 4 for receiving now are executed.

4. At this point, the program has the option of executing the following three steps.

5. The receiver receives all data sent on the telegraph line; therefore, the character received may be used as an echo check on data sent. The telegraph line is the connection between the transmitter and the receiver; so if the echo matches the data sent, the program is certain that the data was on the telegraph lines. The program would receive a character as in steps 5 and 6 under receiving for echo checking. (The scanner is not released (clearing of Scanner flag) at this point unless action of step 7 is desired.) This may be followed by step 6.

6. The program may transmit another character. Note that the scanner counter is used, the scanner is released and the Receiver flag is cleared. To insure maximum transmission rates, the program must reload the transmitter buffer within the following times (after service has been requested).

WPM	Bit Code	Baud	Unit Code	Time
60.0	5	45.5	7.5	30.0 msec
66.6	5	50.0	7.5	27.5 msec
100.0	5	75.0	7.5	18.5 msec
100.0	8	110.0	11.0	17.0 msec

7. The program may desire to stop transmitting. The Receiver and Scanner flags must be cleared. This is accomplished by reading the character which clears the Receiver flag or by performing step 5 (echo check) and releasing the scanner (same as steps 5, 6, and 7 under receiving).

There are programming techniques which implement operator intervention and control such as: If the echo check feature is utilized in the program, the Teletype operator may signal the program while the computer is typing out. To accomplish this, the operator pushes a random key; this action causes an echo check in the computer program. The program then waits for instructions from the operator, and if none are received, resends the failed character.

The null/idle character in the 8-bit code may be distinguished from the character produced when the break key is pushed by examining the eighth (most significant) bit (1 equals null, 0 equals break if parity is not being generated). A large number of break characters would be interpreted by the program as an open line which may be detected by a series of blank characters in the 5-bit code.

The program must determine by a program switch which state the line is in (transmitting or receiving). To insure that the correct character (null, blank) is read when receiving on an open line, the program must examine the character within the times listed above in step 6.

The programming operations used to accomplish transmitting and receiving characters in a many station Teletype communication system involves knowing when stations require servicing, being able to determine the specific station that requires servicing, and transferring characters between the 630 DCS and the PDP-6.

The method of receiving a character is as follows:

1. The receiver turns on a flag at the completion of receiving a Teletype character.
2. A rotating priority type scanner using a counter is continually sampling the Receiver flags. When a flag is found on, the computer stops. The number in the counter is unique to the receiver requesting service. (The number may range from 00 to 77_8 depending upon the number of stations connected. For discussion purposes, it is suggested that all labeling, identification, and references to stations be in accordance with the following scheme:

There are up to eight stations in a group, numbered 0-7 and assigned the units position of a 2-digit octal number. There are up to eight groups in the scanner system, numbered 0-7 and assigned the eights position of the 2-digit octal number. Thus station 24 is station 4 in group 2.)

3. When the counter stops, a second flag (the Scanner flag) is turned on. It is this flag that notifies the computer that the scanner has found a receiver that requires service. Notification may be in the form of a program interrupt, program test by check status, and/or skip test of a flag.

4. The program requests that the number contained in the counter (station number) be placed into an input register (in-out register, accumulator register, core memory register). In some Type 633 models, the line number may be ORed to the input register at the program's option. (The program may release the scanner at this time. Doing so does not clear the receiver flag. After one complete revolution, the scanner finds the Receiver flag still on and again requests service for the station. Thus, the program may temporarily ignore a station request for service and permit the scanner to search for a station requiring higher priority service.) The program may reset the counter to zero. This in effect gives the lower numbered stations higher priority.

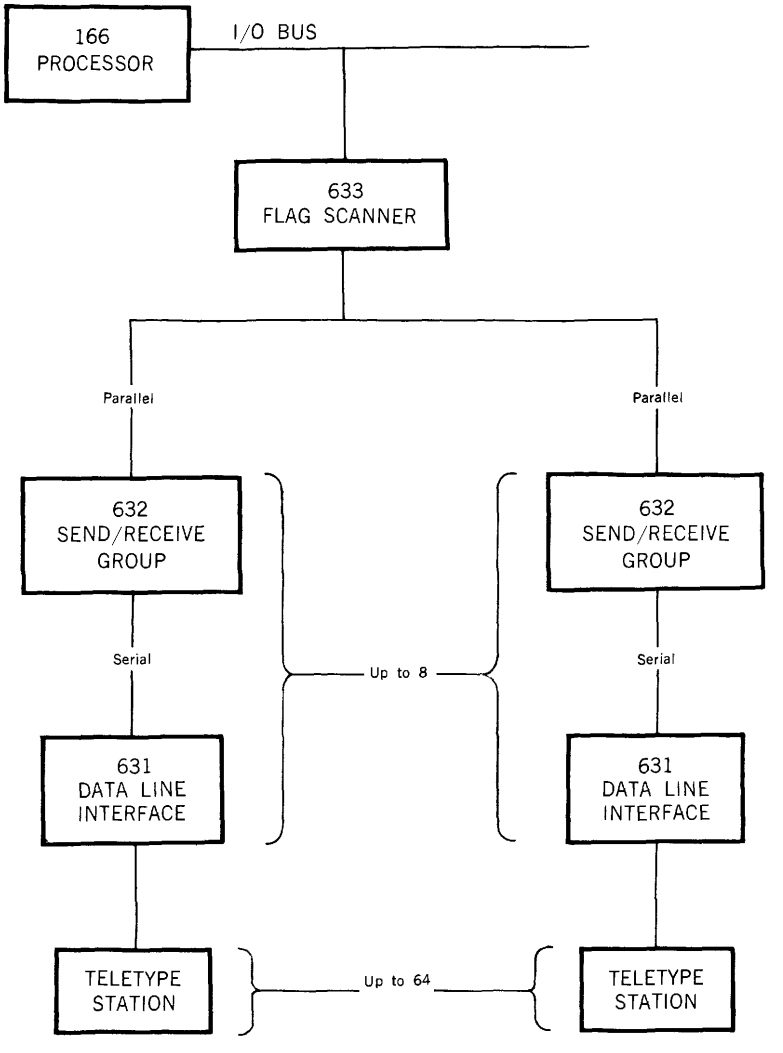


Figure 24 Data Communication System Type 630

5. The program then reads the character from the service requesting receiver into an input register (in-out register, accumulator register, core memory register). In some Type 633 models, the incoming character may be ORed to the input register at the program's option. If the "OR" option is used, that portion of the input register, where the character is received, must be zero prior to read-in. Transmitter garbling may otherwise occur.

6. Reading the character clears the Receiver flag.

7. The Scanner flag should be cleared at this time.

8. To insure that characters are not lost, when receiving at maximum data rates, the program must examine the character within the following times (after service has been requested).

WPM	Bit Code	Baud	Unit Code	Time
60.0	5	45.5	7.5	41.0 msec
66.6	5	50.0	7.5	37.5 msec
100.0	5	75.0	7.5	25.0 msec
100.0	8	110.0	11.0	21.5 msec

If delays greater than above are experienced, the character read in will be garbled; however, the next character received will be correct. The Receiver flag must be turned off (step A6) even if the program exceeds the above limits. Otherwise; the completion of the next character cannot be determined.

APPENDIX 1
INSTRUCTIONS

FULL WORD TRANSMISSION INSTRUCTIONS

Mnemonic	Mode	Code	Effect	Comments
MOVE		200	C(E) => C(AC)	
	I	201	E => C(AC)	
	M	202	C(AC) => C(E)	
	S	203	C(E) => C(E)	
MOVS		204	$C(E)_L$ => $C(AC)_R$ $C(E)_R$ => $C(AC)_L$	
	I	205	0 => $C(AC)_R$ E => $C(AC)_L$	
	M	206	$C(AC)_L$ => $C(E)_R$ $C(AC)_R$ => $C(E)_L$	
	S	207	$C(E)_L$ => $C(E)_R$ $C(E)_R$ => $C(E)_L$	
MOVN		210	−C(E) => C(AC)	
	I	211	−E => C(AC)	The negative is the twos complement
	M	212	−C(AC) => C(E)	
	S	213	−C(E) => C(E)	

MOVM		214	If C(E) ≥ 0, then
			C(E) => C(AC)
			If C(E) < 0, then
			−C(E) => C(AC)
				May set AC OV flag

	I	215	E => C(AC)	

	M	216	If C(AC) ≥ 0, then
			C(AC) => C(E)
			If C(AC) < 0, then
			−C(AC) => C(E)	

	S	217	If C(E) ≥ 0, then
			C(E) => C(E)
			If C(E) < 0, then
			−C(E) => C(E)	

HALF WORD TRANSMISSION INSTRUCTIONS

Mnemonic	Mode	Code	Effect	Comments
HLL		500	$C(E)_L$ => $C(AC)_L$	
	I	501	0 => $C(AC)_L$	
	M	502	$C(AC)_L$ => $C(E)_L$	
	S	503	$C(E)_L$ => $C(E)_L$	
HRR		540	$C(E)_R$ => $C(AC)_R$	
	I	541	E => $C(AC)_R$	
	M	542	$C(AC)_R$ => $C(E)_R$	
	S	543	$C(E)_R$ => $C(E)_R$	

HALF WORD TRANSMISSION INSTRUCTIONS (continued)

Mnemonic	Mode	Code	Effect	Comments
HRL		504	$C(E)_R$ => $C(AC)_L$	
	I	505	E => $C(AC)_L$	
	M	506	$C(AC)_R$ => $C(E)_L$	
	S	507	$C(E)_R$ => $C(E)_L$	
HLR		544	$C(E)_L$ => $C(AC)_R$	
	I	545	0 => $C(AC)_R$	
	M	546	$C(AC)_L$ => $C(E)_R$	
	S	547	$C(E)_L$ => $C(E)_R$	
HLLZ		510	$C(E)_L$ => $C(AC)_L$ 0 => $C(AC)_R$	
	I	511	0 => $C(AC)_L$ 0 => $C(AC)_R$	
	M	512	$C(AC)_L$ => $C(E)_L$ 0 => $C(E)_R$	
	S	513	$C(E)_L$ => $C(E)_L$ 0 => $C(E)_R$	
HRRZ		550	$C(E)_R$ => $C(AC)_R$ 0 => $C(AC)_L$	
	I	551	E => $C(AC)_R$ 0 => $C(AC)_L$	
	M	552	$C(AC)_R$ => $C(E)_R$ 0 => $C(E)_L$	
	S	553	$C(E)_R$ => $C(E)_R$ 0 => $C(E)_L$	
HLLO		520	$C(E)_L$ => $C(AC)_L$ 777777_8 => $C(AC)_R$	
	I	521	0 => $C(AC)_L$ 777777_8 => $C(AC)_R$	
	M	522	$C(AC)_L$ => $C(E)_L$ 777777_8 => $C(E)_R$	
	S	523	$C(E)_L$ => $C(E)_L$ 777777_8 => $C(E)_R$	
HRRO		560	$C(E)_R$ => $C(AC)_R$ 777777_8 => $C(AC)_L$	
	I	561	E => $C(AC)_R$ 777777_8 => $C(AC)_L$	
	M	562	$C(AC)_R$ => $C(E)_R$ 777777_8 => $C(E)_L$	
	S	563	$C(E)_R$ => $C(E)_R$ 777777_8 => $C(E)_L$	
HRLO		524	$C(E)_R$ => $C(AC)_L$ 777777_8 => $C(AC)_R$	
	I	525	E => $C(AC)_L$ 777777_8 => $C(AC)_R$	
	M	526	$C(AC)_R$ => $C(E)_L$ 777777_8 => $C(E)_R$	
	S	527	$C(E)_R$ => $C(E)_L$ 777777_8 => $C(E)_R$	

HALF WORD TRANSMISSION INSTRUCTIONS (continued)

Mnemonic	Mode	Code	Effect	Comments
HLLE		530	$C(E)_L$ => $C(AC)_L$ If $C(E)_0 = 0$, then 0 => $C(AC)_R$ If $C(E)_0 = 1$, then 777777_8 => $C(AC)_R$	
	I	531	0 => C(AC)	
	M	532	$C(AC)_L$ => $C(E)_L$ If $C(AC)_0 = 0$, then 0 => $C(E)_R$ If $C(AC)_0 = 1$, then 777777_8 => $C(E)_R$	
	S	533	$C(E)_L$ => $C(E)_L$ If $C(E)_0 = 0$, then 0 => $C(E)_R$ If $C(E)_0 = 1$, then 777777_8 => $C(E)_R$	
HRRE		570	$C(E)_R$ => $C(AC)_R$ If $C(E)_{18} = 0$, then 0 => $C(AC)_L$ If $C(E)_{18} = 1$, then 777777_8 => $C(AC)_L$	
	I	571	E => $C(AC)_R$ If $Y_{18} = 0$, then 0 => $C(AC)_L$ If $Y_{18} = 1$, then 777777_8 => $C(AC)_L$	
	M	572	$C(AC)_R$ => $C(E)_R$ If $C(AC)_{18} = 0$, then 0 => $C(E)_L$ If $C(AC)_{18} = 1$, then 777777_8 => $C(E)_L$	
	S	573	$C(E)_R$ => $C(E)_R$ If $C(E)_{18} = 0$, then 0 => $C(E)_L$ If $C(E)_{18} = 1$, then 777777_8 => $C(E)_L$	
HRLZ		514	$C(E)_R$ => $C(AC)_L$ 0 => $C(AC)_R$	
	I	515	E => $C(AC)_L$ 0 => $C(AC)_R$	
	M	516	$C(AC)_R$ => $C(E)_L$ 0 => $C(E)_R$	
	S	517	$C(E)_R$ => $C(E)_L$ 0 => $C(E)_R$	
HLRZ		554	$C(E)_L$ => $C(AC)_R$ 0 => $C(AC)_L$	
	I	555	0 => C(AC)	
	M	556	$C(AC)_L$ => $C(E)_R$ 0 => $C(E)_L$	
	S	557	$C(E)_L$ => $C(E)_R$ 0 => $C(E)_L$	

HALF WORD TRANSMISSION INSTRUCTIONS (continued)

Mnemonic	Mode	Code	Effect	Comments
HLRO		564	$C(E)_L$ => $C(AC)_R$ 777777_8 => $C(AC)_L$	
	I	565	0 => $C(AC)_R$ 777777_8 => $C(AC)_L$	
	M	566	$C(AC)_L$ => $C(E)_R$ 777777_8 => $C(E)_L$	
	S	567	$C(E)_L$ => $C(E)_R$ 777777_8 => $C(E)_L$	
HRLE		534	$C(E)_R$ => $C(AC)_L$ If $C(E)_{18}$ = 0, then 0 => $C(AC)_R$ If $C(E)_{18}$ = 1, then 777777_8 => $C(AC)_R$	
	I	535	E => $C(AC)_L$ If Y_{18} = 0, then 0 => $C(AC)_R$ If Y_{18} = 1, then 777777_8 => $C(AC)_R$	
	M	536	$C(AC)_R$ => $C(E)_L$ If $C(AC)_{18}$ = 0, then 0 => $C(E)_R$ If $C(AC)_{18}$ = 1, then 777777_8 => $C(E)_R$	
	S	537	$C(E)_R$ => $C(E)_L$ If $C(E)_{18}$ = 0, then 0 => $C(E)_R$ If $C(E)_{18}$ = 1, then 777777_8 => $C(E)_R$	
HLRE		574	$C(E)_L$ => $C(AC)_R$ If $C(E)_0$ = 0, then 0 => $C(AC)_L$ If $C(E)_0$ = 1, then 777777_8 => $C(AC)_L$	
	I	575	0 => C(AC)	
	M	576	$C(AC)_L$ => $C(E)_R$ If $C(AC)_0$ = 0, then 0 => $C(E)_L$ If $C(AC)_0$ = 1, then 777777_8 => $C(E)_L$	
	S	577	$C(E)_L$ => $C(E)_R$ If $C(E)_0$ = 0, then 0 => $C(E)_L$ If $C(E)_0$ = 1, then 777777_8 => $C(E)_L$	

BYTE MANIPULATION INSTRUCTIONS

Mnemonic	Mode	Code	Effect	Comments
IBP		133	If $C(E)_P \geq C(E)_S$, then $C(E)_P - C(E)_S \Rightarrow C(E)_P$ If $C(E)_P < C(E)_S$, then $36 - C(E)_S \Rightarrow C(E)_P$ and $C(E)_R + 1 \Rightarrow C(E)_R$	
ILDB		134	$C(E)_P - C(E)_S \Rightarrow C(E)_P$ $C(C(E)) \Rightarrow C(AC)_{JR}$	See the effective BIS flag on the increment instruction.
LDB		135	$C(C(E)) \Rightarrow C(AC)_{JR}$	
IDPB		136	$C(E)_P - C(E)_S \Rightarrow C(E)_P$ $C(AC)_{JR} \Rightarrow C(C(E))$	See the effective BIS flag on the increment instruction.
DPB		137	$C(AC)_{JR} \Rightarrow C(C(E))$	

FIXED POINT ARITHMETIC INSTRUCTIONS

Mnemonic	Mode	Code	Effect	Comments
ADD		270	C(E) + C(AC) => C(AC)	
	I	271	E + C(AC) => C(AC)	May set the OV, CRY0, CRY1 flags
	M	272	C(E) + C(AC) => C(E)	
	B	273	C(E) + C(AC) => C(E), C(AC)	
SUB		274	C(AC) − C(E) => C(AC)	
	I	275	C(AC) − E => C(AC)	May set the OV, CRY0, CRY1 flags
	M	276	C(AC) − C(E) => C(E)	
	B	277	C(AC) − C(E) => C(E), C(AC)	
IMUL		220	C(AC) × C(E) => C(AC)	
	I	221	C(AC) × E => C(AC)	
	M	222	C(AC) × C(E) => C(E)	May set OV flag
	B	223	C(AC) × C(E) => C(E),	
IDIV		230	C(AC)/C(E) => C(AC) Remainder => C(AC + 1)	
	I	231	C(AC)/E => C(AC) Remainder => C(AC + 1)	
	M	232	C(AC)/C(E) => C(E)	May set OV flag
	B	233	C(AC)/C(E) => C(E), C(AC), Rem => C(AC + 1)	
MUL		224	C(AC) × C(E) => C(AC) Low order part to C(AC + 1)	
	I	225	C(AC) × E => C(AC) Low order part to C(AC + 1)	May set OV flag
	M	226	C(AC) × C(E) => C(E)	
	B	227	C(AC) × C(E) => C(E), C(AC) Low order part to C(AC + 1)	

FIXED POINT ARITHMETIC (continued)

Mnemonic	Mode	Code	Effect	Comments
DIV		234	C(AC), C(AC + 1)/C(E) => C(AC), Rem => C(AC + 1)	
	I	235	C(AC), C(AC + 1)/E => C(AC), Rem => C(AC + 1)	
	M	236	C(AC), C(AC + 1)/C(E) => C(E)	
	B	237	C(AC), C(AC + 1)/C(E) => C(E), C(AC), Rem => C(AC + 1)	May set OV flag

FLOATING POINT ARITHMETIC INSTRUCTIONS

Mnemonic	Mode	Code	Effect	Comments
FAD		140	C(AC) + C(E) => C(AC)	May set OV flag
	L	141	C(AC) + C(E) => C(AC), C(AC + 1)	Low order bits to AC + 1, Left adjusted with no exponent
	M	142	C(AC) + C(E) => C(E)	
	B	143	C(AC) + C(E) => C(E), C(AC)	
FSB		150	C(AC) − C(E) => C(AC)	May set OV flag
	L	151	C(AC) − C(E) => C(AC), C(AC + 1)	Low order bits to AC + 1, Left adjusted with no exponent
	M	152	C(AC) − C(E) => C(E)	
	B	153	C(AC) − C(E) => C(E), C(AC)	
FMP		160	C(AC) × C(E) => C(AC)	May set OV flag
	L	161	C(AC) × C(E) => C(AC), C(AC + 1)	Low order bits to AC + 1, Left adjusted with no exponent
	M	162	C(AC) × C(E) => C(E)	
	B	163	C(AC) × C(E) => C(E), C(AC)	
FDV		170	C(AC)/C(E) => C(AC)	May set OV flag
	L	171	C(AC)/C(E) => C(AC), C(AC + 1)	Remainder to AC + 1, Left adjusted with no exponent
	M	172	C(AC)/C(E) => C(E)	
	B	173	C(AC)/C(E) => C(E), C(AC)	
FADR		144	C(AC) + C(E) => C(AC)	Same as FAD except for rounding
	L	145	C(AC) + C(E) => C(AC), C(AC + 1)	Low order bits to AC + 1, Left adjusted with no exponent, The results in AC are rounded
	M	146	C(AC) + C(E) => C(E)	
	B	147	C(AC) + C(E) => C(E), C(AC)	
FSBR		154	C(AC) − C(E) => C(AC)	May set OV flag
	L	155	C(AC) − C(E) => C(AC)	Same as FAD except for rounding
	M	156	C(AC) − C(E) => C(E)	
	B	157	C(AC) − C(E) => C(E), C(AC)	Low order bits to AC + 1, Left adjusted with no exponent,

FLOATING POINT ARITHMETIC INSTRUCTIONS (continued)

Mnemonic	Mode	Code	Effect	Comments
FMPR		164	C(AC) × C(E) => C(AC)	Same as FMP except for rounding.
	L	165	C(AC) × C(E) => C(AC), C(AC + 1)	Low order bits to AC + 1, Left adjusted with no exponent.
	M	166	C(AC) × C(E) => C(E)	
	B	167	C(AC) × C(E) => C(E), C(AC)	
FDVR		174	C(AC)/C(E) => C(AC)	May set OV flag
	L	175	C(AC)/C(E) => C(AC), C(AC + 1)	Remainder to AC + 1, Left adjusted with no exponent.
	M	176	C(AC)/C(E) => C(E)	
	B	177	C(AC)/C(E) => C(E), C(AC)	
FSC		132	C(AC) × 2^{E} => C(AC)	E = Effective address. May set OV flag

BOOLEAN INSTRUCTIONS

Mnemonic	Mode	Code	Effect	Comments
CLEAR		400	0 => C(AC)	
	I	401	0 => C(AC)	
	M	402	0 => C(E)	
	B	403	0 => C(E), C(AC)	
AND		404	C(E) ∧ C(AC) => C(AC)	
	I	405	E ∧ C(AC) => C(AC)	
	M	406	C(E) ∧ C(AC) => C(E)	
	B	407	C(E) ∧ C(AC) => C(E), C(AC)	
ANDCA		410	C(E) ∧ C($\overline{AC}$) => C(AC)	
	I	411	E ∧ C($\overline{AC}$) => C(AC)	
	M	412	C(E) ∧ C($\overline{AC}$) => C(E)	
	B	413	C(E) ∧ C($\overline{AC}$) => C(E), C(AC)	
SETM		414	C(E) => C(AC)	
	I	415	E => C(AC)	
	M	416	C(E) => C(E)	
	B	417	C(E) => C(E), C(AC)	
ANDCM		420	C($\overline{E}$) ∧ C(AC) => C(AC)	
	I	421	$\overline{E}$ ∧ C(AC) => C(AC)	
	M	422	C($\overline{E}$) ∧ C(AC) => C(E)	
	B	423	C($\overline{E}$) ∧ C(AC) => C(E), C(AC)	
SETA		424	C(AC) => C(AC)	
	I	425	C(AC) => C(AC)	
	M	426	C(AC) => C(E)	
	B	427	C(AC) => C(E), C(AC)	

BOOLEAN INSTRUCTIONS (continued)

Mnemonic	Mode	Code	Effect	Comments
XOR		430	C(E) $\forall$ C(AC) => C(AC)	
	I	431	E $\forall$ C(AC) => C(AC)	
	M	432	C(E) $\forall$ C(AC) => C(E)	
	B	433	C(E) $\forall$ C(AC) => C(E), C(AC)	
IOR		434	C(E) $\vee$ C(AC) => C(AC)	
	I	435	E $\vee$ C(AC) => C(AC)	
	M	436	C(E) $\vee$ C(AC) => C(E)	
	B	437	C(E) $\vee$ C(AC) => C(E), C(AC)	
ANDCB		440	C($\overline{E}$) $\wedge$ C($\overline{AC}$) => C(AC)	
	I	441	$\overline{E}$ $\wedge$ C($\overline{AC}$) => C(AC)	
	M	442	C($\overline{E}$) $\wedge$ C($\overline{AC}$) => C(E)	
	B	443	C($\overline{E}$) $\wedge$ C($\overline{AC}$) => C(E), C(AC)	
EQV		444	C(E) $\equiv$ C(AC) => C(AC)	
	I	445	E $\equiv$ C(AC) => C(AC)	
	M	446	C(E) $\equiv$ C(AC) => C(E)	
	B	447	C(E) $\equiv$ C(AC) => C(E), C(AC)	
SETCA		450	C($\overline{AC}$) => C(AC)	
	I	451	C($\overline{AC}$) => C(AC)	
	M	452	C($\overline{AC}$) => C(E)	
	B	453	C($\overline{AC}$) => C(AC), C(E)	
ORCA		454	C(E) $\vee$ C($\overline{AC}$) => C(AC)	
	I	455	E $\vee$ C($\overline{AC}$) => C(AC)	
	M	456	C(E) $\vee$ C($\overline{AC}$) => C(E)	
	B	457	C(E) $\vee$ C($\overline{AC}$) => C(E), C(AC)	
SETCM		460	C($\overline{E}$) => C(AC)	
	I	461	$\overline{E}$ => C(AC)	
	M	462	C($\overline{E}$) => C(E)	
	B	463	C($\overline{E}$) => C(E), C(AC)	
ORCM		464	C($\overline{E}$) $\vee$ C(AC) => C(AC)	
	I	465	$\overline{E}$ $\vee$ C(AC) => C(AC)	
	M	466	C($\overline{E}$) $\vee$ C(AC) => C(E)	
	B	467	C($\overline{E}$) $\vee$ C(AC) => C(E), C(AC)	
ORCB		470	C($\overline{E}$) $\vee$ C($\overline{AC}$) => C(AC)	
	I	471	$\overline{E}$ $\vee$ C($\overline{AC}$) => C(AC)	
	M	472	C($\overline{E}$) $\vee$ C($\overline{AC}$) => C(E)	
	B	473	C($\overline{E}$) $\vee$ C($\overline{AC}$) => C(E), C(AC)	
SETO		474	set all bits of AC to 1	
	I	475	same as above	
	M	476	same to C(E)	
	B	477	same to C(E) and C(AC)	

SHIFT INSTRUCTIONS

Mnemonic	Mode	Code	Effect	Comments
ASH		240	$2^E \times C(AC) \Rightarrow C(AC)$	E = Effective address Sets AR OV flag if $C(AR)_0 \forall C(AR)_1 = 1$
ROT		241	$C(AC)_i \Rightarrow C(AC)_{i-E}$ If left, bits moving through $C(AC)_0 \Rightarrow C(AC)_{35}$ If right, bits moving through $C(AC)_{35} \Rightarrow C(AC)_0$	
LSH		242	$C(AC)_i \Rightarrow C(AC)_{i-E}$	
ASHC		244	$2^E \times C(CAC) \Rightarrow C(CAC)$	Sets AR OV flag if $C(AR)_0 \forall C(AR)_1 = 1$
ROTC		245	$C(CAC)_i \Rightarrow C(CAC)_{i-E}$ If left $C(AC)_0 \Rightarrow C(AC+1)_{35}$ If right $C(AC+1)_{35} \Rightarrow C(AC)_0$	
LSHC		246	$C(CAC)_i \Rightarrow C(CAC)_{i-E}$	

MEMORY AND ACCUMULATOR MODIFICATION AND TESTING INSTRUCTIONS

Instructions set the PC Change flag if skip or jump is executed.

Skip instructions test and/or modify the contents of the effective address.
If the condition specified by the mode control is met, the next instruction is skipped.

Jump instructions test and/or modify the accumulator. If conditions specified by the mode control are met, the next instruction is skipped.

Mnemonic	Mode	Code	Effect	Comments
JUMP	—	320	None	For an unconditional jump, this instruction is slower than JRST.
	L	321	If C(AC) < 0, then E => C(PC)	
	E	322	If C(AC) = 0, then E => C(PC)	
	G	327	If C(AC) > 0, then E => C(PC)	
	A	324	E => C(PC)	
	GE	325	If C(AC) $\geq$ 0, then E => C(PC)	
	N	326	If C(AC) $\neq$ 0, then E => C(PC)	
	LE	323	If C(AC) $\leq$ 0, then E => C(PC)	
SKIP	—	330		
	L	331	If C(E) < 0, then skip	
	E	332	If C(E) = 0, then skip	
	G	337	If C(E) > 0, then skip	
	A	334	Always skip	
	GE	335	If C(E) $\geq$ 0, then skip	

MEMORY AND ACCUMULATOR MODIFICATION AND TESTING INSTRUCTIONS (continued)

Mnemonic	Mode	Code	Effect	Comments
	N	336	If C(E) ≠ 0, then skip	
	LE	333	If C(E) ≤ 0, then skip	
AOJ	—	340	C(AC) + 1 => C(AC)	AOJ instructions may set
	L	341	C(AC) + 1 => C(AC) If C(AC) < 0, then E => C(PC)	AC CRY0 and AC CRY1 flag
	E	342	C(AC) + 1 => C(AC) If C(AC) = 0, then E => C(PC)	
	G	347	C(AC) + 1 => C(AC) If C(AC) > 0, then E => C(PC)	
	A	344	C(AC) + 1 => C(AC) E => C(PC)	
	GE	345	C(AC) + 1 => C(AC) If C(AC) ≥ 0, then E => C(PC)	
	N	346	C(AC) + 1 => C(AC) If C(AC) ≠ 0, then E => C(PC)	
	LE	343	C(AC) + 1 => C(AC) If C(AC) ≤ 0, then E => C(PC)	
AOS	—	350	C(E) + 1 => C(E)	AOS instructions may set
	L	351	C(E) + 1 => C(E) If C(E) < 0, then skip	AC CRY0 and AC CRY1 flags
	E	352	C(E) + 1 => C(E) If C(E) = 0, then skip	
	G	357	C(E) + 1 => C(E) If C(E) > 0, then skip	
	A	354	C(E) + 1 => C(E)	
	GE	355	C(E) + 1 => C(E) If C(E) ≥ 0, then skip	
	N	356	C(E) + 1 => C(E) If C(E) ≠ 0, then skip	
	LE	353	C(E) + 1 => C(E) If C(E) ≤ 0, then skip	
SOJ	—	360	C(AC) − 1 => C(AC)	SOJ instructions may set
	L	361	C(AC) − 1 => C(AC) If C(AC) < 0, then E => C(PC)	AC CRY0 and AC CRY1 flags
	E	362	C(AC) − 1 => C(AC) If C(AC) = 0, then E => C(PC)	
	G	367	C(AC) − 1 => C(AC) If C(AC) > 0, then E => C(PC)	
	A	364	C(AC) − 1 => C(AC) E => C(PC)	

MEMORY AND ACCUMULATOR MODIFICATION AND TESTING INSTRUCTIONS (continued)

Mnemonic	Mode	Code	Effect	Comments
	GE	365	C(AC) − 1 => C(AC) If C(AC) ≥ 0, then E => C(PC)	
	N	366	C(AC) − 1 => C(AC) If C(AC) ≠ 0, then E => C(PC)	
	LE	363	C(AC) − 1 => C(AC) If C(AC) ≤ 0, then E => C(PC)	
SOS	—	370	C(E) − 1 => C(E)	SOS instructions may set AC CRY0 and AC CRY1 flags
	L	371	C(E) − 1 => C(E) If C(E) < 0; then skip	
	E	372	C(E) − 1 => C(E) If C(E) = 0, then skip	
	G	377	C(E) − 1 => C(E) If C(E) > 0, then skip	
	A	374	C(E) − 1 => C(E) and skip	
	GE	375	C(E) − 1 => C(E) If C(E) ≥ 0, then skip	
	N	376	C(E) − 1 => C(E) If C(E) ≠ 0, then skip	
	LE	373	C(E) − 1 => C(E) If C(E) ≤ 0, then skip	

ARITHMETIC COMPARE INSTRUCTIONS

Mnemonic	Mode	Code	Effect	Comments
CAM	—	310	Never skip. No effect	
	L	311	Skip if C(AC) < C(E)	
	E	312	Skip if C(AC) = C(E)	
	G	317	Skip if C(AC) > C(E)	
	A	314	Always skip	
	GE	315	Skip if C(AC) ≥ C(E)	
	N	316	Skip if C(AC) ≠ C(E)	
	LE	313	Skip if C(AC) ≤ C(E)	
CAI	—	300	Never skip	
	L	301	Skip if C(AC) < E	
	E	302	Skip if C(AC) = E	
	G	307	Skip if C(AC) > E	
	A	304	Always skip	
	GE	305	Skip if C(AC) ≥ E	
	N	306	Skip if C(AC) ≠ E	
	LE	303	Skip if C(AC) ≤ E	

LOGICAL COMPARE AND MODIFY INSTRUCTIONS

Instruction code format: 110 ww x yy z, where

ww specifies whether to do nothing to the selected bits, to clear the selected bits, to set the selected bits to ones, or complement the selected bits.

x specifies whether the bit selection is done by C(E) or E.

yy specifies no skip, skip on zero, always skip, or skip on not zero.

z specifies whether to use the data word directly or with the left and right halves exchanged.

Instructions set the PC Change flag if skip is executed.

Mnemonic	Mode	Code	Effect	Comments
TDN	—	610	None	
	E	612	If C(E) $\wedge$ C(AC) = 0, then skip.	
	A	614	Always skips.	
	N	616	If C(E) $\wedge$ C(AC) $\neq$ 0, then skip.	
TSN	—	611	None	
	E	613	If $C(E)_S$ $\wedge$ C(AC) = 0, then skip.	
	A	615	Always skips.	
	N	617	If $C(E)_S$ $\wedge$ C(AC) $\neq$ 0, then skip.	
TRN	—	600	None	
	E	602	If E $\wedge$ $C(AC)_R$ = 0, then skip.	
	A	604	Always skips.	
	N	606	If E $\wedge$ $C(AC)_R$ $\neq$ 0, then skip.	
TLN	—	601	None	
	E	603	If E $\wedge$ $C(AC)_L$ = 0, then skip.	
	A	605	Always skips.	
	N	607	If E $\wedge$ $C(AC)_L$ $\neq$ 0, then skip.	
TDZ	—	630	Clear selected bits. Do not skip.	
	E	632	If C(E) $\wedge$ C(AC) = 0, then clear and skip; otherwise, clear and don't skip.	
	A	634	Clear selected bits and skip.	
	N	636	If C(E) $\wedge$ C(AC) $\neq$ 0, then clear and skip; otherwise, clear and don't skip.	
TSZ	—	631	Clear and don't skip.	
	E	633	If $C(E)_S$ $\wedge$ C(AC) = 0, then clear and skip; otherwise, clear and don't skip.	
	A	635	Clear and skip.	
	N	637	If $C(E)_S$ $\wedge$ C(AC) $\neq$ 0, then clear and skip; otherwise, don't skip.	
TRZ	—	620	Clear and don't skip.	
	E	622	If E $\wedge$ $C(AC)_R$ = 0, then clear and skip; otherwise, clear and don't skip.	

LOGICAL COMPARE AND MODIFY INSTRUCTIONS (continued)

Mnemonic	Mode	Code	Effect	Comments
	A	624	Clear and skip.	
	N	626	If E $\wedge$ C(AC)$_R \neq 0$, then clear and skip; otherwise, clear and don't skip.	
TLZ	—	621	Clear and don't skip.	
	E	623	If E $\wedge$ C(AC)$_L = 0$, then clear and skip; otherwise, clear and don't skip.	
	A	625	Clear and skip.	
	N	627	If E $\wedge$ C(AC)$_L \neq 0$, then clear and skip; otherwise, clear and don't skip.	
TDO	—	670	Set and don't skip.	
	E	672	If C(E) $\wedge$ C(AC) = 0, then set and skip; otherwise, set and don't skip.	
	A	674	Set and skip.	
	N	676	If C(E) $\wedge$ C(AC) $\neq$ 0, then set and skip; otherwise, set and don't skip.	
TSO	—	671	Set and don't skip.	
	E	673	If C(E)$_S \wedge$ C(AC) = 0, then set and skip; otherwise, set and don't skip.	
	A	675	Set and skip.	
	N	677	If C(E)$_S \wedge$ C(AC) $\neq$ 0, then set and skip; otherwise, set and don't skip.	
TRO	—	660	Set and don't skip.	
	E	662	If E $\wedge$ C(AC)$_R$ = 0, then set and skip; otherwise, set and don't skip.	
	A	664	Set and skip.	
	N	666	If E $\wedge$ C(AC)$_R \neq 0$, then set and skip; otherwise, set and don't skip.	
TLO	—	661	Set and don't skip.	
	E	663	If E $\wedge$ C(AC)$_L$ = 0, then set and skip; otherwise, set and don't skip.	
	A	665	Set and skip.	
	N	667	If E $\wedge$ C(AC)$_L \neq 0$, then set and skip; otherwise, set and don't skip.	
TDC	—	650	Complement and don't skip.	
	E	652	If C(E) $\wedge$ C(AC) = 0, then complement and skip; otherwise, complement and don't skip.	
	A	654	Complement and skip.	

LOGICAL COMPARE AND MODIFY INSTRUCTIONS (continued)

Mnemonic	Mode	Code	Effect	Comments
	N	656	If C(E) $\wedge$ C(AC) $\neq$ 0, then complement and skip; otherwise, complement and don't skip.	
TSC	—	651	Complement and don't skip.	
	E	653	If $C(E)_S \wedge C(AC) = 0$, then complement and skip; otherwise, complement and don't skip.	
	A	655	Complement and skip.	
	N	657	If $C(E)_S \wedge C(AC) \neq 0$, then complement and skip; otherwise, complement and don't skip.	
TRC	—	640	Complement and don't skip.	
	E	642	If $E \wedge C(AC)_R = 0$, then complement and skip; otherwise, complement and don't skip.	
	A	644	Complement and skip.	
	N	646	If $E \wedge C(AC)_R \neq 0$, then complement and skip; otherwise, complement and don't skip.	
TLC	—	641	Complement and don't skip.	
	E	643	If $E \wedge C(AC)_L = 0$, then complement and skip; otherwise, complement and don't skip.	
	A	645	Complement and skip.	
	N	647	If $E \wedge C(AC)_L \neq 0$, then complement and skip; otherwise, complement and don't skip.	

JUMP AND PUSH DOWN INSTRUCTIONS

Mnemonic	Mode	Code	Effect	Comments
PUSHJ		260	PC => C(C(AC)$_R$ + 1) C(AC) + 1000001_8 => C(AC) E => PC	When stored, program counter contains the location of the instruction +1.
PUSH		261	C(E) => C(C(AC)$_R$ + 1) C(AC) + 1000001_8 => C(AC)	For all push and pop instructions a carry from AR_0 sets the PDL OV flag (see CONO APR, and CONI APR).
POP		262	C(C(AC)$_R$ − 1) => C(E) C(AC) − 1000001_8 => C(AC)	
POPJ		263	C(C(AC)$_R$ − 1) => PC C(AC) − 1000001_8 => C(AC)	
JSR		264	OV, CRY0, CRY1, BIS, PC Change flag and PC => C(E) 0 => BIS flag E + 1 => C(PC)	Flags stored in order shown in bits 0–5. PC stored in bits 13–35. When stored, program counter contains the location of the instruction +1.

JUMP AND PUSH DOWN INSTRUCTIONS (continued)

Mnemonic	Mode	Code	Effect	Comments
JSP		265	PC => C(AC) E => C(PC)	When stored, program counter contains the location of the instruction +1.
JSA		266	C(AC) => C(E) E => $C(AC)_L$ PC => $C(AC)_R$ E + 1 => C(PC)	When stored, program counter contains the location of the instruction +1.
JRA		267	$C(C(AC)_{0-17})$ => C(AC) E => C(PC)	

I/O INSTRUCTIONS

Mnemonic	Mode	Code yy y	Effect	Comments

BLKI		0 0	C(E) + 1000001 => C(E)
			If $C(E)_L \neq 0$, then
			a) if in sequence break mode, then
			(IOT Data) => $C(C(E)_R)$
			and dismiss break.
			b) if not in sequence break mode, then
			(IOT Data) => $C(C(E)_R)$
			and skip.
			If $C(E)_L = 0$, then
			(IOT Data) => $C(C(E)_R)$
			and execute next instruction.	

DATAI		0 4	(IOT Data) => C(E) when ready.	If the device number is 000 000 0, then the instruction is assigned to the processor and reads the C(data switches).

BLKO		1 0	C(E) + 1000001 => C(E)
			If $C(E)_L \neq 0$, then
			a) if in sequence break mode, then
			$C(C(E)_R)$ => (IOT Data)
			and dismiss break.
			b) if not in sequence break mode, then
			$C(C(E)_R)$ => (IOT Data)
			and skip.
			If $C(E)_L = 0$, then
			$C(C(E)_R)$ => (IOT Data)
			and execute next instruction.	

I/O INSTRUCTIONS (continued)

Mnemonic	Mode	Code yy y	Effect	Comments
DATAO		1 4	C(E) => (IOT Data) when ready.	
CONO		2 0	E => device control register	If the device number is 000 000 0, the instruction is assigned to the processor. If the device number is 000 000 1, the instruction is assigned to the processor-PI system.
CONI		2 4	Device control register => C(E) when ready	If the device number is 000 000 0, then the instruction is assigned to the processor. If the device number is 000 000 1, then the instruction is assigned to the processor-PI system.
CONSZ		3 0	If device control register $\wedge$ E = 0, then skip.	The comments for CONI apply to this instruction.
CONSO		3 4	If device control register $\wedge$ E $\neq$ 0, then skip.	The comments for CONI apply to this instruction.

MISCELLANEOUS INSTRUCTIONS

Mnemonic	Mode	Code	Effect	Comments
EXCH		250	C(E) <=> C(AC)	
BLT		251	BLOCK $(C(AC)_L$ => BLOCK $(C(AC)_R)$	
AOBJP		252	C(AC) + 1000001 => C(AC) If C(AC) $\geq$ 0, then jump	This instruction will set the PC Change flag if the jump is executed. It may set the accumulator OV, CRY0, and CRY1 flags.
AOBJN		253	C(AC) + 1000001 => C(AC) If C(AC) < 0, then jump	This instruction will set the PC Change flag if the jump is executed. It may set the accumulator OV, CRY0, and CRY1 flags.

MISCELLANEOUS INSTRUCTIONS (continued)

Mnemonic	Mode	Code	Effect	Comments
JRST		254	Jump and: if bit 9 = 1, then reset or enable the current interrupt priority channel if bit 10 = 1, then halt if bit 11 = 1, then reset OV, CRY0, and CRY1 flags and PC Change flag. bit 12 is unused.	Will set PC Change flag. Bits 9–12 are set by giving an AC number.
JFCL		255	If flags are set, then jump and clear flags.	Flags are selected as follows: Bit 9 = OV Bit 10 = CRY0 Bit 11 = CRY1 Bit 12 = PC Change
XCT		256	Execute instruction at E	

APPENDIX 2
CODES

NUMERICAL

Octal Code	Mnemonic Code	Mode	Page	Octal Code	Mnemonic Code	Mode	Page
132	FSC		31, 91	211	MOVN	I	26, 85
133	IBP		28, 89	212	MOVN	M	26, 85
134	ILDB I		28, 89	213	MOVN	S	26, 85
135	IDPB I		28, 89	214	MOVM		26, 85
136	IDPB I		28, 89	215	MOVM	I	26, 85
137	IDPB I		28, 89	216	MOVM	M	26, 85
140	FAD	L	31, 90	217	MOVM	S	26, 85
141	FAD	M	31, 90	220	IMUL		30, 89
142	FAD	B	31, 90	221	IMUL	I	30, 89
143	FAD		31, 90	222	IMUL	M	30, 89
144	FADR	L	31, 90	223	IMUL	B	30, 89
145	FADR	M	31, 90	224	MUL		29, 89
146	FADR	B	31, 90	225	MUL	I	30, 89
147	FADR		31, 90	226	MUL	M	30, 89
150	FSB	L	31, 90	227	MUL	B	30, 89
151	FSB	M	31, 90	230	IDIV		30, 89
152	FSB	B	31, 90	231	IDIV	I	30, 89
153	FSB		31, 90	232	IDIV	M	30, 89
154	FSBR	L	31, 90	233	IDIV	B	30, 89
155	FSBR	M	31, 90	234	DIV		30, 90
156	FSBR	B	31, 90	235	DIV	I	30, 90
157	FSBR		31, 90	236	DIV	M	30, 90
160	FMP	L	31, 90	237	DIV	B	30, 90
161	FMP	M	31, 90	240	ASH		33, 93
162	FMP	B	31, 90	241	ROT		33, 93
163	FMP		31, 90	242	LSH		33, 93
164	FMPR	L	31, 91	243			
165	FMPR	M	31, 91	244	ASHC		33, 93
166	FMPR	B	31, 91	245	ROTC		33, 93
167	FMPR		31, 91	246	LSHC		33, 93
170	FDV	L	31, 90	247			
171	FDV	M	31, 90	250	EXCH		28, 100
172	FDV	B	31, 90	251	BLT		28, 100
173	FDV		31, 90	252	AOBJP		36, 100
174	FDVR	L	31, 91	253	AOBJN		36, 100
175	FDVR	M	31, 91	254	JRST		36, 100
176	FDVR	B	31, 91	255	JFCL		37, 101
177	FDVR		31, 91	256	XCT		37, 101
200	MOVE	I	26, 85	257			
201	MOVE	M	26, 85	260	PUSHJ		37, 98
202	MOVE	S	26, 85	261	PUSH		37, 98
203	MOVE		26, 85	262	POP		37, 98
204	MOVS	I	26, 85	263	POPJ		37, 98
205	MOVS	M	26, 85	264	JSR		36, 99
206	MOVS	S	26, 85	265	JSP		36, 99
207	MOVS		26, 85	266	JSA		36, 99
210	MOVN		26, 85	267	JRA		36, 99

NUMERICAL (Continued)

Octal Code	Mnemonic Code	Mode	Page
270	ADD		29, 90
271	ADD	I	30, 90
272	ADD	M	30, 90
273	ADD	B	30, 90
274	SUB		29, 90
275	SUB	I	30, 90
276	SUB	M	30, 90
277	SUB	B	30, 90
300	CAI	—	34, 95
301	CAI	L	34, 95
302	CAI	E	34, 95
303	CAI	LE	34, 95
304	CAI	A	34, 95
305	CAI	GE	34, 95
306	CAI	N	34, 95
307	CAI	G	34, 95
310	CAM	—	34, 95
311	CAM	L	34, 95
312	CAM	E	34, 95
313	CAM	LE	34, 95
314	CAM	A	34, 95
315	CAM	GE	34, 95
316	CAM	N	34, 95
317	CAM	G	34, 95
320	JUMP	—	33, 93
321	JUMP	L	33, 93
322	JUMP	E	33, 93
323	JUMP	LE	33, 93
324	JUMP	A	33, 93
325	JUMP	GE	33, 93
326	JUMP	N	33, 93
327	JUMP	G	33, 93
330	SKIP	—	33, 93
331	SKIP	L	33, 93
332	SKIP	E	33, 93
333	SKIP	LE	33, 94
334	SKIP	A	33, 93
335	SKIP	GE	33, 93
336	SKIP	N	33, 94
337	SKIP	G	33, 93
340	AOJ	—	33, 94
341	AOJ	L	33, 94
342	AOJ	E	33, 94
343	AOJ	LE	33, 94
344	AOJ	A	33, 94
345	AOJ	GE	33, 94
346	AOJ	N	33, 94
347	AOJ	G	33, 94
350	AOS	—	33, 94
351	AOS	L	33, 94
352	AOS	E	33, 94
353	AOS	LE	33, 94
354	AOS	A	33, 94
355	AOS	GE	33, 94
356	AOS	N	33, 94
357	AOS	G	33, 94
360	SOJ	—	33, 94
361	SOJ	L	33, 94
362	SOJ	E	33, 94
363	SOJ	LE	33, 95
364	SOJ	A	33, 94
365	SOJ	GE	33, 95
366	SOJ	N	33, 95
367	SOJ	G	33, 94
370	SOS	—	33, 95
371	SOS	L	33, 95
372	SOS	E	33, 95
373	SOS	LE	33, 95
374	SOS	A	33, 95
375	SOS	GE	33, 95
376	SOS	N	33, 95
377	SOS	G	33, 95
400	SETZ		32, 91
401	SETZ	I	32, 91
402	SETZ	M	32, 91
403	SETZ	B	32, 91
404	AND		32, 91
405	AND	I	32, 91
406	AND	M	32, 91
407	AND	B	32, 91
410	ANDCA		32, 91
411	ANDCA	I	32, 91
412	ANDCA	M	32, 91
413	ANDCA	B	32, 91
414	SETM		32, 91
415	SETM	I	32, 91
416	SETM	M	32, 91
417	SETM	B	32, 91
420	ANDCM		32, 91
421	ANDCM	I	32, 91
422	ANDCM	M	32, 91
423	ANDCM	B	32, 91
424	SETA		32, 91
425	SETA	I	32, 91
426	SETA	M	32, 91
427	SETA	B	32, 91
430	OR		32, 92
431	OR	I	32, 92
432	OR	M	32, 92
433	OR	B	32, 92
434	IOR		32, 92
435	IOR	I	32, 92
436	IOR	M	32, 92
437	IOR	B	32, 92

NUMERICAL (Continued)

Octal Code	Mnemonic Code	Mode	Page	Octal Code	Mnemonic Code	Mode	Page
440	ANDCB		32, 92	524	HRLO		27, 86
441	ANDCB	I	32, 92	525	HRLO	I	27, 86
442	ANDCB	M	32, 92	526	HRLO	M	27, 86
443	ANDCB	B	32, 92	527	HRLO	S	27, 86
444	EQV		32, 92	530	HLLE		27, 87
445	EQV	I	32, 92	531	HLLE	I	27, 87
446	EQV	M	32, 92	532	HLLE	M	27, 87
447	EQV	B	32, 92	533	HLLE	S	27, 87
450	SETCA		32, 92	534	HRLE		27, 88
451	SETCA	I	32, 92	535	HRLE	I	27, 88
452	SETCA	M	32, 92	536	HRLE	M	27, 88
453	SETCA	B	32, 92	537	HRLE	S	27, 88
454	ORCA		32, 92	540	HRR		26, 85
455	ORCA	I	32, 92	541	HRR	I	26, 85
456	ORCA	M	32, 92	542	HRR	M	26, 85
457	ORCA	B	32, 92	543	HRR	S	26, 85
460	SETCM		32, 92	544	HLR		26, 86
461	SETCM	I	32, 92	545	HLR	I	26, 86
462	SETCM	M	32, 92	546	HLR	M	26, 86
463	SETCM	B	32, 92	547	HLR	S	26, 86
464	ORCM		32, 92	550	HRRZ		26, 86
465	ORCM	I	32, 92	551	HRRZ	I	26, 86
466	ORCM	M	32, 92	552	HRRZ	M	26, 86
467	ORCM	B	32, 92	553	HRRZ	S	26, 86
470	ORCB		32, 92	554	HLRZ		26, 87
471	ORCB	I	32, 92	555	HLRZ	I	26, 87
472	ORCB	M	32, 92	556	HLRZ	M	26, 87
473	ORCB	B	32, 92	557	HLRZ	S	26, 87
474	SETO		32, 92	560	HRRO		27, 86
475	SETO	I	32, 92	561	HRRO	I	27, 86
476	SETO	M	32, 92	562	HRRO	M	27, 86
477	SETO	B	32, 92	563	HRRO	S	27, 86
500	HLL		26, 85	564	HLRO		27, 88
501	HLL	I	26, 85	565	HLRO	I	27, 88
502	HLL	M	26, 85	566	HLRO	M	27, 88
503	HLL	S	26, 85	567	HLRO	S	27, 88
504	HRL		26, 86	570	HRRE		27, 87
505	HRL	I	26, 86	571	HRRE	I	27, 87
506	HRL	M	26, 86	572	HRRE	M	27, 87
507	HRL	S	26, 86	573	HRRE	S	27, 87
510	HLLZ		26, 86	574	HLRE		27, 88
511	HLLZ	I	26, 86	575	HLRE	I	27, 88
512	HLLZ	M	26, 86	576	HLRE	M	27, 88
513	HLLZ	S	26, 86	577	HLRE	S	27, 88
514	HRLZ		26, 87	600	TRN	—	35, 96
515	HRLZ	I	26, 87	601	TLN	—	35, 96
516	HRLZ	M	26, 87	602	TRN	E	35, 96
517	HRLZ	S	26, 87	603	TLN	E	35, 96
520	HLLO		27, 86	604	TRN	A	35, 96
521	HLLO	I	27, 86	605	TLN	A	35, 96
522	HLLO	M	27, 86	606	TRN	N	35, 96
523	HLLO	S	27, 86	607	TLN	N	35, 96

NUMERICAL (Continued)

Octal Code	Mnemonic Code	Mode	Page	Octal Code	Mnemonic Code	Mode	Page
610	TDN	—	35, 96	650	TDC	—	35, 97
611	TSN	—	35, 96	651	TSC	—	35, 98
612	TDN	E	35, 96	652	TDC	E	35, 97
613	TSN	E	35, 96	653	TSC	E	35, 98
614	TDN	A	35, 96	654	TDC	A	35, 97
615	TSN	A	35, 96	655	TSC	A	35, 98
616	TDN	N	35, 96	656	TDC	N	35, 97
617	TSN	N	35, 96	657	TSC	N	35, 98
620	TRZ	—	35, 96	660	TRO	—	35, 97
621	TLZ	—	35, 97	661	TLO	—	35, 97
622	TRZ	E	35, 96	662	TRO	E	35, 97
623	TLZ	E	35, 97	663	TLO	E	35, 97
624	TRZ	A	35, 96	664	TRO	A	35, 97
625	TLZ	A	35, 97	665	TLO	A	35, 97
626	TRZ	N	35, 96	666	TRO	N	35, 97
627	TLZ	N	35, 97	667	TLO	N	35, 97
630	TDZ	—	35, 96	670	TDO	—	35, 97
631	TSZ	—	35, 96	671	TSO	—	35, 97
632	TDZ	E	35, 96	672	TDO	E	35, 97
633	TSZ	E	35, 96	673	TSO	E	35, 97
634	TDZ	A	35, 96	674	TDO	A	35, 97
635	TSZ	A	35, 96	675	TSO	A	35, 97
636	TDZ	N	35, 96	676	TDO	N	35, 97
637	TSZ	N	35, 96	677	TSO	N	35, 97
640	TRC	—	35, 98	7 00	BLKI		38, 99
641	TLC	—	35, 98	7 04	DATAI		38, 99
642	TRC	E	35, 98	7 10	BLKO		38, 99
643	TLC	E	35, 98	7 14	DATAO		38, 100
644	TRC	A	35, 98	7 20	CONO		38, 100
645	TLC	A	35, 98	7 24	CONI		38, 100
646	TRC	N	35, 98	7 30	CONSZ		38, 100
647	TLC	N	35, 98	7 34	CONSO		38, 100

ALPHABETICAL

Mnemonic Code	Octal Code	Page
ADD	270-273	29, 89
AND	404-407	32, 91
ANDCA	410-413	32, 91
ANDCB	440-443	32, 92
ANDCM	420-423	32, 91
AOBJN	253	36, 100
AOBJP	252	36, 100
AOJ	340-347	33, 94
AOS	350-357	33, 94
ASH	240	33, 93
ASHC	244	33, 93
BLKI	700	38, 99
BLKO	710	38, 99
BLT	251	28, 100
CAI	300-307	34, 95
CAM	310-317	34, 95
CONI	724	38, 100
CONO	720	38, 100
CONSO	734	38, 100
CONSZ	730	38, 100
DATAI	704	38, 99
DATAO	714	38, 100
DIV	234-237	30, 90
DPB	137	28, 89
EQV	444-447	32, 92
EXCH	250	28, 100
FAD	140-143	31, 90
FADR	144-147	31, 90
FDV	170-173	31, 90
FDVR	174-177	31, 91
FMP	160-163	31, 90
FMPR	164-167	31, 91
FSB	150-153	31, 90
FSBR	154-157	31, 90
FSC	132	31, 91
HLL	500-503	26, 85
HLLE	530-533	27, 87
HLLO	520-523	27, 86
HLLZ	510-513	26, 86
HLR	544-547	26, 86
HLRE	574-577	27, 88
HLRO	564-567	27, 88
HLRZ	554-557	26, 87
HRL	504-507	26, 86
HRLE	534-537	27, 88
HRLO	524-527	27, 88
HRLZ	514-517	26, 87
HRR	540-543	26, 85
HRRE	570-573	27, 87
HRRO	560-563	27, 86
HRRZ	550-553	26, 86
IBP	133	28, 89
IDIV	230-233	30, 89
IDPB	136	28, 89
ILDB	134	30, 89
IMUL	220-223	28, 89
IOR	434-437	32, 92
JFCL	255	37, 101
JRA	267	36, 99
JRST	254	36, 100
JSA	266	36, 99
JSP	265	36, 99
JSR	264	36, 99
JUMP	320-327	33, 93
LDB	135	28, 89
LSH	242	33, 93
LSHC	246	33, 93
MOVE	200-203	26, 85
MOVM	214-217	26, 85
MOVN	210-213	26, 85
MOVS	204-207	26, 85
MUL	224-227	29, 89
OR	430-433	32, 92
ORCA	454-457	32, 92
ORCB	470-473	32, 92
ORCM	464-467	32, 92
POP	262	37, 98
POPJ	263	37, 98
PUSH	261	37, 98
PUSHJ	260	37, 98
ROT	241	33, 93
ROTC	245	33, 93
SETA	424-427	32, 91
SETCA	450-453	32, 92
SETCM	460-463	32, 92
SETM	414-417	32, 91
SETO	474-477	32, 92
SETZ	400-403	32, 91
SKIP	330-337	33, 93
SOJ	360-367	33, 94
SOS	370-377	33, 95
SUB	274-277	29, 89
TDC	650, 652, 654, 656	35, 97
TDN	610, 612, 614, 616	35, 96
TDO	670, 672, 674, 676	35, 97
TDZ	630, 632, 634, 636	35, 96
TLC	641, 643, 645, 647	35, 98
TLN	601, 603, 605, 607	35, 96

ALPHABETICAL (continued)

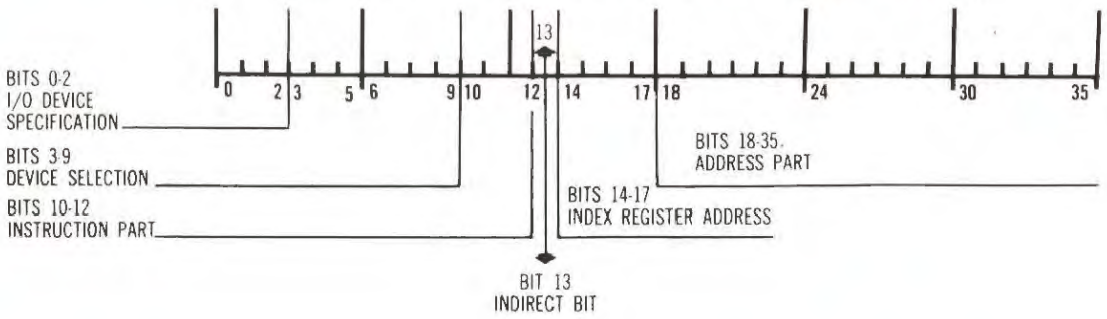
Mnemonic Code	Octal Code	Page	Mnemonic Code	Octal Code	Page
TLO	661, 663, 665, 667	35, 97	TSC	651, 653, 655, 657	35, 98
TLZ	621, 623, 625, 627	35, 97	TSN	611, 613, 615, 617	35, 96
TRC	640, 642, 644, 646	35, 98	TSO	671, 673, 675, 677	35, 97
TRN	600, 602, 604, 606	35, 96	TSZ	631, 633, 635, 637	35, 96
TRO	660, 662, 664, 666	35, 97	XCT	256	37, 101
TRZ	620, 622, 624, 626	35, 96			

APPENDIX 3
GLOSSARY OF ABBREVIATIONS

A — Always
AC — Accumulator (in Memory)
AR — Accumulator Register (in Arithmetic Processor)
B — Both
C — Contents of
CRY0 — Carry 0 flag
CRY1 — Carry 1 flag
E — Effective address
E — Equal to (if used as a mode indicator)
G — Greater than
GE — Greater than or equal to
I — Immediate
i — Any given bit (used as subscript)
JR— Justified right (starting in bit 35)
L — Less than (Skip Mode)
L — Low order part and /or remainder (Transmission Mode)
L — Left half (when used as subscript)
LE — Less than or equal to
M — Memory
N — Not equal to
OV — Overflow flag
P — (used as a subscript only) The P field of the pointer word
PC — Program counter
R — Right half
S — Self
S — Swapped Left and right halves exchanged
S — (used as a subscript only in Byte Manipulation) The S field of the pointer word.
— — Never skip
$\bar{A}$ — Logical complement
A=>B — A replaces B
= — Equivalent
$\not\vee$ — Exclusive OR
V — Inclusive OR
$\wedge$ — AND

INPUT/OUTPUT SUMMARY CHART

INPUT-OUTPUT INSTRUCTION WORD FORMAT



DEVICE / SPEED	OP CODE	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	INPUT BITS / OUTPUT BITS	INPUT INSTRUCTION EFFECT (DATAI, BLKI) / OUTPUT INSTRUCTION EFFECT (DATAO, BLKO)
166 CENTRAL PROCESSOR (CPA)	000	PDL OV FLAG	I/O RESET			MEM-ORY PROTEC-TION FLAG	NON-EX MEM FLAG	CLOCK DISABLE	CLOCK ENABLE	CLOCK FLAG OFF	PC CHG DISABLE	PC CHG ENABLE	PC CHG FLAG	AR OV DISABLE	AR OV ENABLE	AR OV FLAG	BITS 33-35 PRIORITY INTERRUPT ASSIGNMENT			36	
166 PRIORITY INTERRUPT SYSTEM (PRS)	004						PI CLEAR SYSTEM	PI ACTI-VATE SE-LECTED	PI TURN SE-LECTED ON	PI TURN SE-LECTED OFF	PI TURN OFF	PI TURN ON	BITS 29-35 SEVEN PRIORITY INTERRUPT LINES (SELECT CHANNEL 1-7)								
761 PERFORATED TAPE PUNCH (PTP) 63.3 CPS 15.8 MS/CHAR	100													ALPHA/ BINARY 0=α 1=β	BUSY FLAG	DONE FLAG †	BITS 33-35 PRIORITY INTERRUPT ASSIGNMENT			6/8 30-35	CLEAR DONE FLAG SET BUSY FLAG LOAD PUNCH BUFFER
760 PERFORATED TAPE READER (PTR) 400 CPS 2.5 MS/CHAR 15 MS/WORD	104												POWER OFF/ ON 0=OFF 1=ON	ALPHA/ BINARY 0=α 1=β	BUSY FLAG	DONE FLAG †	BITS 33-35 PRIORITY INTERRUPT ASSIGNMENT			8/36	CONTENTS OF READ BUFFER TO PROCESSOR SET BUSY FLAG CLEAR DONE FLAG (GET NEXT CHAR/WORD)
460 CARD PUNCH (CP) 100 CPM	110									CARD ERROR	HOPPER EMPTY	STACKER FULL	NOT READY	EMPTY BIT	BUSY FLAG	DONE FLAG †	BITS 33-35 PRIORITY INTERRUPT ASSIGNMENT			36	LOAD BUFFER, CLEAR CP DONE FLAG
461 CARD READER (CR) 200/800 CPM 2MS PER COL.	114							NOT READY	ERROR	END OF FILE	END OF CARD	DATA MISSED	ALL FLAG	HOLLE-RITH OR BINARY †	BUSY FLAG	DONE FLAG	BITS 33-35 PRIORITY INTERRUPT ASSIGNMENT			6, 12 BITS PER COL-UMN PACKED IN 36	INPUT DATA, CLEAR CR COL DONE FLAG
626 CONSOLE TELEPRINTER (TTY) 10 CPS	120								TTI BUSY OFF	TTI FLAG OFF	TTO BUSY OFF	TTO FLAG OFF	TTI BUSY ON	TTI FLAG ON †	TTO BUSY ON	TTO FLAG ON †	BITS 33-35 PRIORITY INTERRUPT ASSIGNMENT			8	INPUT CHAR CLEAR BUSY FLAG CLEAR INPUT FLAG
																				8	OUTPUT CHAR SET BUSY FLAG CLEAR OUTPUT FLAG
646, 680 LINE PRINTERS (LPT)	124									LINE OVERFLOW ERROR †	ERROR †	BUSY FLAG	DONE FLAG †	BITS 30-32 PRIORITY INTERRUPT ERROR FOR FLAG			BITS 33-35 PRIORITY INTERRUPT ASSIGNMENT			36	
136 DATA CONTROL (DC)	200			BITS 20-22 NUMBER OF CHAR-ACTERS ASSEMBLED —1			DATA MISSED	BUFFER ACCU-MULA-TOR MOVE	DATA ACCU-MULA-TOR RE-QUEST	DATA BUFFER RE-QUEST	IN/ OUT	BITS 28-29 PACKING MODE (6/36 12/18 BIT MODE)		BITS 30-32 DEVICE NUMBER			BITS 33-35 PRIORITY INTERRUPT ASSIGNMENT			36	CONTENTS OF DATA BUFFER B TO PROCESSOR CLEAR DATA BUFFER REQUEST FLAG
																				36	PROC DATA TO DATA BUF B; CLEAR DB RU FLAG PROC DATA TO DATA BUFFER B CLEAR DATA BUF REQ FLAG
346 INCREMENT CRT DISPLAY (DI)	130								LIGHT PEN FLAG †	EDGE FLAG †	STOP FLAG †	DONE FLAG DATA PI †	INITIAL-IZE	BITS 30-32 SPECIAL PRIORITY INTERRUPT ASSIGNMENT			BITS 33-35 PRIORITY INTERRUPT ASSIGNMENT			2x18	COORDINATES TO COMPUTER
									BITS 25-27 SPECIAL PI												
551 MICROTAPE CONTROL (UT)	210		UNIT SELECT ENABLE	ENABLE TAPE END FLAG	ENABLE JOB DONE FLAG	TURN ON/ OFF SE-LECTED UNIT	MOVE FOR-WARD/ RE-VERSE	ENABLE TIME FLAG	BITS 25-26 INITIAL TIME DELAY (0, 20, 150, 300 MS START DELAY)		BITS 27-29 CONTROL FUNCTION			BITS 30-32 UNIT SELECTION			BITS 33-35 PRIORITY INTERRUPT ASSIGNMENT				
551 STATUS REGISTER (CLEARED BY CONO 210)	214								DELAY IN PROG-RESS	REQUEST	ACTIVE	NULL	INCOM-PLETE BLOCK TRANS-FER FLAG	WRITING ON	TIME FLAG	PARITY ERROR	ILLEGAL OP FLAG	TAPE END FLAG	JOB DONE FLAG		
										BITS 28-26 TEST CONDITIONS (STATE OF CONTROL)											
516 MAGNETIC TAPE CONTROL	220	FALSE END OF RECORD	WRITE CLOCK ENABLE	SLICE LEVEL 0/1	PARITY (EVEN/ ODD) 0-BCD/ 1-BINARY	BITS 22-23 DENSITY SELECT 200/556 800/556 BPI		BITS 24, 25-27: FUNCTION 25-27: NO-OP, REWIND, W EOF, WRITE, READ COMP, READ, SPACE FWD, SPACE REV 24: SPECIAL FOR FWD UNLOAD WRITE FULL BLANK TAPE				HOLD SE-LECTED	BITS 29-31 UNIT SELECT (0-7)			MT DISABLE EOR TEST	BITS 33-35 PRIORITY INTERRUPT ASSIGNMENT				
516 STATUS REGISTER (1)	224	ILLEGAL FLAG	136 DATA CON-TROL CON-NECTED	CON-TINUE MOTION SE-LECTED	TRANS-FER NEW COM-MAND TO COM-MAND HOLD BUFFER	TRANS-PORT REWD	TAPE AT END POINT	TAPE AT LOAD POINT	SEE NOTE A BELOW ‡	SEE NOTE B BELOW ‡	END OF FILE FLAG	FILE PRO-TECT RING OUT	TAPE MISSED CHAR	READ COM-PARE ERROR	TAPE PARITY ERROR (LONG)	TAPE PARITY ERROR (LAT)	END OF RECORD FLAG	TAPE TRANS-PORT READY	TAPE CON-TROL FREE		
516 STATUS REGISTER (2)	230	TRANS-PORT UP TO SPEED DELAY IN PROG-RESS	RECORD IN PROG-RESS	END OF RECORD MOTION DELAY IN PROG-RESS	MOTION STOP DELAY IN PROG-RESS	COM-MAND HOLD BUFFER	UNIT SE-LECTED NEW COM-MAND	BITS 24-27 FUNCTION FINAL BUFFER				MOTION DELAY IN PROG-RESS	BITS 29-31 UNIT FINAL BUFFER			FIRST CHAR DE-TECTED	EOR READ DELAY	MISSED CHAR DELAY	EOR WRITE DELAY		
630 DATA COMMUNICATION SYSTEM (DCS) CONTROL REGISTER	300															FLAG	BITS 33-35 PRIORITY INTERRUPT ASSIGNMENT			8	INPUT CHAR (SPECIFIED BY COUNTER), CLEAR UNIT FLAG
																				8	OUTPUT CHAR, CLEAR UNIT FLAG, USE BUF TO SELECT LINE RELEASE COUNTER
630: LINE COUNTER	304																BITS 33-35 PRIORITY INTERRUPT ASSIGNMENT			8	INPUT CHAR (SPECIFIED BY COUNTER), CLEAR UNIT FLAG, RELEASE COUNTER
																				8	OUTPUT CHAR, CLEAR UNIT FLAG, USE BUF TO SELECT LINE RELEASE COUNTER

NOTE:
BITS WHICH CAN'T BE READ OUT
BITS WHICH CAN'T BE READ IN
†(DAGGER) PI (PRIORITY OR PROGRAM BITS INTERRUPT OR CHANNEL)

‡NOTE A: TAPE NEAR END POINT (520 INTERFACE) FIRST OPERATION WRITE (521 OR 522 INTERFACE)
NOTE B: TAPE NEAR LOAD POINT IF 520 INTERFACE B CONTROL USING TRANSPORT (521 INTERFACE) SEE POOL WRITE ECHO NOT OK (522 INTERFACE)

INPUT/OUTPUT SUMMARY CHART — PDP6

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

15016

PRINTED IN U.S.A.

50-8/64

DEC-6-0-EX-SYS-UM-FP-ACT00

MULTIPROGRAMMING SYSTEM MANUAL

PDP-6

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

DEC-6-0-EX-SYS-UM-FP-ACT00

PDP-6 MULTIPROGRAMMING SYSTEM MANUAL

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

PREFACE

This manual describes how to use the Multiprogramming Monitor version 1. The distinguishing feature of this version is that all users' programs together must fit within the limits of core memory and are prevented from interfering with each other by hardware and with each other's I/O by software. Also, DECtape is the principal storage medium for users' files.

As future versions of the monitor are developed, inevitably corresponding changes will be made in existing users' programs, both to take advantage of newly added features and to make corrections where an old feature was changed. Therefore, it is recommended that the monitor- and I/O-interface portion of the user's program be written as a separate program so that it may be updated easily. Every effort will be made to keep such changes to a minimum.

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

ILLUSTRATIONS

TABLES

CHAPTER 1

INTRODUCTION TO THE PDP-6 MULTIPROGRAMMED TIME-SHARING SYSTEM

The Programmed Data Processor-6 time-sharing system provides integrated software and hardware for the simultaneous use of a single system by many users at Teletype consoles. A user may obtain the use of all system facilities and programs from a remote location. Users' programs and data are kept on a separate DECtape for each user to permit permanent storage of results.

A user may start one or more active programs called jobs. Each job is console initiated and runs in an assigned and protected memory area.

BASIC SYSTEM FEATURES

The basic system objective is to provide a simple, modular, time-sharing system.

File Generation - A user may create and edit a text file directly from his time-sharing console. The need for off-line program preparation devices (key punches, paper tape perforators) is greatly reduced.

On-Line Debugging - A user may debug his program on-line through an advanced debugging program (DDT). Corrections can be made and tested instantly. On-line debugging is feasible because a user cannot tie up the entire installation while he stops to decide what to do next.

Common User Service Programs (CUSPS) - Commonly used programs, such as the compiler, the assembler, and file manipulation programs, are available to all users. New programs of general interest are easily made available to all users.

Unattended Batch Processing - The user may call for a special user program called the Batch Control Processor (BCP) which then takes over the supervision of his job in a manner similar to conventional batch processing systems.

Special Console Service - The executive subroutines may be modified so that installations requiring a special service or language may be included.

Real Time Processing - The necessary input/output routines to connect a special device with the programming system can be added. A job could then issue system input/output commands for the special device

in a similar manner as for conventional devices. In some applications, it would be necessary to modify the schedular portion of the monitor so that occurrence of I/O device conditions would cause special service for the user.

Common I/O Service - I/O service routines within the monitor provide a basic set of input/output commands. Programs are easily written which communicate with many I/O devices using identical commands for each device. All I/O subroutines use the hardware priority interrupt system so that the monitor never waits for data transmission.

Common System Service - Monitor routines, at the request of a user program, perform special system functions such as giving the time, date, and job reinitializing.

HARDWARE REQUIREMENTS

The following is the minimum hardware configuration to allow multiprogrammed time sharing.

1-166	Arithmetic Processor with 626 Teleprinter
1-161C (or 163C)	16384-Word Core Memory
1-136	Data Control
1-551	DECtape Control
1-555	Dual DECtape Transport (2 drives)
1-760	Paper Tape Reader (for maintenance programs)

In addition to the above hardware, one Type 162 16-Word Fast Memory and any number of additional Type 161C (or 163C) Core Memories (up to a total of 262,144 words) may be added to the system without requiring any modifications to system software.

Optional Hardware with Software Support

With very minor modifications to system software, the standard Time-Sharing Monitor can handle any configuration that contains the following devices:

1-630	Data Communications System with up to 64 half- or full-duplex 8-level teletypewriters
4-555	Dual DECtape Transports (up to 8 drives)
1-516	Magnetic Tape Control with up to 8 Magnetic Tape Transports Type 50, Type 570, or IBM Type 729 (any model)

1-346	Incremental Display (with optional light pen and character generator)
1-461	Card Reader (200 or 800 cpm)
1-646	Line Printer (300, 600 or 1000 lpm)
1-761	Paper Tape Punch

CHAPTER 2

MULTIPROGRAMMING SYSTEM BLOCK DIAGRAM DISCUSSION

Figure 2-1, the Multiprogramming System Block Diagram, depicts the flow of data to the user consoles and external files, and the software-hardware interaction.

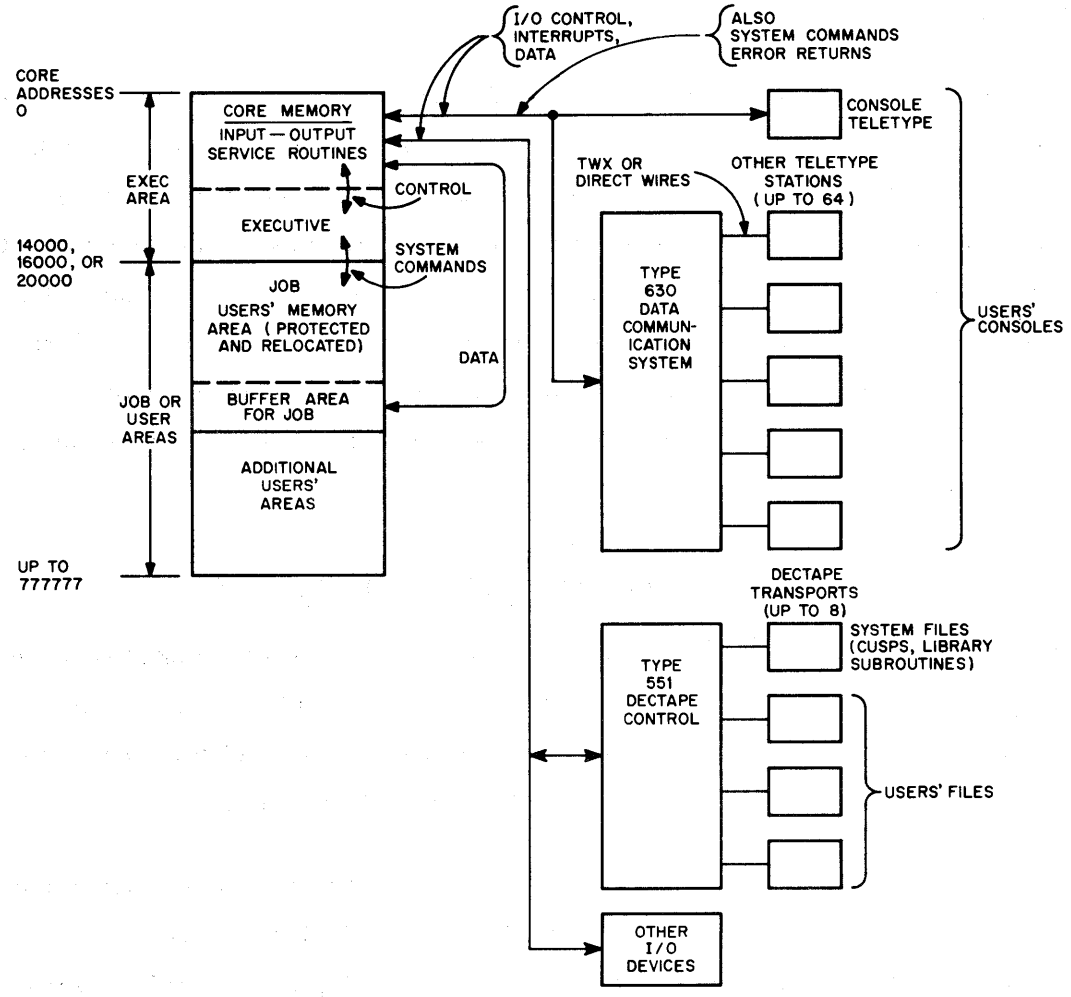


Figure 2-1 Multiprogramming System Block Diagram

USER CONSOLES

As shown in the system block diagram, a user's console is either the console Teletype at the computer or one of 64 Teletypes at (possibly) remote locations connected to the computer via the data communications

system. The console is a user's communication with the multiprogramming system and with his files. All the commands necessary for starting and controlling a job are made via this console; system responses and error messages are printed on the user's console. Also, the console is useful for entering data and receiving results.

Teletype stations connect to the data communications system through standard TWX lines or direct wire. Characters are processed sequentially as they arrive according to one of several schemes as selected by the operating program. Characters are transmitted to the user's program a character at a time, a line at a time, or a full buffer (95 characters) at a time.

A user's console has three states: the detached state, the monitor state, and the user state. The detached state exists whenever the console is idle. If a user types a legitimate monitor command on the detached console, the monitor automatically assigns a free job number to the console and places the console in the monitor state. In this state, the monitor interprets characters typed in as monitor commands. Certain monitor commands start the user's program running and simultaneously place the console in the user state so that characters typed in are processed by the user's program and not by the monitor. Certain error conditions or typing the proper control character (C with the control key held down) cause the monitor to return the console to the monitor state. Certain other monitor commands return the console to the detached state (DETACH and KJOB - kill job).

MULTIPROGRAMMING EXECUTIVE FUNCTIONS

The multiprogramming system consists of routines to allocate resources (core storage, arithmetic processor time, I/O devices) among the various users and routines to perform input/output. The resource-allocation portion is called the exec or the monitor. Also included in the exec are routines to perform such bookkeeping operations as timing the user program or supplying the date on request.

The monitor and I/O routines form a collection of programs that are permanently in core memory during time-shared operation. The monitor makes use of the PDP-6 time-sharing hardware to load several users' programs into core memory simultaneously and to prevent each user's program from interfering with any other. If more than one user requests that his program be run, the monitor switches control to each program in a round-robin fashion. Switching is rapid so that all programs appear to run simultaneously. If a user's program must wait for the completion of an input/output operation, the monitor immediately switches to another user so that no computing time is wasted.

INPUT/OUTPUT FUNCTIONS

The I/O service routines relieve the programmer of the burden of writing a unique program for each different I/O device. One set of input/output commands controls any device; to change devices, the program need change only the device name in the initialize command (see INIT, Chapter 4). The I/O service routines perform the more common format translations.

The I/O service routines take full advantage of the program interrupt system of the PDP-6 to allow overlapped computation with input/output. The hardware features of the PDP-6 permit devices with relatively high data rates (high density magnetic tape, for example) to be operated without a separate data channel and without taking up a large percentage of the memory cycles.

The I/O service routines are organized for ease in adding a service routine for a new or unique device. In addition to the new service routine for controlling the new device, only one of the monitor subprograms, a routine called IOINIT, need be modified to inform the system about the new device. Thus, the monitor for a particular installation consists of a number of executive routines that are the same for all installations; a selected set of standard library I/O subroutines to control only those devices actually present at the installation; and one subroutine, unique to the particular installation, that informs the system of which devices are present.

USER'S CORE STORAGE

Figure 2-2 is a simplified diagram of the job storage area showing the normal usage of a user's core memory allotment. The user requests as much core memory space as he requires. Any number of contiguous 1K (1024-word) blocks may be assigned to the user. The first 140 (octal) locations are the job data area and are reserved by the exec for use when providing service for the user. Some system information and temporary storage for accumulators are found in this area. To protect the system information, the exec keeps a private copy while allowing the user's program to run and restores the job data area upon stopping the user's program.

The user's program, data, debugging programs, arrays, etc., are loaded in the user's area starting at location 140. Although the actual location may be anywhere in memory, the program is stored in core as if the first address actually were 140; reference to the proper addresses is made through the hardware relocation system. The memory protection system prevents a user's program from referencing any address outside its own job area and thus presents disturbing another user's program. In addition, hardware prohibits a user's program from trying to halt the processor or influence the I/O or priority interrupt system.

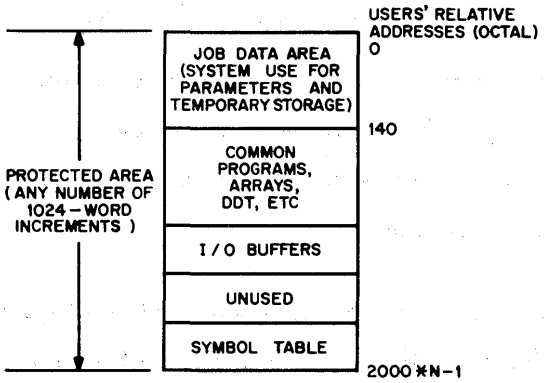


Figure 2-2 Simplified User's Core Storage Map

When a user's program performs input/output, data is transmitted directly to or from buffers in the job's own area. Buffers are created dynamically as required. Space for buffers is taken from the unused memory at the end of the user's program; the address of the first unused location is kept in the job data area.

During program loading, the loader builds a symbol table for linking global symbols and for reference by DDT. If DDT is not being used, the symbol table is just additional unused space at run time. If, however, DDT is being used, the user must write his program (or assign enough extra core space) so that the I/O buffers do not build on top of the symbol table.

USER AND SYSTEM FILES

A user's active files are stored on his own reel of DECtape. However, information may originate from, or be sent to, any other device when desired. To call for a file, the user (or the user's program) must specify a device name, a file name with an optional second name called the extension, and (where ambiguous) a data mode. The device name is a 1- to 6-character mnemonic such as PTP for paper tape punch or DTA5 for DECtape transport number 5. Table 4-1 gives a complete list of device names. The file name is any 1- to 6-character name (consisting of letters and numbers) selected by the user at the time the file was written. The name extension is an additional 1- to 3-character mnemonic describing the information contained in the file; for example, REL for a relocatable binary file, MAC for a Macro language source file, LST for an assembly listing file. Typical data modes are ASCII (A) for ASCII data consisting of 7-bit characters packed five to a word, binary (B), dump (D) for core images, and image (I) for unprocessed data received directly from a device.

System files consisting of copies of each of the active CUSP and library programs are kept on one DECtape reel mounted on device SYS (usually DECtape transport 0). The files on this tape are accessible by all users. A listing of the file names of a standard system-files tape (called the CUSP tape) and directions on how to use each of these files appear in Chapter 6.

CHAPTER 3

TIME-SHARING CONSOLE OPERATION

The user has complete access to all system functions and programs by typing on his time-sharing console. User requests are processed in one of two distinctly different ways. Certain requests, such as starting or stopping a program or assigning I/O devices, are performed by the time-sharing monitor itself. Other functions, such as listing a DECtape file, assembling or compiling source language programs, or loading relocatable binary programs, are performed by library programs that are loaded and run in exactly the same manner as any user's program. These library programs are called CUSPs (Common User Service Programs). This chapter explains how to give the commands accepted by the time-sharing monitor; succeeding chapters explain how to give commands to the various programs in the standard CUSP library.

CONSOLE STATES

From the user's point of view, his time-sharing console is in one of three modes (Figure 3-1): the monitor mode, the user mode, or the detached mode. In the monitor mode, characters typed in are presented to the monitor command processor to be interpreted as a monitor command. In the user mode, characters typed in are processed or ignored at the option of the user program. In the user mode the time-sharing console acts as an ordinary input/output device. A time-sharing console is in a third state (the detached state) if nothing has been typed on it since the monitor was started or if the KJOB (kill job) or DETACH command is typed. The console is automatically placed in the monitor mode as soon as any proper monitor command is typed, in which case, the monitor types back:

JOB N INITIALIZED

where N is a number assigned by the monitor for the purpose of identifying this job.

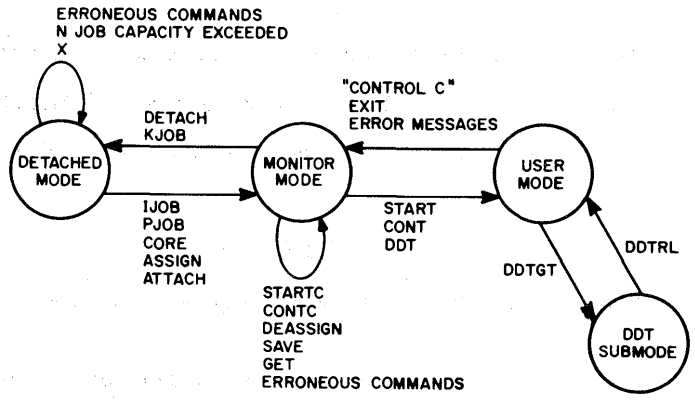


Figure 3-1 Console State Diagram

There are three ways to place the console into the user mode and three ways to return the console to the monitor mode. Typing one of the three monitor commands—START, CONT, and DDT (explained below), places the console in the user mode. Holding down the control key and typing C places the console in the monitor mode and stops the user program if it is running. The other two methods of returning to monitor mode are under program control. The user's program may give the EXIT program command which causes the word EXIT to print on the time-sharing console, the program execution to stop, and the console to enter monitor mode. If the user's program (or the monitor while servicing the user's program) commits a blunder, such as an attempted memory reference outside the user's alloted memory area, the monitor types:

ERROR IN JOB N

where N is the job number assigned by the monitor to the user's program, followed by a 1-line description of the error. After the error printout, the user's program is stopped and the console is placed in monitor mode.

Upon completion of most commands, the monitor types an acknowledgement (usually a carriage-return/line feed) and waits for further commands. If the user types 'control' C before the acknowledgement is typed, the monitor command in progress is stopped before completion.

CONSOLE COMMAND FORMAT

All console commands have the same general format. Command names are strings of from one to six letters and numbers. Characters after the sixth are ignored. Arguments, when required, follow the command name on the same line. Space, and any printing character that is not a letter or numeral, is considered a separator and may be used to separate a command name from its argument(s). Spaces and nonprinting characters preceding the command name are ignored. A semicolon preceding a line causes the line to be disregarded. Other separator characters preceding the line cause the line to be treated as an error. A command line is terminated by carriage-return/line-feed, vertical tab, form-feed, or alt mode. Additional character processing occurs which is the same as for the Teletype ASCII mode described on page 4-18.

If the monitor does not understand the typed-in command, it types back the command up to the place it could not interpret and appends a question mark followed by the words "MONITOR COMMAND ERROR." If the command name is recognized, but a necessary argument for the command is missing, the monitor types back:

TOO FEW ARGUMENTS

If the user types in more arguments than necessary, the extra arguments are ignored.

When a command is typed in correctly, the monitor processes the command without further acknowledgment until either processing is complete, an error occurs in processing the command, or 'control' C is

typed. Most commands are processed instantly. The completion of any command causes an extra carriage-return/line-feed or a line of printing followed by carriage-return/line-feed to be typed out.

In periods of heavy use of the time-sharing system, the job capacity of the monitor may be exceeded. The job capacity is limited by table space within the monitor to a number of jobs equal to the number of time-sharing consoles physically connected to the system. If any console initiates more than one job, it is possible to exceed the job capacity of the system. When the job capacity is exceeded, any command typed in to initialize a new job (implicitly as well as explicitly) causes the monitor to respond with:

JOB CAPACITY EXCEEDED

If this message occurs, the command is not executed and the user must wait until some other user relinquishes his job number (see the explanation of the KJOB command on page 3-9).

When a user detaches his console from his job, a logical console is left attached to the job while the physical console is allowed to attach itself to some other logical console. Therefore, when the number of active jobs is equal to the console-handling capacity of the system, the monitor is unable to find a logical console to which to attach the next physical console that types in. If all logical consoles are in use, a physical console that is not already attached to a logical console cannot communicate with the monitor. To inform the user that the console capacity of the system is exceeded, the monitor types the letter X each time the user types any character other than X. To make this problem as infrequent as possible, the monitor console capacity is one more than the job capacity. Thus, when the job capacity is used up, the next user to type receives the "JOB CAPACITY EXCEEDED" message via the one remaining logical console; but, if yet another user starts typing, he receives the "X" indication. However, if a console receives the "JOB CAPACITY EXCEEDED" message, the monitor breaks the connection to the logical console automatically at the end of the message, thus freeign the logical console for the use of anyone who might be receiving the "X" message.

> NOTE: When the "JOB CAPACITY EXCEEDED" message is received, the only console command that has a useful effect is ATTACH (see the explanation of ATTACH on page 3-10).

Console Commands for Initializing a Job

The following commands are most often typed by the user when he first sits down at the console.

IJOB — Initialize JOB. (Required only if someone else has been using the same console.) This command initializes the job to the following state: no core assigned; no devices except the Teletype assigned; the Teletype given the logical name TTY. The monitor responds with:

JOB N INITIALIZED

PJOB Print JOB number. The monitor types out the number assigned to the job to which the console is currently attached.

ASSIGN ASSIGN I/O devices to this job. This command requires one or two arguments. In the form:

ASSIGN DEV

the physical device DEV is reserved for the exclusive use of this job. (See Table 4-1 for a complete list of devices and their physical names.)

In the form:

ASSIGN DEV NAME

device DEV is reserved, and in addition, NAME (called the logical name) is made synonymous with DEV. If NAME happens to be the same as the physical name of some other device, the logical name takes precedence. With the ability to rename devices, a user may write his programs to use arbitrarily named devices which he assigns to the most convenient physical devices at run time.

For multiple devices such as DECtape or magnetic tape, the physical name consists of a 3-letter device-type mnemonic followed by the unit (transport) number. For example, DTA3 means DECtape (DTA) transport number 3. To locate a free transport, use only the first three characters of the physical device name in place of DEV. For example:

User types:	ASSIGN DTA
Monitor responds:	DEVICE DTA4 ASSIGNED
Then user types:	ASSIGN DTA
Monitor responds:	DEVICE DTA5 ASSIGNED

In the example above, the monitor is asked to find two free DECtape transports. Transports 4 and 5 were free and are now accessible only to the user typing the ASSIGN command.

The monitor may type back one of the three following error messages:

LOGICAL NAME ALREADY IN USE, DEVICE X ASSIGNED	The logical name specified is the same as a logical given in a previous ASSIGN command. Nevertheless, the device is reserved but without a logical name specified.
ALREADY ASSIGNED TO JOB N	Another user is using the device.
NO SUCH DEVICE	Either the physical device name is misspelled or (in the case of searching for a multiple device) all transports are in use.

Examples

User types:	ASSIGN DTA, ABC	
Monitor responds:	DEVICE DTA6 ASSIGNED	(successful)
User then types:	ASSIGN DTA, DEF	(find another transport)
Monitor responds:	NO SUCH DEVICE	(all in use)
User then types:	ASSIGN PTP, ABC	(reserve the paper tape punch)
Monitor responds:	LOGICAL NAME ALREADY IN USE	(punch is reserved but ABC still refers to DTA6)
	DEVICE PTP ASSIGNED	
User then types:	ASSIGN DTA1, DEF	
Monitor responds:	ALREADY ASSIGNED TO JOB2	(another user has reserved DTA1)

See the DEASSIGN console command for relinquishing devices.

Console Commands for Starting a Job

The following three commands must always be typed in order to start a CUSP or user's program. The initialization commands discussed previously need not necessarily be used; but IJOB is desirable to cancel whatever the previous user at the same console had done.

CORE

Assign CORE memory space. The CORE command requires one argument, a decimal integer, specifying the total number of 1024-word blocks of core memory needed to run his program. The core assigned becomes inaccessible to all other users.

If the core assignment is successful, the monitor makes no comment. If there is not a free block of core of the size requested, the monitor responds with:

N FREE 1K BLOCKS LEFT, NONE ASSIGNED

where N is the number of available 1K blocks of core. When this message occurs, the user is left with no core assigned even though he may have had core assigned when he typed the command.

Sometimes the monitor cannot satisfy a core request even though enough core is available to do so. This problem is best illustrated by an example. Suppose that at the start of the day, two users of a 32K PDP-6 system request 9K of core apiece. With a 6K monitor, only 8K of core remains. Then suppose that the user who was assigned the first 9K block relinquishes his core. The memory available for the user's programs then consists of a free 9K block, an in-use 9K block (by the second user), and a free 8K block. A request for 10K of core cannot now be satisfied because the monitor cannot move programs to join two smaller free blocks. It is up to each computer installation to develop ground rules for the use of core. In this particular instance, the remaining user must save his program, request 9K of core

(using a "CORE 9" console command), and restore his program. The monitor assigns the lowest 9K block available, thus making the 17K of unused core into one long block.

To increase his core allotment, the user must retype the CORE command giving the total amount of core required. Similarly, to decrease the amount of core (or to relinquish all core) the CORE command must be retyped. When changing the amount of core assigned, the new core area is always the lowest block of core available that is large enough to fill the request. Thus, the user does not necessarily get back any of the same core with which he started.

If a user interrupts his program (by typing 'control' C) while I/O operations are in progress and types:

CORE 0

all files are closed, end-of-file is written on output magnetic tapes, and all devices not reserved by the ASSIGN console command are returned to the common pool. If instead, the user types:

CORE N

where N is not zero, all devices in current use may be permanently and irretrievably lost to the system. The only recourse is to use one of the error restart sequences described in Appendix 4. A second use of the CORE command which may cause the monitor to fail is typing:

CORE N

while the user program is actually running. (This is possible through the use of STARTC or CONTC described below.)

GET

Load the user's core area with a core image that was previously dumped. This console command accepts two arguments: a device name and a file name. The device and file names specify where to find a previously saved program. (A file name extension of "DMP" is assumed.) This command is used both for loading CUSP programs from the system library and for retrieving previously saved user programs.

Upon completion of loading a program (usually after a moderate delay), the monitor types:

JOB SETUP

The job is fully loaded but not started. Other messages indicate error conditions. These are:

NO CORE ASSIGNED	The user has 0 blocks of core
N 1K BLOCKS OF CORE NEEDED	The user has not assigned enough core to load his program. He must now type CORE N and retype the GET command.

FILE NOT FOUND	The file name typed is not found in the directory for the specified device.
DEVICE NOT AVAILABLE	Another user has control of the specified device.
TRANSMISSION ERROR	A parity or other error condition occurred during data transmission.

START — START the user's program running. This command starts execution of the user's program at a location specified in the job data area. The console is taken out of the monitor mode and placed in the user mode so that succeeding input is directed to the user's program. No check is made to see if a program is actually loaded. START accepts an optional argument, an octal number which, if specified, is taken as the starting address instead of the address found in the job data area. The only error condition is:

NO CORE ASSIGNED

Console Commands for Bookkeeping and Debugging

The following commands are most often used during the running of the user's program for debugging or bookkeeping purposes.

SAVE — SAVE the user's program on a dump file for later restoration using the GET command.

If DDT is loaded, the entire user's core area starting at location 75 (JOBCDP) is dumped, including DDT's symbol table. If DDT is not loaded, only the core up to the contents of JOBFF is dumped.

The SAVE console command requires the same two arguments as the GET console command. The user's program, the starting addresses of the user's program and DDT, and other critical parameters are saved in a dump file on the specified device. The file name is the one specified by the typist, and the file name extension is always "DMP." During operation of the SAVE command, all I/O devices are released and the state of the accumulators and program counter is lost. Therefore, after performing a SAVE operation, the user must restart his program from the beginning.

If the SAVE command is successful, the monitor types out (after a moderate delay):

JOB SAVED

Other messages indicate error conditions. These are:

NO CORE ASSIGNED	There is no core.

N 1K BLOCKS OF CORE NEEDED	The user has assigned less core than is to be dumped according to the contents of JOBFF.
DEVICE NOT AVAILABLE	Some other user is using the specified device.
DIRECTORY FULL	No room is left in the directory to enter the new file name.
TRANSMISSION ERROR	Some error condition, such as a bad spot on the tape, occurred during data transmission.

DDT

Start the user's debugging program, DDT. The starting address of DDT, the user's debugging program, is a parameter kept in the job data area. When the DDT console command is typed, the monitor starts the user's program at the starting address of DDT. The console is taken out of the monitor mode and placed in the user mode so that input is directed to DDT.

Error messages are:

NO CORE ASSIGNED	The user has no blocks of core.
NO DDT	The starting address of DDT, as stored in the job data area, is 0; therefore DDT is not loaded.

CONT

CONTinue execution. When a user's program (including DDT) is stopped by typing 'control' C (C while holding down the 'control' key), execution is resumed by typing the CONT console command. If the user's program stops running because of an error condition or if the user's program deliberately stops by calling "EXIT," the CONT command does not restart the job.

Error messages are:

NO CORE ASSIGNED	
CAN'T CONTINUE	Either the program was never started or it was stopped by other than 'control' C.

Console Commands for Terminating a Job

The following commands are most often used at the end of a computer run when the user is about to leave his console. These commands are also used as a convenient means for relinquishing facilities during a session on the computer.

NOTE: Never abandon a time-sharing console with facilities attached. By so doing, those facilities are lost to other users for the remainder of the day.

DEASSIGN Return I/O devices to the common pool. This command cancels device reservations made via the ASSIGN command. The DEASSIGN command may be typed alone or with one argument. When an argument is typed, it must be the logical or physical name of some device that was previously reserved by the ASSIGN command. If no arguments are typed, all devices currently reserved by user via ASSIGN commands are affected. The DEASSIGN command may be typed even though the user's program continues to use the devices affected.

Error messages are:

NO SUCH DEVICE	The specified device does not exist.
DEVICE WASN'T ASSIGNED	The specified device was not reserved by the ASSIGN command.

KJOB Kill the JOB. This console command does the following:

1. Deassigns and stops all I/O devices connected to the job.
2. Returns all the user's core allotment to the monitor.
3. Places the console in the detached state.
4. Makes the job number assigned to the killed job available to the next user who initializes a job.

Console Commands for Starting More Than One Job

The following commands are useful for starting and monitoring more than one job from one console. These commands may also be used for starting a job and relinquishing the time-sharing console to another user.

STARTC START the user's program as in the START command, but leave the console in the monitor Command mode. The STARTC command differs from the START command in only one respect. STARTC leaves the console in the monitor mode so that further monitor commands may be typed while the user's program runs.

Error messages are the same as for START.

CONTC CONTinue running the user's program with the console in the monitor Command mode. CONTC bears the same relation to CONT that STARTC bears to START. The CONTC command is often used following a sequence such as: a START command; a control statement typed to the user's program in order to initialize the computation; 'control' C. CONTC allows the computation to continue while the user types additional console commands to start other jobs such as I/O conversions or edits.

Error messages are the same as for CONT.

DETACH Place the console in the DETACHed mode. The DETACH command disconnects the user's console from the job, placing the console in the detached mode but not affecting the status of the original job. In the detached state, the user may initialize another job by typing IJOB, PJOB, CORE, or ASSIGN. In combination with the STARTC or CONTC command, the user can initiate two or more jobs.

There are no error conditions or acknowledgements. However, when the user tries to initialize a new job following a DETACH console command, he may discover that the job capacity of the monitor is exceeded. The introduction to this chapter explains the job exceeded condition. See the explanation of ATTACH below for corrective action.

ATTACH Reconnect the console to a previously initiated but now detached job. The ATTACH command allows a user to reconnect to a job from which he has detached. The ATTACH command requires one argument which must be the job number of the job to which the user desires to connect. There is no acknowledgement if the ATTACH operation is successful. Following the ATTACH, the console is always in the monitor mode. If the job happens to be running, typing "CONT" places the console in the user mode without affecting the operation of the job. A user may type ATTACH on any console, not necessarily the one from which he detached. It is not necessary to type DETACH before typing ATTACH in order to switch the console between two or more previously initialized jobs.

If an error condition occurs, the console is left attached to the job to which it was connected before the ATTACH was typed. The error messages are:

TTYn ALREADY ATTACHED	Either the job number typed is erroneous and by coincidence is in active use at some other console, or another user has attached to the job number specified.
JOB NEVER WAS INITIATED	The job number typed belongs to an inactive job, possibly because a user has attached to the job and typed "KJOB."
ILLEGAL JOB NUMBER	The job number typed is not the number of a legitimate job, including 0 and numbers which are in excess of the job capacity of the monitor.

NOTE: Do not ATTACH to any job without the consent of user who originated that job.

CHAPTER 4

PROGRAMMING FOR USERS' PROGRAMS

The user's program area is the portion of core memory available to the user when his program is running. The first address of this area is defined by the contents of the memory relocation register when the user's program runs, and the last address is defined by the contents of the memory protection register. The user's area is shown in Figure 2-1.

In order to make the following discussion clearer, it is necessary to define some terms and review briefly the operation of the PDP-6 time-sharing hardware. The user's program runs while the computer is in a special mode called user mode. In this mode, the contents of the memory relocation register in the arithmetic processor are added to each memory address before the address is sent to the memory system. The address before this addition takes place is called the relative address; after the addition the address is called the absolute address. In the following discussion, all addresses in the user's area are relative addresses.

To take advantage of the fast accumulators, accumulator addresses (0 through 17) are not relocated. Thus, relative locations 0 through 17 cannot be referenced by the user's program. The Time-Sharing Monitor saves the user's accumulators in this area while the system is servicing a programmed operator request. The contents of the memory protection register are compared with the eight high-order bits of each relative address as the user's program runs. If the relative address exceeds the contents of the memory protection register, the memory violation flag is set and control traps to the Time-Sharing Monitor. Some instructions trap to absolute location 40 when executed in the user mode. These instructions include JRST instructions which attempt to halt and all undefined operation codes. The action on traps to location 40 is the same as the action for programmed operators in executive mode, with the addition that the user mode is turned off when the trap occurs. Input/output interrupts also turn off the user mode and trap to the absolute locations normally used in executive mode. All numbers in the following discussion are octal unless otherwise indicated.

SYSTEM PARAMETERS IN JOB AREA

The first 140 registers of the user job area comprise the job data area which is reserved for storing specific information concerning the job, such as the starting address of the user's program and the I/O device assignments made for this program. Also in this area are registers for saving accumulators during requests for service to the executive system or while some other user's program is running. An understanding of the functions of the specifically reserved locations is required for efficient use of the services of the

Time-Sharing Monitor. The monitor makes its own copy of critical registers in the job data while the job is running and restores these locations to the original values whenever the user's program becomes inactive (due to waiting for I/O or running another user's program). These registers are labeled "protected" in the description below. Some of the protected locations can be modified through the use of the CALL UUO described later in this chapter. The mnemonic assignments given below are made on the library file, JOBDAT, as internal symbol definitions. User's programs must be written with job data area mnemonics declared as externals and loaded with file JOBDAT. A listing of JOBDAT and the definitions of system UUO's (unused operation codes) appears in the appendix.

SYSTEM PUSHDOWN LIST

In addition to specific-purpose locations described below, the Time-Sharing Monitor maintains a pushdown list in the user's area for general purpose temporary storage. All the temporary storage required for serving each user is contained within the user's area and thus, the Time-Sharing Monitor can switch to serving another user at any time without forgetting what it was doing for the original user.

JOBAC — The block of core memory from user's location 0 through 17 cannot be referenced by the user's program because the arithmetic processor substitutes absolute registers 0 through 17. The block of 20 registers starting at JOBAC, defined as user's 0, stores the user's accumulators while servicing Executive requests via the programmed operators.

JOBDAC — A second block of 20 registers stores the accumulators while a job is inactive. Since a job may become inactive either while the user's program is computing or while the Executive program is processing a user's request, the information saved may belong to either the user's program or the Executive program.

JOBUUO, JOB41 — Two registers at user's 40 and 41 are used for processing user's programmed operators (001 through 037). The Time-Sharing Monitor reserves programmed operators 040 through 077 for various purposes. The remaining programmed operators (except 000 which is considered illegal) are executed as described in the PDP-6 Programming Handbook using user's locations 40 and 41 instead of absolute 40 and 41. JOBUUO is user's location 40 and JOB41 is location 41.

JOBPC — The location JOBPC saves the program counter and arithmetic processor flags while the job is inactive.

JOBPDP	Protected. The Executive program maintains a pushdown list in the last 16 locations of the user's job data area. JOBPDP contains the pushdown pointer to this pushdown list.
JOBREL	Protected. The left half of JOBREL is zero. The right half contains the highest relative location available to the user, in particular, the contents of the memory protection register.
JOBUXT	Protected. JOBUXT contains the program counter and processor flags for returning to the user's program after the Executive is called by a programmed operator.
JOBLEV	Protected. JOBLEV contains the contents of absolute location 40 when the job is inactive.
JOBSAV	Protected. JOBSAV is a general purpose temporary register for use by the programmed operator processor.
JOBPOV	Protected. JOBPOV contains an address in the user's program to which to jump if a pushdown list overflow trap occurs. If JOBPOV contains 0, no jump occurs, but the monitor stops the user's program and prints an error message on the user's console.
JOBJDA	Protected. A block of 20 registers starting at JOBJDA contains the correlation between input/output devices and the device channel assignments requested by the user's program by means of the INIT prog. op.
JOBDDT	Protected. JOBDDT contains the starting address of DDT, the user's debugging program. When the user types DDT on his console, the monitor finds the starting address of DDT in this location. If JOBDDT contains zero, it is assumed that DDT is not loaded.
JOBSYM	JOBSYM contains a pointer to the symbol table created by the Linking Loader. DDT uses this symbol table for printing and interpreting symbolic addresses. The left half of JOBSYM contains a negative count of the length of the symbol table and the right half contains the lowest register used.

JOBSA — JOBSA contains two addresses. The left half contains the highest register used by the program plus one. The user's input/output buffers are placed in the area immediately following this highest register. The right half contains the starting address of the user's program. Initially, this address is set to the starting address seen by the Loader.

JOBFF — The right half of JOBFF contains the address of the first free location following the user's program. This register is updated by the input/output service routines whenever an input/output buffer is created. The user's program should restore the contents of JOBFF after finishing all use of a buffer in order to recover the core memory used. The left half of JOBFF is not used and normally contains zero.

The remaining locations in the job data area are reserved for future expansion and development. Proposed usage for some of these locations are:

JOBTEM — Two locations beginning with JOBTEM to be reserved for general purpose temporary system storage.

JOBINF — The location JOBINF would contain the job number assigned to the user in bits 0 through 11.

JOBCDP — JOBCDP would contain a byte pointer that points to a command string. The Time-Sharing Executive would accept a command typed on the user's Teletype and determine which system program must be called to process the command. The Executive would then load the system program and set up JOBCDP to point to the command string. The system program then examines the command string, one character at a time, and performs the command operation.

JOBTRP — A block of 20 locations beginning with JOBTRP will contain instructions to be executed when an input/output error or unusual condition occurs. One location is associated with each device channel. If the trap location contains 0, the program is dismissed if an attempt is made to trap on that channel. Otherwise, a JSR instruction is presumed to be in the trap location and control is transferred to the specified trap subroutine as if by a JSR regardless of what instruction is actually at the trap location.

JOBUSY — JOBUSY will contain a pointer to a list of symbols that are defined at the end of the loading process. The user will be able to define all such symbols by means of DDT. DDT will automatically adjust each location where the newly defined symbols are called for.

In addition to the named locations above, all locations between JOBFF and 137 inclusive are reserved for the system pushdown list. The system program loader always starts loading at user's location 140.

USER INPUT/OUTPUT INSTRUCTIONS

The input/output service routines provide the input/output facilities for programs running under the PDP-6 Time-Sharing Monitor System. The object of these routines is to simplify coding for the various input/output devices as much as possible and yet to provide flexibility for performing the most sophisticated functions of input/output. Full advantage is taken of the program interrupt facility of the PDP-6 to overlap in/out operations with computation.

The user's program communicates with the I/O service routines by means of the programmed operators. The terms programmed operator (prog. op.) and unused operation code (UUO) are synonymous. Typical prog. op. functions are to set up a buffer area, to transmit a buffer full of data to an output device, or to release a device for use by other users.

Ring Buffers

Core memory serves as an intermediate buffer between the user's program and the in/out device. An I/O buffer consists of a header block (three words for bookkeeping) and a data storage area. The storage area is subdivided into one or more individual buffers arranged in a ring. To be specific, assume that the device is an input device. Once input operations are under way, the I/O service routines fill a buffer, make the buffer available to the user's program, advance to the next buffer, and begin to fill it if it is free. After filling the last buffer in the storage area, the next buffer to be filled is the first buffer in the area; hence the name ring buffer.

The user's program follows along behind, emptying the next buffer if it is full, or waiting for the next buffer to fill up if it is not yet full.

For output, the user's program and the I/O routines exchange roles, the user filling the buffers and the I/O routines emptying them.

Buffer Structure

The buffer structure consists of two separate segments, the header and the storage area. The header consists of the following three words.

Word 1	The right half contains the address of the second word of the buffer currently in use by the user's program. Bit 0=1 means no input or output has occurred for this ring.
Word 2	A byte pointer to the last item referenced in the buffer.
Word 3	A count of the number of bytes remaining in the buffer.

For most applications, the user's program finds all the information it requires about the state of the buffer area in the header. When referring to a buffer by name, the address of the buffer header, not the storage area, is implied.

The storage area is subdivided into a number of individual, equal-length buffers. The size of each buffer depends on the requirements of the device serviced by the buffer. The first two words of each buffer are reserved for special functions; the first two words are:

Word 1	Unused, but reserved for a block number from fixed address devices such as DECtape or drum.
Word 2	In the right half, the address of the second word of the next buffer in the ring. In the left half, bit 0 is a flag as explained below; bits 1 through 17 contain the buffer size in words (excluding these two words of overhead).

Bit 0 of the second word of the buffer, called the use bit, is a flag that indicates whether the buffer contains active data. This bit is set to 1 by the I/O routines when the buffer is full on input (or being emptied on output) and set to 0 when the buffer is empty on output (or being filled on input). The use bit prevents the I/O routines and the user's program from interfering with each other by attempting to use the same buffer simultaneously.

In all data processing modes, the first data word of the buffer is reserved for a count of the number of data words in the buffer (excluding itself). Thus, a buffer may be only partially filled. The left half of this word is reserved for other bookkeeping purposes depending on the particular device.

Job Initialization

To clear all previous device assignments and prepare the I/O service routines for receiving further initialization commands, execute the instruction:

CALL [SIXBIT /RESET/]

This must be the first instruction in each program. RESET instantly stops any I/O operations that the job has started and, thus, is the only method for releasing a device without waiting for I/O completion. This function is necessary when, for example, the line printer is out of paper and cannot finish printing. RESET also reclaims the buffer area by setting the contents of JOBFF to the address contained in the left half of JOBSA.

Device Initialization

INIT Prog. Op.

A device is initialized and buffer headers specified by executing the command:

```
INIT D, MODE
SIXBIT /device name/
XWD OBUF,IBUF
error return
normal return
```

INIT performs the following functions:

1. The 4-bit channel number D is assigned to the device name appearing in the SIXBIT statement. Henceforth, the I/O service routines interpret the number D as referring to that device. The device name is either a logical or physical name with logical names taking precedence (see "ASSIGN" console command).

2. The data processing mode is selected by mode (see below).

3. The buffer area headers are specified to the device service routines. Each header must be three locations reserved by the user in his program. Only those headers which are to be used need be specified; the output header need not be specified if the program performs only input from a device.

4. The buffer header block is initialized as follows: The first and third words are set to zero. The left half of the second word is set up with an appropriate byte-pointer size field.

INIT Data Modes

One item of information specified in the INIT command is the data processing mode. The following modes are available:

Octal Code	Mnemonic	Meaning
0	A	ASCII, 7-bit characters, packed left justified by the PDP-6 byte instructions.
1	AL	ASCII line, same as A except that the buffer is terminated by a form-feed, vertical tab, line-feed or ALT mode.
10	I	Image, a device dependent mode, the buffer is filled with unprocessed data exactly as supplied by the device.
13	IB	Image binary, 36-bit words. This mode is similar to binary mode except that no automatic formatting or checksumming is done by the service routines.
14	B	Binary, a blocked format consisting of a word count (the first word of the buffer) followed by N 36-bit data words. Checksumming is done automatically by the service routines or by the device itself. The word-count N is in the right half of first word of each block, and if a checksum is computed (cards, paper tape), it is in bits 6-17. This 12-bit folded checksum is computed by summing the 36-bit data words in the block (2's complement add), followed by a 1's complement sum of bits 0-11, 12-23, 24-35.
16	DR	Dump respecting record boundaries. Data is transmitted between any contiguous block of core and one record on the I/O device. In dump mode, ring buffers are not used. The control for the size of the data block and the location in core memory is from a command list explained below.
17	D	Dump, same as DR except that record boundaries are ignored on input and arranging the data into standard-length records is automatic on output.

If the INIT prog. op. specifies a mode that is not appropriate for the given device, the user's job is stopped and the following error message is typed on the user's console:

ILL DEVICE DATA MODE FOR DEVICE X AT USER LOC N

where X is the physical name of the device and loc N is the address of the INIT (or SETSTS described below) prog. op.

INIT Control Modes

In addition to the four bits to specify one of the data modes described above, some of the remaining address bits of the INIT prog. op. specify special control functions. Because most of these bits specify device-dependent processing and do not perform the same function for all devices, only the bits that are defined for all devices are described below. The other bits are explained in the separate discussions for each device.

Bit	Name	Action
31	IOWC	I/O word count. If a 1, forces the service routines to accept the word count in the first data word of the buffer (as computed by the user) instead of computing a word count from the contents of the byte pointer in the buffer header (output only).
30	IOCON	I/O discontinuous mode. If a 1, specifies stopping the device after each buffer is filled (input only).

A device may be initialized separately on two device channels if one INIT prog. op. specifies the output buffer header and the other specifies the input buffer header. All further prog. ops. may then refer to either channel number; the OUTPUT prog. op. can use the channel that was initialized for input, for example.

If any of the I/O prog. ops. that require a device channel specification are given (except INIT, CLOSE, and RELEAS as explained below) before a device is actually initialized on that channel (by the INIT prog. op.), the monitor stops the user's program and types the following error message on the user's console:

I/O TO UNASSIGNED CHANNEL AT USER LOC N

where N is the address in the user's area of the offending prog. op.

Buffer Initialization

INBUF Prog. Op.

To set up a ring buffer for input data, a program executes the prog. op.:

INBUF D,N

where N specifies an N-buffer ring and D specifies a device channel number (the INIT prog. op. must be used previously).

The storage space for the buffer ring is taken from the free storage following the program. The location JOBFF (a permanent global symbol defined on the system symbol file, JOBDAT) contains the address of the first free location following the program. The contents of JOBFF are updated after each INBUF prog. op. If the user wishes to abandon a buffer for one device and set up a buffer for another device, the program should restore the contents of JOBFF to the original value. Then the I/O service routines can reuse the old buffer area in core memory for setting up the new buffer. The RESET operation explained above also resets JOBFF to its initial value calculated when the program is loaded.

OUTBUF Prog. Op.

An ouput buffer area is set up by:

OUTBUF D,N

If no buffer area is set up when the first input or output command is issued, a two-buffer ring is set up automatically from the free storage following the program.

File Initialization

LOOKUP and ENTER Prog. Ops.

These programmed operators initialize a file of data. An ENTER must be given before output, and LOOKUP before input. These prog. ops. have no effect for single-file devices such as paper tape or Teletype input/output. With a multiple-file device, DECtape in particular, these programmed operators specify a file name extension for the file. The format for LOOKUP and ENTER is:

LOOKUP D,E	ENTER D,E
error return	error return
OK return	OK return

D is the device number assigned to the DECtape. E is the first address of a 4-word directory entry block that must be reserved by the user. For both LOOKUP and ENTER, the user's program must put the desired file name in sixbit code into E, the first word of the directory block. Likewise, the file name extension must be put into the left half of the word at E+1. For ENTER only, the date may be placed in the rightmost 12 bits of E+2.

The monitor action on ENTER is:

1. The monitor searches the directory of the specified device for a matching file name and file name extension. If no extension is specified (the extension bits are zero), the file name extensions are excluded from the directory search.

2. If a matching entry is found, the new entry replaces the old. Otherwise, the new entry is appended to the directory.

3. File retrieval information is added to the directory entry. The date, if not already specified, is inserted in the third word of the directory entry.

4. The I/O service routines are initialized to begin outputting at the location specified by the directory entry.

5. If the directory is full and there is no room for appending the new entry, the monitor takes the error return. If the entry is successful, the monitor takes the OK return.

The monitor action on LOOKUP is:

1. The monitor searches the directory as for ENTER.

2. If no matching entry is found, the monitor takes the error return.

3. The complete four-word directory entry is copied from the device directory into the four-word block at E.

4. The file retrieval information is given to the I/O service so that inputting may begin at the beginning of the file.

5. If the LOOKUP is successful, the monitor takes the OK return.

Data Transmission

The two prog. ops., INPUT and OUTPUT, are used to synchronize I/O operations with the user's program. The format for INPUT and OUTPUT is:

INPUT D,0 OUTPUT D,ADR

D is the device number assigned to the desired device. ADR, optional and usually zero, is explained below.

The functions of these two prog. ops. are similar, one serving for input operations and the other for output operations. The commands operate as follows:

INPUT Prog. Op. (Non-Dump Mode)

This prog. op. releases the current buffer to the I/O routines for refilling and advances the buffer header pointers to the next buffer. If the next buffer is not full when the input prop. op. is given, the I/O routines delay until the buffer is full or an end-of-file indication is received. Since the input device normally continues until all buffers are full, the input prog. op. serves to synchronize the user and the input device.

OUTPUT Prog. Op. (Non-Dump Mode)

This prog. op. computes a word count from the position of the byte pointer in the buffer header, stores the word count in the right half of the first data word in the buffer, and makes the buffer available to the

device for outputting. The buffer header pointers are advanced to the next buffer, and the item count is set to the maximum number of bytes that can fit in the buffer. The service routines assure that the new buffer is empty by waiting until it is output, if necessary, and then clear the buffer to zero.

If ADR is nonzero, it is taken as a pointer to the word of the next buffer. The sequence of buffers in the ring thus can be altered. The buffer pointed to by ADR can be in an entirely separate ring from the previous buffer. Once a new buffer position is established, the following buffers are taken from the ring starting at ADR. When using this feature in output operations, the I/O routines compute an improper word count unless one of the following precautions is taken:

1. In the INIT prog. op. have bit 31 a 1 to prevent the service routines from computing a word count (useful if the user's program computes its own word count and places it in the right half of the first data word of the buffer or if outputting directly from an input buffer).

2. Set the byte pointer of the buffer header to point to the last item in the new buffer.

Examples

When the INIT command is given, the item count in each buffer header is set to zero and the byte size is set according to the data processing mode. The INBUF and OUTBUF commands set the first word of the buffer header to point to the first buffer in the ring. The remaining header words are set up when the first input or output command is given. If the item count is ignored by the user's program during output operations, the output prog. op. must be given once following the INIT prog. op. to set up the output buffer header. This one appearance of OUTPUT does not cause data transmission to the device. If the user's output subroutine checks the item count before putting data into the output buffer, this initial output command happens automatically.

A typical Get One Character subroutine from an input device is:

```
GET:    0
        SOSG IBUF+2          ;DECREMENT ITEM COUNT AND TEST
        INPUT D,             ;IF NO DATA, CALL INPUT
        ILDB AC,IBUF+1       ;GET NEXT CHARACTER IN AC
        JRST @GET            ;EXIT
```

A typical Output One Character Subroutine would be:

```
PUT:    0
        SOSG OBUF+2          ;DECREMENT ITEM COUNT AND TEST
        OUTPUT D,            ;IF NO ROOM LEFT CALL OUTPUT
        IDPB AC,OBUF+1       ;DEPOSIT CHARACTER FROM AC
        JRST @PUT
```

INPUT and OUTPUT Prog. Ops. (Dump Mode)

When using dump mode (either 16 or 17, D or DR), the ring buffer scheme is entirely eliminated. Instead, input and output are done directly from the user area. Control of I/O operations is via a command list. The location of the command list is given by the INPUT or OUTPUT prog. op. in the format:

INPUT D,ADR OUTPUT D,ADR

D is the device number assigned to the desired device.

Command List Format

The command list format varies for the specific device but is in general:

```
IOWD A,B        ;TRANSFER A WORDS STARTING AT CORE LOCATION B.
XWD 0, A        ;GO TO A FOR THE NEXT COMMAND
0               ;TERMINATE THE COMMAND LIST
```

The command list is assumed to be in sequential locations except when a GOTO word is encountered. The input/output service routines delay until the command list is completely processed before returning control to the user's program.

Status Checking and Setting Prog. Ops.

Certain errors, such as parity or checksum errors, can occur during data processing. For each 4-bit device number, a status word is provided which contains error indication bits and other useful conditions. The status word is examined or tested by the commands:

GETSTS D,ADR or STATZ D, MASK or STATO D,MASK

These three prog. ops. are exactly analogous to the PDP-6 instructions CONI, CONSZ, and CONSO except that D is the 4-bit device channel number.

Status Bits

The bits of the status word have the following meanings:

Bit	Meaning
18	IOIMPM, Improper Mode. The selected mode is unobtainable.
19	IODERR, Device Error. The device's self-checking circuits indicate an error (parity failure,etc).
20	IODTER, Data Error. The computed checksum failed or invalid data was received.

Bit	Meaning
21	IOBKTL, Block Too Large. A block of data from a high-speed device (DECtape, mag tape, drum) was too large to fit in one buffer.
22	IODEND, Data End Encountered. End-of-file.
23	IOACT, Device Active and Currently Transmitting Data.

Bits 26 through 35 read back the mode transmitted to the I/O service routines by the most recent INIT or SETSTS (see below) prog. op. Bits 24 and 25 give device-dependent conditions that are explained below for each particular device.

At times it is necessary to change certain bits in the status word; for example, clearing error bits or the end-of-file bit, or changing data modes during I/O operations. The prog. op.:

SETSTS D,BITS

delays until the device on device channel D stops transmitting data, then replaces the status word with BITS. All bits of the status word are affected by this prog. op. except bit 23 (IOACT). If the data mode is changed by this prog. op., the byte pointers in the buffer headers are changed appropriately. Note that there is only one status word per device even though a device is initialized on more than one device channel.

Terminating a File

To terminate a file use the CLOSE prog. op. The format is:

CLOSE D,

CLOSE finishes output operations on device channel D by dumping the last buffer if it is partially full and by writing an end character if the device requires one. For input operations, all buffers are emptied by setting the use bits to zero and the item count in the buffer header to zero. For either input or output, the device is returned to the same state as before the first input or output command.

CLOSE does not necessarily terminate the input file and the output file if the device is 2-way. If the OUTPUT prog. op. was used on channel D, the output file is terminated when CLOSE is used on channel D. Similarly, the input file is terminated only if the INPUT prog. op. was used. Also, the user's program may deliberately inhibit the termination of either the input or output file by using CLOSE with the following format:

CLOSE D,N

If N is equal to 1, the output file is not terminated regardless of whether an OUTPUT prog. op. was used. If N is equal to 2, the input file is not terminated. If N is equal to 3, neither file is terminated and the only action is a delay until the device becomes inactive.

Relinquishing a Device

When all I/O to a device is finished, the user's program must give the prog. op.:

```
RELEASE D,
```

RELEASE performs a CLOSE operation unless CLOSE was used previously. Then the device on channel D is made available for other users except when the device is reserved by the ASSIGN console command. A new INIT prog. op. must be given if further operations with the device are required.

If a device is initialized on one channel for input and another for output, only the appropriate channel is affected by RELEASE.

I/O Synchronization

In some instances it is desirable to delay until a device has completed its input/output activities. For example, a program that backspaces magnetic tape might follow up each backspace with a check for the load-point status bit. Because the backspace prog. op. returns to the user before the operation is complete, the user's program must somehow delay in order to check for the load-point status at the end of the backspace operation rather than during the middle. The following prog. op. provides the delay function:

```
CALL D, [SIXBIT /WAIT/]
```

This prog. op. does not return control to the user's program until the device on channel D becomes inactive.

Device Characteristics Checking

The following prog. op. allows a user's program to determine a great deal of information about a device both before and after initializing the device. The sequence:

```
MOVE AC, [SIXBIT /NAME/]
CALL AC, [SIXBIT /DEVCHR/]
```

loads into accumulator AC a word describing the characteristics of the device having the physical or logical name NAME. Bits in the device characteristic word are interpreted as follows:

Bit	Meaning
22 through 35	Data modes are permitted for this device. Mode J is permitted if bit 35-J is a 1.
21	If 1, this device is reserved for some job by an INIT prog. op.
20	If 1, this device is reserved for some job by the ASSIGN console command.
17	If 1, this is an output device.
16	If 1, this is an input device.
15	If 1, this device has a directory (DECtape is currently the only such device). If 0, the prog. ops. ENTER and LOOKUP act as SKIPA instructions.
14	If 1, this device is a Teletype.
13	If 1, this device is magnetic tape. If 0, the MTAPE prog. op. acts as a no-op.
12	If 1, the device is accessible to this job by an INIT prog. op.. However, the device may become unavailable between the DEVCHR and the INIT prog. op. unless bit 21 or 20 is already 1.
6 through 11	Image mode byte size; 0 if image mode is not permitted.
5	If 1, the device is a Teletype that is in use (someone has typed on it). If 0, the device either is not a Teletype or is in the detached mode.
4	If 1, this device is the user's console Teletype. If 0, this device is not the user's console Teletype. This bit is set to 0 when the user types the DETACH command.
3	If 1, this device is the line printer which accepts special carriage control characters as described in the PDP-6 Handbook, F-65.
2	If 1, this device is a card reader for which each line is filled out to 80 characters with spaces.
1	Unused.
0	If 1, the device is a DECtape for which the directory is in core.

Bits 0, 4, and 5 are intended for the use of the monitor rather than for providing useful indications for the user.

If the user's program requests the characteristic word for a nonexistent device name, the word returned is zero.

DEVICE DEPENDENT FUNCTIONS

The paragraphs below explain the unique features of each I/O device. All devices accept the prog. ops. explained above unless otherwise specified. Buffer sizes below are octal and include the two bookkeeping words. Table 4-1 is a summary of the characteristics of all devices.

TABLE 4-1 DEVICE SUMMARY

Physical Name	Name	Hardware Type Number	Prog. Op.	Data Modes	Buffer Size (octal)
CTY	Console Teletype	626	INPUT OUTPUT	A, AL	23
TTY1, TTY2, ... , TTY77	Teletype	630	INPUT OUTPUT	A, AL	23
PTR	Paper Tape Reader	760	INPUT	A, AL, IB, B, I	43
PTP	Paper Tape Punch	761	OUTPUT	A, AL, IB, B, I	43
LPT	Line Printer	646	OUTPUT	A, AL	34
CDR	Card Reader	461	INPUT	A, AL, B, I	36
DTA1, DTA2, ... , DTA0,	DECtape	551/555	INPUT OUTPUT LOOKUP ENTER USETO USETI UGETF CALL [SIXBIT /UTPCLR/]	A, AL, IB, B, I, DR, D	202
MTA0, MTA1, ... , MTA7	Magnetic Tape	516	INPUT OUTPUT MTAPE	A, AL, IB, B, I, DR, D	203

Teletype

Device Name - TTY0, TTY1, ..., TTY76, TTY77, CTY

Line number N of the Type 630 Data Communications System is referred to as TTYN. The console Teletype is CTY. The Time-Sharing Monitor automatically gives the logical name, TTY, to the user's console whenever a job is initialized.

Teletype device names are assigned dynamically. An idle line is considered nonexistent. For interconsole communication, it is necessary for one of the two users to type "DEASSIGN TTY" in order to make his Teletype available to the other user's program as an output or input device. Typing "ASSIGN TTYN" or "IJOB" is the only way to reassign a Teletype that has been deassigned.

Buffer Size - 23

Data Modes A. ASCII - All characters typed in appear in the input buffer as typed with the following exceptions:

RUBOUT	erases the previous character. Successive rubouts erase characters to the left until the beginning of the current bufferful. For each character erased, a backslash is typed. If there is no character to erase, a carriage-return/line-feed is typed.
RETURN	is followed automatically with line-feed, both of which appear in the input buffer.
'control' U	types back as ↑U followed by carriage-return/line-feed. This character deletes the entire current bufferful of ASCII characters.
'control' O	similar to 'control' U but has special action during output.
'control' P	does not appear in the input buffer. This character informs the I/O service routine that there is a Type 35 Teletype on the line. The difference between the two Teletype models is that the Type 35 has a tab, form-feed, and vertical tab mechanism that the Type 33 does not have. Alternate uses of 'control' P turn the special "Type 35" mode on and off.
'control' Z	types as ↑Z and appears in the buffer as 032. This character serves as an end-of-file.

TAB	echoes as an appropriate number of spaces to place tab "stops" at every eighth space unless 'control' P is typed.
VT	(vertical tab) types as four line-feeds.
FORM	(form-feed) types as eight line-feeds.
'control' C	types as ↑C followed by carriage-return/line-feed. This character places the console in the monitor mode, ready to accept monitor commands.

On output all characters are typed just as they appear in the output buffer with the exceptions, TAB, VT, and FORM, which are processed the same as on typein.

If, during output operations, an echo-check failure occurs (the transmitted character was not the same as the intended character), the I/O routine suspends output until the user types the next character. If that character is 'control' C, the console is placed in monitor mode. If it is 'control' O, all Teletype output buffers that currently are full are ignored, thus cutting the output short. All other characters cause the service routines to continue output. The user may cause a deliberate echo check by typing in while typeout is in progress. For example, to return to monitor control mode while typeout is in progress, the user must type any character ("X", for example) until an echo check occurs and output is suspended; then and only then he types 'control' C.

AL. ASCII Line - All input character processing is the same as for the ASCII mode. However, the buffer is terminated on RETURN (carriage-return), LINE FEED, FORM, VT, "control Z", and ALT (alt mode, (175) sometimes labeled ESC, escape). One of these characters always appears at the end of each buffer-ful. RETURN is always followed by LINE FEED so that RETURN is never the last character in the buffer.

On output, AL mode is no different from the A mode.

DDT Submode

In order to allow a user's program and the DDT debugging program to both use the console Teletype without interfering with one another, the Teletype service routine provides the DDT submode. This mode is entered by the prog. op.:

```
CALL [SIXBIT/DDTGT/]
```

and left by the prog. op.:

```
CALL [SIXBIT/DDTRL/]
```

and does not affect the Teletype status if it is initialized with the INIT prog. op. It is not necessary to use the INIT prog. op. in order to do I/O in the DDT submode. I/O in DDT mode is always to the user's console and not to any other device.

In the DDT submode, the user's program is responsible for its own buffering. Input is usually one character at a time, but if the typist types characters faster than they are processed, the Teletype service routine supplies a bufferful of characters at a time.

To input characters in DDT mode, use the sequence:

```
MOVEI AC,BUF
CALL AC, [SIXBIT /DDTIN/]
```

BUF is the first address of a 21-word block in the user's area. This prog. op. delays if necessary, until one character is typed in. Then all characters (in 7-bit packed format) typed in since the previous occurrence of DDTIN are moved to the user's area in locations BUF, BUF+1, etc. The character string is always terminated by a null character (000). Rubouts are not processed by the service routine but are passed on to the user. The special control characters 'control' O and 'control' U have no effect. Other characters are processed as in ASCII mode.

To perform output in DDT mode, use the sequence:

```
MOVEI AC,BUF
CALL AC, [SIXBIT /DDTOUT/]
```

BUF is the first address of a string of packed 7-bit characters terminated by a null (000) character. The Teletype service routine delays until the previous DDTOUT operation is complete, then moves the entire character string into the monitor, begins to output the string, and restarts the user's program. Character processing is the same as for ASCII mode output.

Paper Tape Reader

Device Name - PTR
Buffer Size - 43

Data Modes (Input Only)

A. ASCII - Blank tape (000), rubout (377), and null characters (200) are ignored. All other characters are truncated to seven bits and appear in the buffer. Turning the reader off serves as an end-of-file and results in the character 032 ('control' Z) appearing in the buffer.

AL. ASCII Line - Character processing is the same as for the A mode. The buffer is terminated by LINE FEED, FORM, or VT.

I. Image - There is no character processing. The buffer is packed with 8-bit characters exactly as read from the input tape. Turning the reader off is the end-of-file indication but does not cause a character to appear in the buffer.

IB. Image Binary - Characters not having the eighth hole punched are ignored. Characters are truncated to six bits and packed six to the word without further processing. This mode is useful for reading binary tapes having arbitrary blocking format.

B. Binary - Checksummed binary data is read in the following format. The right half of the first word of each physical block contains the number of data words that follow and the left contains half a folded checksum. The checksum is formed by adding the data words using 2's complement arithmetic, then splitting the sum into three 12-bit bytes and adding these using 1's complement arithmetic to form a 12-bit checksum. The data error status flag (IODERR) is raised if the checksum miscompares. Because the checksum and word count appear in the input buffer, the maximum block length is 40. The byte pointer, however, is initialized so as not to pick up the word count and checksum word.

Again, turning the reader off is the end-of-file indication but does not result in putting a character in the buffer.

Paper Tape Punch

Device Name - PTP

Buffer Size - 43

Data Modes

A. ASCII - The eighth hole is punched for all characters. Tape-feed without the eighth hole (000) is inserted after form-feed. A rubout is inserted after each vertical or horizontal tab. Null characters (000) appearing in the buffer are not punched.

AL. ASCII Line - The same as A mode. Format control must be performed by the user's program.

I. Image - Eight-bit characters are punched exactly as they appear in the buffer with no additional processing.

IB. Image Binary - Binary words taken from the output buffer are split into six 6-bit bytes and punched with the eighth hole punched in each line. There is no format control or checksumming performed by the I/O routine. Data punched in this mode is read back by the paper tape reader in the IB mode.

B. Binary - Each bufferful of data is punched as one checksummed binary block as described for the paper tape reader.

Special Prog. Op. Service

The first output prog. op. of a file causes about two fanfolds of blank tape to be punched as leader. Following a CLOSE prog. op., an additional fanfold of blank tape is punched as trailer. No end-of-file character is punched automatically.

Line Printer

Device Name - LPT

Buffer Size - 34

Data Modes

A. ASCII - ASCII characters are transmitted to the line printer exactly as they appear in the buffer. See the PDP-6 Handbook, F-65, for a list of the vertical spacing characters.

AL. ASCII Line - This mode is exactly the same as A and is included for programming convenience. All format control must be performed by the user's program including placing a RETURN, LINE-FEED at the end of each line.

Special Prog. Op. Service

The first output prog. op. of a file and the CLOSE prog. op. at the end of a file cause an extra form-feed to be printed to keep files separated.

Card Reader

Device Name - CDR

Buffer Size - 36

Data Modes

A. ASCII - All 80 columns of each card are read and translated to 7-bit ASCII code. Blank columns are translated to spaces. At the end of each card a carriage-return/line-feed is appended. A card with the character 12-11-0-1 punched in column 1 is an end-of-file card. Columns 2 through 80 are ignored, and an end-of-file character 032 appears as the last character in the input buffer. The end-of-file button on the card reader has the same effect as the end-of-file card. As many complete cards as can fit are placed in the input buffer, but cards are not split between two buffers. Using the standard-sized buffer, only one card is placed in each buffer. The left arrow character 137 appears in each column containing an invalid punch. The appendix contains the translation table for punching the special ASCII characters on cards.

AL. ASCII Line - Exactly the same as the A mode.

I. Image - All 12 punches in all 80 columns are packed into the buffer as 12-bit bytes. The first 12-bit byte is column one. The last word of the buffer contains columns 79 and 80 as the left and middle bytes respectively. The end-of-file card and the end-of-file button are processed the same as in the A mode with the character 0032 appearing in the buffer as the last character of the file. Cards are not split between two buffers.

B. Binary - Card column one must contain a 7-9 punch to verify that the card is in binary format. The absence of the 7-9 punch results in raising the IOIMPM (improper mode) flag in the card reader status word. Card column two must contain a 12-bit checksum as described for the paper tape reader binary format. Columns 3 through 80 contain binary data, 3 columns per word for 26 words. Cards are not split between two buffers. The end-of-file card and the end-of-file button are processed the same as in the A mode with a word containing 003200000000 appearing as the last word in the file.

DECtape

Device Name - DTA0, DTA1, ..., DTA7

Buffer Size - 202

Data Modes

A. ASCII - Data is written on DECtape exactly as it appears in the buffer. No processing or checksumming of any kind is performed by the service routine. The self-checking of the DECtape system is sufficient assurance that the data is correct. See the description of DECtape format below for further information concerning blocking of information.

AL. ASCII Line - Same as A.

I. Image - Same as A. Data consists of 36-bit words.

IB. Image Binary - Same as I.

B. Binary - Same as I.

DR. Dump Records - This mode is accepted but actually functions as dump mode 17.

D. Dump - Data is read into or written from anywhere in the user's core area without regard to the standard buffering scheme. Control for read or write operations must be via a command list in core memory. The command list format is as described earlier in this chapter under Input/Output Prog. Ops. (Dump Mode), except that the GOTO command is not implemented; any positive number appearing in a command list terminates the list. Dump data is blocked into standard-length DECtape blocks by the DECtape control automatically. Unless the number of data words is an exact multiple of the standard length of a DECtape block (200), after each output prog. op., the remainder of the last block written is wasted. The input prog. op. must specify the same number of words that the corresponding output prog. op. specified in order to skip over the wasted fractions of blocks.

DECtape Format

A standard reel of DECtape consists of 1102 (octal) prerecorded blocks each capable of storing 200 36-bit words of data. Between blocks are recorded block numbers which label the blocks for addressing purposes. These block numbers run from 0 to 1101. Block 0 is normally not used during time-sharing and is reserved for a bootstrap loader. Block 1 is the directory which contains the names of all files on the tape and the block number where each file begins. Blocks 2 through 1101 are usable for data.

If in the process of DECtape I/O, the I/O service routine is requested to use a block number larger than 1101 or smaller than 0, the monitor stops the user's program and types the following error message on the user's console:

ILL DT BLOCK NO., BUFFER AT X DEVICE DTAY

where X is the address of the ring buffer involved and DTAY is the offending DECtape unit.

The directory format is:

Word 0	Left half	Last block number in use.
	Right half	The word number in the directory block where the first directory entry is found (normally 5). Thus, words 1 through 4 are normally unused.

Each directory entry contains:

First word		The file name in sixbit code.
Second word	Left half	The file name extension in sixbit code.
	Right half	The first block number of the file.
Third word	Bits 0-23	Unused (zero).
	Bits 24-35	The date compressed according to the formula: ((year-1964)*12+(month-1))*31+day-1
Fourth word		Unused in nondump files. For dump files requiring a single-word command list, the command list word may be stored here by the user's program.

Unused words in the directory contain zero. Thus, zero marks the end of the list of directory entries. Thirty entries (decimal) fit in one directory.

When a directory search is begun for either LOOKUP or ENTER, the service routine examines the right half of word 0 of the directory. If this half word is not in the range from 1 through 174, the monitor stops the user's program and types the following error message on the user's console:

BAD DIRECTORY FOR DEVICE DATAX;EXEC CALLED FROM USER LOC N

The file format is:

A file consists of any number of DECtape blocks. A dump file consists of the required number of adjacent blocks, but a nondump file may consist of nonadjacent blocks. For nondump files only, each block contains:

Word 0	Left half	The link. The link is the block number of the next block in the file. If the link is zero, this block is the last in the file.
	Right half	A count of the number of words in this block that are used (maximum 177).
Words 1 through 177		Data packed exactly as the user placed it in his buffer.

Special Prog. Op. Service

Several prog. ops. are provided for manipulating DECtape. These prog. ops. allow the user to manipulate block numbers and to operate upon directories.

Prog. Op.	Effect
USETI D,E	Sets the DECtape on device channel D to input block E next. Input operations on this DECtape must not be active because otherwise the user has no way of determining which buffer contains block E.
USETO D,E	Similar to USETI but sets the output block number. USETO waits until the device is inactive before setting up the new output block number.
UGETF D,E	Automatically increments the free block pointer, places the number of the first unused block in user's location E, and does a USETO D,E.
ENTER D,E error return	User's locations E, E+1, E+2, and E+3 must be reserved for a directory entry. The DECtape service routine searches the directory for a file name and extension that match the contents of E and the left half of E+1. If the extension (left half of E+1) is zero, it is not included in the search. If no match is found and four free words are found at the end of the directory, the service routine places the first free block number into the right half of E+1, places the date in E+2 (unless already nonzero), and copies the resulting four words into the four free words at the end of directory. Then the free block pointer is updated and a USETO is performed. If a match is found, similar actions occur, but the new entry replaces the old. If there is no room in the directory, ENTER returns to the next location. Otherwise, ENTER skips one location.
LOOKUP D,E error return	Similar to ENTER but sets up an input file. The contents of E and the left half of E+1 (if nonzero) are matched against the corresponding words of each directory entry. If a match is found, the entire 4-word entry is read from the directory into the 4-word block at E. Then a USETI is automatically performed using the first block number of the file which is found in the right half of E+1. If no match is found, LOOKUP returns to the user's program at the next location. Otherwise, LOOKUP skips one location.
CALL D, [SIXBIT /UTPCLR/]	UTPCLR clears the directory of the DECtape on device channel D. A cleared directory has word 0 containing 1 in the left half and 5 in the right half and all other words containing zero. Only the directory block (block 1) is affected by UTPCLR; the other blocks are unaffected. This prog. op. does nothing if the device on channel D is not DECtape.

In addition to the prog. ops. above, INPUT, OUTPUT, CLOSE, and RELEAS have special effects. When performing nondump input operations, the DECtape service routine reads the links in each block to determine the next block to read and when to raise the end-of-file flag.

When an OUTPUT prog. op. is given, the DECtape service routine examines the left half of the first data word in the output buffer (the word containing the word count in the right half). If this half word contains zero, the next free block number is placed there, the free block pointer is updated, and the service routine gets set to write in that free block upon the next OUTPUT prog. op. If this half word contains 1, it is replaced with a 0 before being written out, and the file is thus terminated. If this half word is greater than 1, it is not changed and the service routine uses it as the block number for the next OUTPUT prog. op..

For both INPUT and OUTPUT, block 1 (the directory) is treated as an exception case. The "link" in block 1 is actually the free block pointer and, therefore, is not altered on output. Thus, if the user's program gives:

```
USETI D,1
```

to read block 1, it is treated as a 1-block file.

The prog. op. CLOSE places a 1 in the left half of the first word in the last output buffer, thus, terminating the file.

The prog. op. RELEAS writes onto block 1 the copy of the directory which is normally kept in core, but only if any changes have been made. Certain console commands, such as KJOB or CORE 0, perform an implicit RELEAS of all devices and, thus, write out a changed directory even though the user's program failed to give a RELEAS.

Special Status Bits

If an attempt to write is made on a unit with the write-lock switch on, the device error flag (IODERR) is raised.

Important Considerations

If an attempt is made to use a unit that is turned off or that has no tape mounted, all DECtape units will be hung up until a tape is actually mounted. Good practice is to mount scratch tapes on all transports not otherwise in use.

The DECtape service routine reads the directory from a tape the first time it is required to perform a LOOKUP, ENTER, or UGETF. The directory image in core is deleted following the next RELEAS unless

the user assigns a tape unit to his job using the ASSIGN console command. If the unit is assigned, the DECtape service routine reads the directory from block 1 only once, the first time it is required and thereafter uses the directory image in core memory regardless of the appearance of RELEAS prog. ops. To inform the DECtape service routine that a new tape has been mounted on an assigned unit, the user must use one of the console commands IJOB, KJOB, ASSIGN, or DEASSIGN, whichever is appropriate to the situation. The directory from the old tape could be transferred to the new tape, thus destroying the information on that tape unless the user reassigns the DECtape transport every time he mounts a new reel.

Magnetic Tape

Magnetic tape format is IBM compatible and is not described here.

Device Name - MTA0, MTA1, ..., MTA7

Buffer Size - 203

Data Modes

A. ASCII - Data is written on magnetic tape exactly as it appears in the buffer. No processing or checksumming of any kind is performed by the service routine. The parity checking of the magnetic tape system is sufficient assurance that the data is correct. Normally, all data, both binary and ASCII, is written with odd parity and at 556 bits per inch. A maximum of 200 words per record is standard. The wordcount is not written on the tape.

AL. ASCII Line - Same as A.

I. Image - Same as A but data consists of 36-bit words.

IB. Image Binary - Same as I.

B. Binary - Same as I.

DR. Dump Records - Variable length records are read into or written from anywhere in the user's core area without regard to the standard buffering scheme. Control for read or write operations must be via a command list in core memory. The command list format is as described earlier in this chapter under Input/Output Prog. Ops. (Dump Mode). For input operations a new record is read for each word in the command list (except GOTO words); if the record terminates before the command word is satisfied, the service routine skips.

to the next command word. If the command word runs out before the record terminates, the remainder of the record is ignored. For each output command word, exactly one record is written.

D. Dump - This mode is accepted but actually functions as DR mode 16.

Special Prog. Op. Service

The CLOSE prog. op. performs a special function for magnetic tape. When an output file is closed (both dump and nondump), the I/O service routine automatically writes two end-of-file marks and backspaces over one of them. If another file is now opened, the second end-of-file is wiped out leaving one end-of-file between files. At the end of the in-use portion of the tape, however, there appears a double end-of-file character which is defined as the logical end of tape. When an input dump file is closed, the I/O service routine automatically skips to the next end-of-file.

A special prog. op. called MTAPE provides for such tape manipulation functions as rewind, backspace record, backspace file, etc. The format is:

MTAPE D,FUNCTION

where D is the device channel on which the magnetic tape unit is initialized. FUNCTION is selected according to the following table:

Function	Action
1	Rewind to load point
11	Rewind and unload
7	Backspace record
17	Backspace file
3	Write end of file
6	Skip one record
13	Write 3 inches of blank tape
16	Skip one file
10	Space to logical end of tape

MTAPE waits for the magnetic tape unit to complete whatever action is in progress before performing the indicated function. Bits 18 through 25 of the status word are then cleared, the indicated function is initiated, and control is returned to the user's program immediately. It is important to remember that

when performing buffered input/output, the I/O service routine can be reading several blocks ahead of the user's program. The MTAPE prog. op. affects only the physical position of the tape and does not change the data that has already been read into the buffers.

Special Status Bits

Special bits of the status word are reserved for selecting the density and parity mode of the magnetic tape. The following bits are set and cleared by the INIT or SETSTS prog. op..

Bit	Name	Action
29	IORCK	I/O no read check. Suppress automatic error correction if bit 29 is a 1. Normal error correction is to repeat the desired operation 10 times before setting an error status bit.
27,28	IODENS	I/O density. 00 or 10 =556 bpi 01 =200 bpi 11 =800 bpi
26	IOPAR	I/O parity. 0 for odd parity, 1 for even parity.

The following bits indicate special magnetic tape conditions and are set by the magnetic tape service routine when the conditions occur.

Bit	Name	Meaning
19	IODERR	I/O Device ERRor. When set to one during an output operation means that the write protect ring is out.
24	IOBOT	I/O Beginning of Tape. The tape is at the load point.
25	IOTEND	I/O Tape END. The tape is at or past the end point.

MISCELLANEOUS MONITOR FUNCTIONS

The following monitor functions provide various user services. All of these functions are obtained by the CALL prog. op. using the following format:

CALL AC, [SIXBIT /FUNCT/]

AC specifies the accumulator that either contains an argument for the function or receives the function value. FUNCT is the name of the monitor routine which performs the action.

Date

The prog. op.:

```
CALL AC, [SIXBIT /DATE/]
```

loads the date into accumulator AC. The date is compressed into 12 bits by the formula:

```
date=((year-1964)*12+(month-1))*31+day-1
```

Exit

The prog. op.:

```
CALL [SIXBIT /EXIT/]
```

closes and releases all I/O devices, stops execution, prints the message "EXIT" on the user's console, and places the console into monitor mode.

Setddt

The sequence:

```
MOVEI AC,ADR
CALL AC, [SIXBIT /SETDDT/]
```

places ADR into location JOBDDT of the job data area as the starting address used when the user types the DDT console command. The user's program cannot alter the contents of JOBDDT except by using the SETDDT function because that location is protected by the monitor.

Setpov

SETPOV allows the user to program an error correction procedure for pushdown list overflows. The sequence:

```
MOVEI AC,ADR
CALL AC, [SIXBIT /SETPOV/]
```

places ADR into location JOBPOV of the job data area. On a pushdown overflow, the monitor traps the user's program to location ADR if ADR is nonzero. Location JOBPOV is protected by the monitor and cannot be changed except by the use of SETPOV. If SETPOV has not been used and a pushdown list overflow occurs, the monitor stops the user's program and types the following message on the user's console:

```
PDL OV AT USER LOC N
```

Switch

The prog. op.:

CALL AC, [SIXBIT /SWITCH/]

places the contents of the console data switches into accumulator AC.

Timer

The prog. op.:

CALL AC, [SIXBIT /TIMER/]

places the time of day, in 60ths of a second from midnight, into accumulator AC (50ths of a second if the power line frequency is 50 cps.).

Special Loader Function

The following is a very special purpose loader function and is subject to change or deletion with no advance notice. This function is provided for the convenience of the loader. The prog. op.:

CALL [SIXBIT /LDRBLT/]

does the following:

1. All devices are released.

2. The contents of accumulator 2 (henceforth referred to as the loader offset) are added to the contents of JOBSYM in the offset job data area.

3. The user's program is assumed to be loaded above the location in which it will run by an amount equal to the loader offset. The user's program is block transferred down appropriately. The lowest location transferred is 30, the highest is determined by the contents of the left half of JOBSA in the offset job data area.

4. A SETDDT operation is performed on the contents of JOBDDT in the new job data area.

5. The job is stopped and the message:

LOADER FINISHED

is typed on the user's console. The console is returned to monitor mode.

6. The contents of accumulator 2 are set to zero.

EXAMPLES

Typical Program Initialization

```
BEG:        CALL [SIXBIT /RESET/]
            :
            :
```

General Device Initialization

```
INIDEV:     0                       ;JSR HERE
            INIT 3,14               ;BINARY MODE,CHANNEL 3
            SIXBIT /DTA5/           ;DEVICE DECTAPE UNIT 5
            XWD OBUF,IBUF           ;BOTH INPUT AND OUTPUT
            JRST NOTAVL             ;WHERE TO GO IF DTA5 IS BUSY
;FROM HERE DOWN IS OPTIONAL DEPENDING ON THE DEVICE AND PROGRAM
;REQUIREMENTS
            MOVE 0, JOBFF
            MOVEM 0,SVJBFF          ;SAVE THE FIRST ADDRESS OF
                                    ;THE BUFFER RING IN CASE THE
                                    ;SPACE MUST BE RECLAIMED.
            CALL AC, [SIXBIT /DATE/]
            MOVEM AC,OUTNAM+2       ;PUT THE DATE INTO THE DIRECTORY
                                    ;ENTRY
            INBUF 3, 4              ;SET UP 4 INPUT BUFFERS
            OUTBUF 3,1              ;SET UP 1 OUTPUT BUFFER
            LOOKUP 3, INNAM         ;INITIALIZE AN INPUT FILE
            JRST NOTFND             ;WHERE TO GO IF THE INPUT FILE
                                    ;NAME IS NOT IN THE DIRECTORY
            ENTER 3, OUTNAM         ;INITIALIZE AN OUTPUT FILE
            JRST NOROOM             ;WHERE TO GO IF THERE IS NO
                                    ;ROOM IN THE DIRECTORY FOR
                                    ;A NEW FILE NAME.
            JRST@ INIDEV            ;RETURN TO MAIN SEQUENCE
OBUF:       BLOCK 3                 ;SPACE FOR OUTPUT BUFFER HEADER
INBUF:      BLOCK 3                 ;SPACE FOR INPUT BUFFER HEADER
INNAM:      SIXBIT /NAME/           ;FILE NAME
            SIXBIT /EXT/            ;FILE NAME EXTENSION (OPTIONALLY 0),
                                    ;RIGHT HALF WORD RECEIVES THE
                                    ;FIRST BLOCK NUMBER
            0                       ;RECEIVES THE DATE
            0                       ;UNUSED FOR NONDUMP I/O
OUTNAM:     SIXBIT /NAME/           ;SAME INFORMATION AS IN INNAME
            SIXBIT /EXT/
            0
            0
```

General Input One Character Subroutine

```
GETCHR:  0                      ;JSR HERE
         SOSLE IBUF+2           ;DECREMENT THE ITEM COUNT
         JRST GETOK             ;DATA IS ALREADY IN THE BUFFER
         INPUT 3,               ;REFILL THE BUFFER
         STATZ 3,740000         ;CHECK THE 4 ERROR BITS
         JRST INERR             ;WHERE TO GO ON AN ERROR
         STATZ 3,20000          ;CHECK END OF FILE BIT
         JRST ENDFIL            ;WHERE TO GO ON END OF FILE
GETOK:   ILDB AC,               ;FETCH CHARACTER FROM THE BUFFER
         JRST@ GETCHR           ;RETURN
```

General Output One Character Subroutine

```
PUTCHR:  0                      ;SIMILAR TO GETCHR BUT NO END
                                ;OF FILE CHECKING IS REQUIRED
         SOSLE OBUF+2
         JRST PUTOK
         OUTPUT 3,
         STATZ 3,740000
         JRST OUTERR
PUTOK:   IDPB AC, OBUF+1
         JRST@ PUTCHR
```

Simplified I/O Subroutines

```
GETCHR:  0
         SOSG IBUF+2
         INPUT 3,
         ILDB AC,IBUF+1
         JRST@ GETCHR
PUTCHR:  0
         SOSG OBUF+2
         OUTPUT 3,
         ILDB AC, OBUF+1
         JRST@ GETCHR
```

Terminating A File

```
DROPDV:  0                      ;JSR HERE
         CLOSE 3,               ;WRITE END OF FILE AND TERMINATE
                                ;INPUT
         STATZ 3, 740000        ;RECHECK FINAL ERROR BITS
         JRST OUTERR            ;ERROR DURING CLOSE
         RELEAS 3,              ;RELINQUISH THE USE OF THE
                                ;DEVICE, WRITE OUT THE DIRECTORY
         MOVE 0, SVJBFF
         MOVEM 0, JOBFF         ;RECLAIM THE BUFFER SPACE
         JRST@ DROPDV           ;RETURN TO MAIN SEQUENCE
```

Halting A Program

```
ENDPRG:   CALL [SIXBIT /EXIT/]          ;CLOSES AND RELEASES ALL DEVICES
```

User-Generated Buffers

The following code illustrates avoidance of the INBUF prog. op. Analogous code may replace OUTBUF. This code also illustrates the operation of INBUF. SIZE must be set equal to the greatest number of data words expected in one physical record.

```
GO:       INIT 1, 0                     ;INITIALIZE ASCII MODE
          SIXBIT /MTA0/                 ;MAGNETIC TAPE UNIT 0
          XWD 0, MAGBUF                 ;INPUT ONLY
          JRST NOTAVL
          MOVE 0, [XWD 400000,BUF1+1 ]  ;THE 400000 IN THE LEFT HALF
                                        ;MEANS THE BUFFER WAS NEVER
                                        ;REFERENCED.
          MOVEM 0, MAGBUF
          MOVE 0, [POINT BYTSIZ,0,35]   ;SET UP NONSTANDARD BYTE SIZE
          MOVEM 0, MAGBUF+1
          JRST CONTIN                   ;GO BACK TO MAIN SEQUENCE
MAGBUF:   BLOCK 3                       ;SPACE FOR BUFFER HEADER
BUF1:     0                             ;BUFFER 1, 1ST WORD UNUSED
          XWD SIZE+1,BUF2+1             ;LEFT HALF CONTAINS BUFFER SIZE,
                                        ;RIGHT HALF HAS ADDRESS OF NEXT
                                        ;BUFFER
          BLOCK SIZE+1                  ;SPACE FOR DATA,1ST WORD RECEIVES
                                        ;WORD-COUNT . THUS ONE MORE WORD
                                        ;IS RESERVED THAN IS REQUIRED
                                        ;FOR DATA ALONE
BUF2:     0                             ;SECOND BUFFER
          XWD SIZE+1,BUF3+1
          BLOCK SIZE+1
BUF3:     0                             ;THIRD BUFFER
          XWD SIZE+1,BUF1+1             ;RIGHT HALF CLOSES THE RING
          BLOCK SIZE+1
```

Dump Output

Dump input is similar to dump output. This routine outputs fixed-length records.

```
DMPINI:   0                             ;JSR HERE TO INITIALIZE A FILE
          INIT 0,16                     ;CHANNEL 0, DUMP MODE
          SIXBIT /MTA2/                 ;MAGNETIC TAPE UNIT 2
          0                             ;NO RING BUFFERS
          JRST NOTAVL                   ;WHERE TO GO IF UNIT 2 IS BUSY
          JRST @ DMPINI                 ;RETURN
```

```
DMPOUT:  0                      ;JSR HERE TO OUTPUT THE OUTPUT AREA
         OUTPUT 0, OUTLST       ;SPECIFIES DUMP OUTPUT ACCORDING
                                ;TO THE LIST AT OUTLIST
         STATZ 0, 740000        ;CHECK ERROR BITS
         CALL [SIXBIT /EXIT/]   ;QUIT IF AN ERROR OCCURS
         JRST@ DMPOUT           ;RETURN
DMPDON:  0                      ;JSR HERE TO WRITE AN END OF FILE
         CLOSE 0,               ;WRITE THE END OF FILE
         STATZ 0,740000         ;CHECK FOR ERROR DURING WRITE
                                ;END OF FILE OPERATION
         CALL [SIXBIT /EXIT/]   ;QUIT IF ERROR OCCURS
         RELEAS 0,              ;RELINQUISH THE DEVICE
         JRST@ DMPDON           ;RETURN
OUTLST:  IOWD BUFSIZ, BUFFER    ;SPECIFIES DUMPING A NUMBER OF
                                ;WORDS EQUAL TO BUFSIZ, STARTING
                                ;AT LOCATION BUFFER
         0                      ;SPECIFIES THE END OF THE COMMAND
                                ;LIST
BUFFER:  BLOCK BUFSIZ           ;OUTPUT BUFFER, MUST BE CLEARED
                                ;AND FILLED BY THE MAIN PROGRAM
```

CHAPTER 5

ERROR DIAGNOSTICS

MONITOR DETECTED ERRORS

Whenever a job commits an error so serious that it must be stopped, the monitor types the following preamble followed by an appropriate message to describe the fault:

ERROR IN JOB N

where N is the job number assigned to the offending job. Following the typing of an error message on the next line, the monitor returns the console to monitor mode.

In certain circumstances, such as misuse of the computer console EXAMINE switch to examine nonexistent memory, an error occurs while no job is running. In this case the monitor types:

ERROR IN JOB 0

The operator's console is the first console that received typein when the monitor was started (see Appendix 4). If the error cannot be explained by someone's misuse of the manual keys, the fault is either a monitor bug or hardware failure and should be brought to the attention of Digital personnel.

ILLEGAL MEMORY REFERENCES

If the user's program references memory above its core allotment, a hardware-initiated trap to the monitor occurs. The exec then stops the user's program and types the following message on the user's console:

ILL MEM REF AT USER LOC N

where N is the address of the offending instruction.

If the user's program attempts to jump to an address above its core memory assignment, the message becomes:

PC EXCEEDS MEM BOUND AT USER LOC M

where M is the address to which the user's program jumped.

The following message occurs only through hardware failure or a bug in the Time-Sharing Monitor. The non-existent memory trap causes the monitor to stop the user's job and type the following message on the user's console:

NON EX MEM AT EXEC LOC N

or

NON EX MEM AT USER LOC N

depending on whether the trap occurred from the user's program or from the exec. The repeated occurrence of this message should be reported to Digital personnel. The recommended recovery procedure is to examine the memory indicator lamps for memories that are hung up. Clear any such memories and restart the monitor. If no such memories are discovered, reload and restart the monitor.

If the user's program requests the monitor to perform I/O into or from any location above the user's core allotment, in the protected portion of the job data area, or (during interrupts) in the accumulators, the monitor stops the user's program and types the following message on the user's console:

ADDRESS CHECK FOR DEVICE X; EXEC CALLED FROM N

where X is the physical device name and N is the address of the offending prog. op. This error message frequently occurs when the user does not have enough core to satisfy all INBUF and OUTBUF programmed operators. If the improper memory reference occurs during an interrupt, the form of the message is slightly different:

ADDRESS CHECK FOR DEVICE X DURING INTERRUPT

This message frequently occurs when the user's program (via a program bug) destroys the ring buffer structure so that one buffer in the ring points outside the user's core to the "next" buffer.

The difference between the ADDRESS CHECK message and the ILL MEM REF message is that the former is detected by subroutines in the exec while the latter is detected by hardware.

ILLEGAL INSTRUCTIONS

Illegal instructions are all I/O instructions; prog. op. 0 and all undefined prog. ops. above 40; and JRST 10, or JRST 4,. All these instructions trap to the exec which stops the user's program and types the following message on the user's console:

ILL PROG. OP. USED AT USER LOC N

where N is the address of the offending prog. op., or:

ILL INST AT USER LOC N

depending on whether the offending instruction was a prog. op. or was a JRST or I/O instruction respectively.

OUTPUT TO AN INPUT DEVICE AND VICE VERSA

If the user's program attempts output to an input-only device or vice versa, upon the occurrence of the faulty INPUT or OUTPUT prog. op. the monitor stops the user's program and types the following error message on the user's console:

INPUT DEVICE X CANNOT DO OUTPUT; EXEC CALLED FROM USER LOC N

or

OUTPUT DEVICE X CANNOT DO INPUT; EXEC CALLED FROM USER LOC N

where X is the physical name of the device involved and N is the address of the offending prog. op.

MONITOR ERROR

If an undeterminable error occurs, either through a bug in the monitor or hardware failure, the monitor stops the user's program that is running at the time of the failure and types the following message on that user's console:

ERROR IN MONITOR AT EXEC LOC N

This message should be brought to the attention of Digital maintenance personnel. The recovery procedure is to reload the monitor and restart.

CHAPTER 6

THE STANDARD CUSP LIBRARY

THE CONTENTS OF THE CUSP TAPE

The following is an enumeration of the files found on a standard CUSP tape. Only those programs that are not described in separate manuals are covered below in complete detail. Table 6-1 contains a summary of the commands and error messages relevant to the most commonly used CUSPs.

The following programs are found on a standard CUSP tape. Those labeled "Dump" are loaded by the GET console command. Those labeled REL (relocatable binary) are loaded by the binary program loader which is itself a dump program.

Program Name	Type	Full Name	Size
9KF2	Dump	FORTRAN Compiler (two-pass version)	9K
FOLA	Dump	FORTRAN On-Line Assembler (for two-pass FORTRAN)	9K
F2	Dump	FORTRAN II (one-pass version)	22K
MACRO	Dump	MACRO Assembler	9K
PIP1	Dump	Peripheral Interchange Program (1K version)	1K or more
PIP2	Dump	Peripheral Interchange Program (2K version)	2K or more
EDITOR	Dump	DECtape Editor	1K
LOADER	Dump	Relocatable Binary Program Loader	1K plus object program size plus I/O buffers for storage map
DESK	Dump	Desk Calculator	2K
DDT	Rel	Digital Debugging Technique (on-line debugger)	1.5K plus symbol table
JOBDAT	Rel	Job Data Area Definitions	0 (defines external symbols only)
FORLIB	Rel	FORTRAN operating system and subroutine library	Up to 2.7K

Program Name	Type	Full Name	Size
FUDGE	Dump	File Updating Generator	2K or more
RIM DTA	Dump	RIM Paper Tape to DECtape Dump Conversion	1K plus the size of the program being copied

GENERAL CUSP COMMAND FORMAT

Many of the CUSPs produce output and accept input from all devices. These CUSPs (FORTRAN, FOLA, PIP, LOADER) require that the user type in the names of the devices he wishes to use. The typein formats for these CUSPs are similar. In general the format is:

X,X←X,X,X ...) or ALTMODE

where each "X" represents a file descriptor as illustrated below. The files to the left of the left arrow are destination (output) files; those to the right are source (input) files. The file descriptors have the following format:

device:filename.extension

where "device" is the logical or physical name of the desired device, "filename" is a 1- to 6-character file name, and "extension" is a 1- to 3-character file name extension.

For the convenience of the user, the command format may be abbreviated as follows. The file name extension need not be typed, in which case the period should also be omitted. For devices other than DECtape, no file name is required, in which case only a device name followed by a colon must be typed. If the device name is omitted, the most recently typed device name is used. Thus to specify more than one file from the same device, the user types:

```
PTR:,,                    (three paper tape files)
PTR:,                     (two files)
PTR:                      (one file)
DTA3:file1,file2,file3    (three DECtape files)
```

When a CUSP requires more than one output file, the first file on the left is the binary file (or the intermediate file in the case of two-pass FORTRAN) and the second file is the listing file.

Most CUSPs allow the user to select several options or modes and to perform certain device-manipulation functions (such as rewinding magnetic tape). These options, called switches, are selected by typing a

single letter preceded by a slash or by typing several letters enclosed in parentheses. Each letter selects an option as determined by the particular CUSP being used. For example, to command PIP to copy a binary paper tape, the user types:

PTP;/B← PTR:

where "B" selects binary mode for PIP. To copy a card file, the user might type:

DTA2:CARDS(CS) ←CDR:

where "CS" selects both Card input format and the insertion of Sequence numbers.

PERIPHERAL INTERCHANGE PROGRAM

PIP1 and PIP2 are programs to transfer data from one I/O medium to another. PIP performs simple editing functions during data transmission.

Size: PIP1=1K or more, PIP2=2K or more (both use all available storage).
Equipment: Teletype for control, other devices as desired.
Loading: Run as a user program.

Operating Instructions

Commands to PIP are in the standard CUSP format consisting of one output file and any number of input files. Nonstandard formats and dump files cannot be copied with PIP.

PIP2 accepts the switches specified below. PIP1 accepts only the switches not marked with an asterisk. Switches may appear in the command line anywhere to the left of the first comma (or anywhere if a comma is not required).

If switches are not typed, PIP transmits ASCII data (including sequence numbers if present) from each successive source to the destination, changing only the character following a sequence number to be either a tab or a carriage return. Thus, a binary file cannot be copied without the B switch.

Switch	Action
A	ASCII line mode processing. (Do not break up lines between buffers.)
B	Process binary data.
C*	Suppress trailing spaces at the end of each line and change multiple blanks into tabs. (Useful for card input.)
D*	Delete all source file names from the directory of the destination device. The destination device name supersedes any other device name that is typed.

Switch	Action
F	FORTRAN format, one line per record.
L*	List the directory of the source device. Only one source may be specified.
N	Remove sequence numbers.
R*	Rename the source file to have the name of the destination file. No source device name is necessary because the destination device is implied. Only one source may be specified.
S	A sequence number starting with 00010 and incrementing by 10 is added to the front of each line. A tab is inserted after the sequence number unless the line consists of form-feed, vertical tab, or line-feed, in which case a carriage-return is inserted. If a file is already sequenced, the old sequence numbers are removed. Sequence numbers are never split between words and are identified by a one appearing in bit 35.
T*	The same as C, except do not replace multiple blanks with tabs.
W	Rewind magnetic tape. Source files are ignored.
X*	Copy an entire DECtape, recovering unused blocks. Empty files are not copied. Does not disturb data previously written on the destination tape unless file names conflict. Although the source device must be DECtape, the destination may be any output device.
Z	Clear the destination DECtape and create a fresh directory. Sources are ignored.

When PIP has finished copying a file, it types a carriage-return/line-feed and is ready for the next command.

Some of the error messages that PIP might type out are:

DECTAPE INPUT REQUIRED	A directory manipulation command (L,R,D) was given for a device other than DECtape.
USE PIP2	One of the * commands was given to PIP1.

Examples: PIP1 and PIP2

To copy three paper tapes onto one DECtape file, type:

```
DTA1:BIGFIL ← PTR:,,
```

To do the same as above, but with inserting sequence numbers, type:

```
DTA1:BIGFIL/S ← PTR:,,
```

To list a file, type:

```
LPT: ← DTA1:BIGFIL
```

To do the same as above, but stripping off seauence numbers, type:

```
LPT:/N← DTA1:BIGFIL
```

To rewind magnetic tape:

```
MTA1:/W←
```

To copy binary data:

```
DTA1:PIP2.REL/B← PTR:
```

To create a fresh directory:

```
DTA1:/Z←
```

PIP2 Examples:

To clean up a DECtape and recover unused blocks, mount the source tape on unit 1 and a scratch tape on unit 2 (for example). Then type:

```
DTA2:/Z←
DTA2:/X← DAT1:
```

Return your old tape to the scratch tape supply and take possession of the tape on unit 2.

To rename files, type:

```
DTA2NEWNAM.EXT/R← OLDNAM
DTA2:AFTER/R    BEFORE
```

To delete files, type:

```
DTA2:/D ← FILE1, FILE2, FILE3
```

To list a directory:

```
TTY:/L ← DTA2:
```

FORTRAN AND FOLA

FORTRAN II is implemented in two versions, a 9K version that produces an intermediate, assembly-language file which must be assembled in a second pass by FOLA (FORTRAN On-Line Assembler), and a 22K version that contains FOLA within itself and produces relocatable binary output in one pass. The 22K version accepts all the commands that the 9K version accepts and all the commands that FOLA accepts.

Size: 9KF2=9K (F2=22K) (FOLA=9K)
Equipment: Teletype for control, other devices as desired.
Loading: Run as a user program.

Operating Instructions

For a description of the FORTRAN II language, see programming manual DEC-6-0-TP-FII-LM-FP-ACT00. The user may specify up to two destination files and any number of source files. File names are in the standard format. In general, the command format is:

For 9KF2: Intermediate File, Intermediate Listing ← Sources

For F2: Binary File, Listing ← Sources

For FOLA: Binary File, Listing ← Intermediate File

For a listing file only, use the form:

, Listing ← Sources

For no listing, use:

Binary File, ← Sources

To scan for language errors but to produce no output, use:

, ← Sources

Switches

There are two switches recognized by FORTRAN (marked by *), three by FOLA (marked by **), and three recognized by both. All eight are recognized by 22KF2.

Switch	Action
T*	The source is in paper-tape image format.
M**	Do not print a symbol map after compilation.
Z	Clear the directory on the device before inserting the new file.
R*	The source file is a set of FORTRAN II rules instead of source statements.
K	Skip a file before compilation.
W	Rewind before compilation.
N**	Do not put errors on TTY is there is a listing file.
S**	List source only.

The K and W switches are for mag-tape only.

See the PDP-6 FORTRAN II Manual for the differences between card and paper-tape image formats. To facilitate using card image format with TTY or PTR input, a tab in columns 1-6 will cause a skip to column 7. No colon should appear in columns 1-6.

More than one switch may be enclosed within parentheses; a single switch may appear after a forward slash. Any ASCII characters other than the above recognized switches will be ignored.

The switches N, S, T, M, and R may appear anywhere in the command string before the terminating carriage-return. If switch K, W, or Z is desired, it must appear before the terminating character (comma, left arrow, or carriage-return) for the desired file. The Z switch is ignored if it appears after the left arrow.

FORTRAN is normally dumped with a complete set of syntax rules preloaded. The rules need not be read in again unless they are changed. If they are to be changed, the compiler need not be reloaded; instead "GET" the previous version, and type in the following command string to read in rules:

(R)DTA1:RULES

(assuming the rules are on DTA1 in a file called "RULES"). The compiler will terminate by calling exit; "EXIT" will be typed out. The compiler may then be saved. The writing of syntax rules is described in the document, Writing Syntax Rules, DEC-6-00-TP-SC-UM-IP-PRE2. FORTRAN types "EXIT" when compilation is complete. The lack of any messages to the contrary indicates there were no errors. FORTRAN must be reloaded from the CUSP tape to perform additional compilations.

MACRO

MACRO is an assembly program featuring a powerful macro-instruction processor and relocatable assembly with automatic linking of externally defined symbols at load time.

Size: 9K (a 13K version is available by special request)
Equipment: Teletype for control, other devices as desired.
Loading: Run as a user program.

Operating Instructions

For a description of the MACRO assembly language, see programming manual DEC-6-0-TP-MAC-LM-FP-ACT01. The user may specify up to two destination files and any number of source files. File names are in the standard format. In general, the command format is:

Binary File, Listing ← Sources

For a listing without the binary output type:

, Listing ← Sources

For binary with no listing file type:

Binary File, ← Sources

To scan for language errors but to produce no output type:

, ← Sources

Switches

MACRO recognizes the following switches:

Switch	Action
S	Suppress the listing.
L	Reinstate the listing.
E	List macro-instruction expansions.
X	Stop listing macro-instruction expansions.
F	Fix the symbol table so that all symbols currently defined become initial symbols.
N	No error listings on the console.
A	Advance a magnetic tape forward one file.
B	Backspace a magnetic tape one file.
T	Skip to the logical end of a magnetic tape.
W	Rewind magnetic tape.
Z	Clear a DECtape directory.

The last five switches must be typed immediately following the device specification to which they apply.

MACRO terminates reading a source file when it encounters the pseudo-instruction TAPE or END. If an END statement does not appear before the end of the last source file typed in, MACRO types out the message:

NO END STATEMENT ENCOUNTERED ON INPUT FILE

Note that a TAPE pseudo-instruction preceding an END pseudo-instruction in the source file hides the END.

When the assembly is complete, MACRO types:

END OF ASSEMBLY

and is ready to assemble another program.

EDITOR

The EDITOR allows the user to make modifications to an ASCII sequenced file on DECtape. Programming Manual, DEC-06-TP-EDIT-LM-FP-ACT01, describes the operating procedures for the EDITOR.

Size: 1K

Equipment: Teletype for control, one DECtape transport for the edited file.

Loading: Run as a user program.

LOADER

The LOADER allows the user to load binary relocatable programs produced either by MACRO or FORTRAN (FOLA). The LOADER links externally defined symbols to the proper addresses, building a table of such symbols. This table is available to DDT for user's on-line debugging.

Size: 1K plus the size of the loaded program plus the size of the required symbol table.

Equipment: Teletype for control, other devices as desired.

Loading: Run as a user program.

Operating Instructions

The general form for commands to the loader is:

Storage Map File ← Sources

If the storage map file is never specified and the user requests a storage map, it is typed on the Teletype console.

File specifications are in standard format; however, switches are not in standard format. The characters (and / precede switches as in the standard form, but are equivalent to one another. The list of switches is terminated by any nonalphanumeric character. Moreover, certain switches have the same ability to separate file descriptors as does comma; these are marked below with an asterisk.

Switches

The following switches are accepted by the LOADER:

Switch	Action
A	List all global symbols in storage map regardless of program length.
D*	Load DDT, enter load with symbols mode.

Switch	Action
F*	Library search FORLIB, exit from load with symbols mode.
G*	Terminate loading, begin automatic library search of FORLIB, list undefined globals. An octal number may be typed preceding the "G". The numeric value is used as the starting address (if nonzero). The rest of the command line is ignored.
I	Set LOADER to ignore starting addresses in binary input.
J	Set LOADER to accept starting addresses in binary input.
L	Enter library search mode.
M*	Print storage map and undefined list.
N	Leave library search mode.
P	Prevent automatic library search.
Q	Allow automatic library search.
S	Load with local symbols.
U*	List undefined globals on destination file.
W	Load without local symbols.
X	Suppress listing global symbols in zero length programs.
Z	Restart loading. Core cleared, loader set to initial state. The rest of the command line is ignored.

A command string to the loader may consist of one or more lines. A line is terminated by typing carriage-return or alt mode. The loader signals readiness to accept one line of console input by typing an asterisk at the beginning of the line. Relocatable binary files are loaded in the order the source specifications occur. All loading required by one line of a command string occurs after the carriage-return is input, but before the asterisk giving the go-ahead to type the next line is output.

The storage map file may be specified at the beginning of any command line.

The character ALTMOD has the same effect as the G switch except that no starting address may be typed. When the loader is finished, the message:

LOADER FINISHED

is typed on the user's console and the console is placed in monitor mode. The user then has the option to dump his program using the SAVE monitor command or to start his program by typing START.

DESK

The DESK calculator performs numeric calculations on-line. It features single-precision floating point arithmetic, built-in trigonometric and logarithmic functions, and the ability to define functions. Programming Manual, DEC-06-UP-DESK-UM-FP-ACT01, describes the operating instructions for DESK.

Size: 2K

Equipment: Teletype for control and for results.

Loading: Run as a user program.

DDT

DDT provides a powerful on-line debugging language in which the user refers to his program by symbols appearing in the MACRO language source program. The operating instructions for DDT are found in Programming Manual DEC-6-0-UP-DDT-UM-FP-ACT00.

Size: 2K plus the required symbol table.

Equipment: Teletype.

Loading: Load DDT with the LOADER at the same time the program to be debugged is loaded. If DDT is to be loaded first, use the D switch to the LOADER as described above. Otherwise, be sure to load with local symbols (using the S switch) for those programs which are to be debugged. A suitable command to the Loader for loading DDT is either:

SYS:DDT

or

/D

The file JOBDAT is included within the DDT file and need not be loaded separately if DDT has been loaded.

> NOTE: The DDT file produces about three pages of external symbols on a storage map. It is wise to order the loading process so that all required storage maps are obtained before DDT is loaded.

JOBDAT

The JOBDAT file contains the symbolic definitions of addresses in the job data area. Users' programs that refer to the job data area should have all such locations declared as external symbols which are then defined by the JOBDAT file at load time.

Size: 0

Equipment: none

Loading: Load with the LOADER by typing:

SYS:JOBDAT

FORLIB

The file FORLIB contains the FORTRAN operating system and the FORTRAN library of standard functions in Linking Loader format. The operating system is described in the document, FORTRAN II Format and I/O Processor, DEC-06-OS-FII-GM-FP-ACT00. The arithmetic functions are described in the FORTRAN Langauge Manual, DEC-6-0-TP-FII-LM-FP-ACT00.

Size: Between 1K and 2K depending on how much of the library is required.

Equipment: Teletype for error messages and any desired devices.

Loading: Load by the Linking Loader in library search mode (L switch) after all programs requiring the FORTRAN library have been loaded. The loader automatically loads FORLIB from device SYS if undefined external symbols remain when the command to terminate loading is received. However, the user may wish to load FORLIB followed by, say, DDT in which case a suitable command to the LOADER is:

SYS:FORLIB/L,/D

FUDGE

FUDGE is a program for editing relocatable binary files, such as FORLIB, which consist of several independent programs collected into a single file. The operating instructions for FUDGE are given in the document, Description and Operating Instructions File Updating Generator (FUDGE), DEC-06-0-UP-LIB-UM-FP-ACT00.

Size: 2K or more

Equipment: Teletype for control, any three additional devices, one to receive the edited file, one to supply the old master file, one to supply the replacement programs.

Loading: Run as a user program.

RIMDTA

RIMDTA is a user program to read paper tapes in RIM format (see the MACRO manual) and to write out an equivalent dump file onto DECtape. The principal use of this program is to read maintenance tapes onto a single DECtape for convenient handling.

Size: 1K plus the size of the RIM object program.

Equipment: Teletype for control, paper tape reader for reading the RIM tape, any output device for the output files.

Loading: Run as a user program.

Operating Instructions

Only one file is specified to RIMDTA, the output file. Input from the paper tape reader (PTR) is assumed. The output file descriptor is in standard format; no commas, left arrows, or switches are required. The user may specify a date in the following format which appears in the directory for each file.

DEV:FILNAM.EXT(month/day/year)

"Month", "day", and "year" are 2-digit decimal numbers. If the file name extension is not specified, "DMP" is automatically supplied.

When RIMDTA is done, it types out either:

DONE

or

TRANSMISSION ERROR.

"DONE" means that the operation is successful and RIMDTA is ready for the next tape. "TRANSMISSION ERROR" means that the attempt to write on DECtape is unsuccessful (often because the WRITELOCK switch is in the locked position).

Files generated by RIMDTA are loaded and run in exec mode by the block 0 loader described in Appendix 4.

TABLE 6-1 SUMMARY OF COMMON USER SERVICE PROGRAMS

Program	Size	Number of DEST. Files	Switches	Commands	Error Messages
EDITOR	1K	EDITOR operates upon one file (sequenced.)	There are no switches in the EDITOR Special delimitors: CR(CARRIAGE RETURN)=Ends a command. .(POINT)=Has the value of the last line number typed. ALT MODE=Special delineator for S; when used alone advances and prints the line.	*Typed when ready for a command. Sx control A (CR)=Select unit x and clear the directory. Sx, name control A ALT MODE=Same as above but creates file "NAME". Sx, name (CR)=Open file "NAME" on unit x. Sx, name ALT MODE=Create a new file called "NAME" on unit x. E(CR)=CLOSE THE CURRENT FILE. In1(CR)=Insert at line n1. In1,I2(CR)=As above, but set indexing increment to I2. Dn1(CR)=Delete line n1. Dn1, n2 (CR)=Delete line in n1 through n2. Pn1(CR)=Print line n1. Pn1,n2(CR)=Print lines n1 through n2. Rx(CR)=Resequence the current file. Rx,y(CR)=Resequence, stepping by y. ALT MODE=Return from text mode to command mode, or if in command mode, prints the next line.	"*ILC*" Illegal command. "*NLN*" A or D referenced a nonexistent line. "*ILS*" The index step (in insert) has caused a line already in the file to be skipped. "*ILR*" The index step has caused an already existing line to be referenced. "*UNA*" DECtape unit is not available. "*DCE*" DECtape directory full. "*NFO*" No file has been selected. "*FAU*" File name already in use. "*NCF*" Not a current file. "*DDE*" Device data error or write lock.
PIP1	1K Minimum (Uses all available storage)	1	A=ASCII line. B=Binary. F=FORTRAN format, one line per record. N=Delete sequence numbers. S=Sequence. W=Rewing mag tape. Z=Create fresh directory.	General Form: Destination ⟵ Source 1. Source 2, ... DTA1:Big Fil ⟵ PTR:,, (Copy 3 paper tape files onto DECtape). DTA1:Big Fil/S ⟵ PTR:,, (Same as above but add sequence no.). LPT: ⟵ DTA1:Big Fil (List the file). MTA1/W ⟵ (Rewind mag tape). DTA2:/Z ⟵ (Create a fresh directory).	"DEVICE name NOT AVAILABLE" "DIRECTORY FULL" "NO FILE NAMED file name" "OUTPUT DEVICE name DEVICE ERROR" "OUTPUT DEVICE name DATA ERROR" "OUTPUT DEVICE name IMPROPER MODE" "OUTPUT DEVICE name BLOCK TOO LARGE" "USE PIP2" "INPUT FILE file name DEVICE ERROR" etc.

TABLE 6-1 SUMMARY OF COMMON USER SERVICE PROGRAMS (continued)

Program	Size	Number of DEST. Files	Switches	Commands	Error Messages
PIP2	2K Minimum (Uses all available storage)	1	A=ASCII line. B=Binary. C=Card input suppress trailing spaces. Imbedded multiple spaces to tabs. D=Delete a file. F=FORTRAN format. L=List directory of source. N=Delete sequence numbers. R=Rename source file with name of dest. S=Add new sequence numbers. W=Rewind mag tape. X=Copy and clean up DECtape. Z=Create a fresh directory. T=Same as C, but without tabs.	General Form: Same as PIP1, but in Addition: DTA2:/X ⟵ DTA1: (Copy and clean up). DTA2:After/R ⟵ Before (Rename A file). DTA2:/D ⟵ File 1, File 2 (Delete two files). LPT:/L ⟵ DTA2: (List the directory).	Same as PIP1 and "DECTAPE INPUT REQUIRED"
FOLA	5K	2	N=No error listings on the console. S=List source only. M=Don't print symbol table. W=Rewind mag tape. Z=Clear DECtape directory. K=Advance forward mag tape one file.	General Form: Binary, Listing ⟵ Source 1, Source 2, ... PTP:, LPT: ⟵ PTR: DTA2: BINARY, DTA3: LIST ⟵ DTA1: FOLAIN ⟵ DTA3: FOLAIN (No output, errors only). PTP: ⟵ PTR: (Binary onto paper tape, no listing). ,LPT: ⟵ PTR: (No binary, listing on printer).	"??"=Command error "device name UNAVAILABLE" "DEVICE ERROR ON INPUT" "DATA ERROR ON INPUT" "IMPROPER INPUT MODE ON device name" "INPUT BLOCK TOO LARGE ON device name" "OUTPUT ERROR" "device name DIRECTORY FULL" "file name NOT IN DIRECTORY" "device name IS NOT A DEVICE" "device name CANNOT DO BINARY" "NO END STATEMENT ON FINAL PROGRAM"

TABLE 6-1 SUMMARY OF COMMON USER SERVICE PROGRAMS (continued)

Program	Size	Number of DEST. Files	Switches	Commands	Error Messages
FORTRAN	9K or 22K	2	T=Teletype or paper tape input. R=Read in rules (paper tape image assumed). For 22K version, all FOLA switches are recognized. For 9K version, only W, Z, and K.	22K compiler: Binary, Listing ⟵ Source Ex: DTA3:BINFIL, LPT: ⟵ DTA2:FORTIN ⟵ PTR:(T) (No output, errors only). 9K Compiler: Fola Input, Listing ⟵ Source Ex: DTA2:FOLAIN, DTA1: LIST ⟵ DTA5: FORTIN (Card image). DTA3:FOLAIN ⟵ PTR: (No listing, paper tape input). DTA3:FOLAIN ⟵ DTA1:PIMAGE(T) (Teletype image).	Same as FOLA.
MACRO	9K	2	S=Suppress listing. L=Reinstate listing. E=Expand macros on listing. X=Turn off expansion F=Fix symbol table. N=No error listings on the console. A=Advance forward mag tape one file. B=Backspace mag tape one file. T=Skip to logical end of tape. W=Rewind mag tape. Z=Clear DECtape directory.	General Form: Binary, Listing ⟵ Source 1, Source 2, ... PTP:, LPT:, ⟵ PTR:, , , (4 Input files). ⟵ DTA3: Macin (No output, errors on user TTY). DTA1: Bin Out ⟵ DTA2: Macin (No listing errors on TTY). , LPT: ⟵ MTA1: Assemb (No binary). DTA2: Bin Out, LPT: ⟵ (S)DTA6:MAC1, (L)PTR:, (No listing of file MAC1, two files from PTR).	"INPUT ERROR ON DEVICE X" "DATA ERROR ON DEVICE X" "NO END STATEMENT ENCOUNTERED ON INPUT FILE" "CANNOT FIND file name"

TABLE 6-1 SUMMARY OF COMMON USER SERVICE PROGRAMS (continued)

Program	Size	Number of DEST. Files	Switches	Commands	Errors Messages
LOADER	1K	1	S=Load local symbols. W=Do not load local symbols. L=Enter library search mode. N=Leave library search mode. I=Ignore all subsequent starting addresses. M=Print storage map. Z=Restart loading. ALT MOD=Finish loading. Enter L mode and load FORLIB if any global symbols are undefined. A=List all globals including those in zero length programs. D=Load DDT from SYS Tape. F=Leave S mode, enter L mode and load FORLIB from SYS Tape. Number G=Same as ALT MOD. If number #0, it is the starting address. J=Opposite of I. P=Prevent automatic library search for ALT MOD. Q=Opposite of P. U=List undefined globals only. X=Suppress listing globals in zero length programs. Normal mode is X, W, N, Q, J.	General Form: Memory Map ⟵ Switches, Source 1, ... LPT: ⟵ DTA3:BIN1(S),PTR:,DTA4:USRDDT (S)PTR: (Map on user TTY).	"ILL FORMAT file name" "device UNAVAILABLE" "INPUT ERROR file name" "ILL COMMON file name" "EXIT"= not enough core "char CHAR.ERROR IN LOADER COMMAND" "char SYNTAX ERROR IN LOADER COMMAND" "char SWITCH ERROR IN LOADER COMMAND" "n WORDS OF OVERLAP"= n words of extra core required "symbol n m MUL.DEF.GLOBAL file name" where n is the ignored value and m is the real value. "CANNOT FIND file name" "DIR FULL"

APPENDIX 1

MULTIPROGRAMMING EXECUTIVE COMPONENTS

This appendix contains a brief functional description of each of the components of the Multiprogramming Monitor. This section is intended to convey an idea of the size, function, and organization of the monitor, but not the complete details on methods of operation. Components sizes are subject to change without notice.

DIRECT JOB CONTROL ROUTINES

These routines are directly associated with processing a user's job. They handle the console communications, job scheduling, etc.

Program Name	Size (octal)	Function
SYSCON	252	Processes UUOs and illegal instructions. Dispatches to the appropriate subroutine for each UUO. Returns to the user for user's UUOs.
COMINI	135	Interprets console commands. Dispatches to system subroutines to perform the actions requested.
CORE	100	Processes all core allocation requests.
CLOCK	144	Handles processor interrupts. Calls the scheduling algorithm to schedule a new job. Dispatches to error routines for job error traps. Processes the clock queue.
RUNCSS	270	Controls the actual starting and stopping of jobs.
ERRCON	425	Transmits error messages to the user's console. Interprets error conditions.
IOCONT	477	Processes the standard input/output UUOs. SYSCON dispatches to this program.

EXECUTIVE DATA STORAGE

These routines provide data storage tables for the various system functions. These storage tables define the system configuration and (together with the accumulators) the state of the monitor program.

Program Name	Size (octal)	Function
IOINI1	443	Contains all system tables and temporary registers except those for the I/O modules. Also contains the priority interrupt processing subroutines.
IOINI2	2636 (DEC config)	Contains all tables for the I/O subroutine modules. The size of this program is highly dependent on system configuration.
JOBDAT	0	Defines the names of locations in the user's job data area.

INTERNAL EXECUTIVE SUBROUTINES

These programs contain subroutines and other useful functions that are called from other system programs.

CLKCSS	102	Performs the scheduling algorithm.
IOCSS	401	Contains subroutines that are used throughout the I/O service modules.
SYSCSS	35	Processes the miscellaneous system UUOs.
COMCON	272	Processes console commands. COMINI dispatches to this program.
COMCSS	234	Contains subroutines used in processing console commands.
SAVGET	164	Processes the two console commands, SAVE and GET.

SPECIAL PROGRAMS

The following programs perform system initialization and system debugging. They are not normally referenced during time-shared operations.

SYSINI	110	Initializes executive storage areas. Processes manual restarts. Initializes I/O modules.
ONCE	563	Processes the initial conversation in which the operator types in the day and the time.
SYSMAK	36	Moves the core image for job number one down on top of the monitor to the corresponding absolute locations. This program permits nonuser programs to be loaded by the time-sharing loader and to be run in exec mode.
DDT	3203	The executive debugging program. The space SYSMAK and DDT occupy may be used as additional user's program area.

Program Name	Size (octal)	Function

I/O SERVICE MODULES

The following programs process I/O service requests and device interrupts for the various devices that may be used during time-sharing.

SCNSER	1163	Controls the console Teletype and up to 64 lines on the Data Communications System Type 630, both full and half duplex. Processes console communications with the monitor.
DTSER2	1305	Controls up to eight DECtape transports on one DECtape Control Type 551. Interlocks with the magnetic tape service module for the use of the data control.
LPTSER	141	Controls one Line Printer Type 646.
CDRSER	244	Controls one Card Reader Type 461.
PTPSER	231	Controls one Paper Tape Punch Type 761.
PTRSER	140	Controls one Paper Tape Reader Type 760.
MTPSER	551	Controls up to eight magnetic tape transports on one Magnetic Tape Control Type 516. Interlocks with the DECtape service module for the use of the data control.

APPENDIX 2

MONITOR SCHEDULING ALGORITHM

The objective of the scheduling algorithm is to keep the I/O devices as busy as possible while still giving users good response for short computations. To do this the monitor keeps track of which jobs are runable and which are not. At least every 17 msec the monitor reviews the situation and decides whether or not to continue running the current job. This is done by keeping the runable jobs in a queue and serving them in a round robin manner with the head of the queue always being run. Whenever an unrunable job becomes runable it is placed at the front of the queue and is assigned a fixed quantum of time depending on the reason it became runable. It will then be run at the next scheduling period which can never be more than 17 msec away. Whenever the head of queue job exceeds its quantum, it is reassigned a somewhat longer quantum of 250 msec and is placed at the end of the queue.

Jobs may become runable for any of the following reasons:

1. The user typed a console command which started execution such as START, CONT, STARTC, CONTC, or DDT. He is given a 100-msec quantum.

2. The job just completed waiting for some of its own input/output to finish. It is given a 100-msec quantum.

3. The job just completed waiting for some other job's input/output to finish on a device which must be shared in a serial manner such as the magnetic tape control, the DECtape control, or the data control. The job is given a 70-msec quantum.

The effectiveness of a time-sharing system depends to a great extent on the program scheduling strategy used. Realizing this, the entire scheduling algorithm has been incorporated into a single separate subroutine. Thus, it may be changed easily to meet the needs of a particular computer installation.

The program name of this subroutine is CLKCSS and its only entry point is NXTJOB. Its sole function is to return the job number of the highest priority job at that instant. If no job is runable, it returns 0, the so-called null job which counts in AC0 and runs with the PC at 1. NXTJOB is called on the following occasions:

1. Every 60th of a second.

2. Whenever the current job is forced to wait for I/O or is stopped by the user at his console ('control' C).

3. Whenever any job becomes runable while the null job is running.

For further details consult the assembly listing of the subroutine itself.

APPENDIX 3

SYMBOL DEFINITIONS

OPDEF TAPE FOR EXEC SYSTEM

```
      OPDEF ENTER    [77B8]
      OPDEF LOOKUP   [76B8]
      OPDEF USETO    [75B8]
      OPDEF USETI    [74B8]
      OPDEF UGETF    [73B8]
      OPDEF MTAPE    [72B8]
      OPDEF RELEAS   [71B8]
      OPDEF CLOSE    [70B8]
      OPDEF OUTPUT   [67B8]
      OPDEF INPUT    [66B8]
      OPDEF OUTBUF   [65B8]
      OPDEF INBUF    [64B8]
      OPDEF STATZ    [63B8]
     *OPDEF STATUS   [62B8]
      OPDEF GETSTS   [62B8]
      OPDEF STATO    [61B8]
      OPDEF SETSTS   [60B8]
      OPDEF INIT     [41B8]
      OPDEF CALL     [40B8]
```

OPDEF'S FOR FORTRAN OPERATIONS SYSTEM

```
      OPDEF RESET.   [15B8]
      OPDEF IN.      [16B8]
      OPDEF OUT.     [17B8]
      OPDEF DATA.    [20B8]
      OPDEF FIN.     [21B8]
      OPDEF RTB.     [22B8]
      OPDEF WTB.     [23B8]
     **OPDEF REW.    [24B8]
      OPDEF SLIST.   [25B8]
      OPDEF INF.     [26B8]
      OPDEF OUTF.    [27B8]
      OPDEF RERED.   [30B8]
```

---

*Obsolete name

**The following UUOs have the same definition as REW.: REWUN., BSR., WEF., SPR., and WBR.

```
TITLE  JOBDAT - JOB DATA AREA ASSIGNMENTS
SUBTTL 8-2-65

        DEFINE BLK (A,B)
        <JOBPRO=B
        A=Ø
        ENTRY A
        DEFINE BLK (C,D)
        <C=JOBPRO
        JOBPRO=JOBPRO+D
        ENTRY C>>

        BLK JOBAC,2Ø     ;SYSTEM USE AC STORAGE
        BLK JOBDAC,2Ø    ;USER'S AC STORAGE WHILE JOB IS INACTIVE
JOBDHI=JOBDAC+17         ;HIGHEST AC SAVED WHILE INACTIVE
        BLK JOBUUO,1     ;UUO SAVE AND TRAP LOCATIONS
        BLK JOB41,1      ;UUO JSR LOCATION
        BLK JOBPC,1      ;BITS Ø-5=APR FLAGS, C(RH)=PC
        BLK JOBPDP,1     ;PUSH DOWN POINTER FOR SYSTEM USE
JOBPRT=JOBPDP            ;FIRST LOC PROTECTED FROM INTERRUPT SERVICE
JOBPR1=JOBPRT+1          ;FIRST LOC+1
        BLK JOBREL,1     ;C(LH)=Ø, C(RH)=RELOCATION ADDRESS
        BLK JOBUXT,1     ;EXIT FROM UUO AT BREAK
        BLK JOBLEV,1     ;C(4Ø) WHEN INACTIVE
        BLK JOBSAV,1     ;TEMPROARY STORAGE FOR UOO HANDLER
        BLK JOBPOV,1     ;PUSH DOWN OVERFLOW TRAP
        BLK JOBTEM,1     ;(UNUSED)TEMPORARY SYSTEM STORAGE
        BLK JOBTM1,1     ;"
        BLK JOBINF,1     ;UNUSED
        BLK JOBJDA,2Ø    ;JOB IO DEVICE CHANNEL ASSIGNMENTS
        BLK JOBDDT,1     ;ADDRESS OF USER'S DDT
JOBPFI=JOBDDT-1          ;LAST LOC PROTECTED FROM IO SERVICE
JOBPFU=JOBDDT            ;LAST LOC PROTECTED FROM USER
        BLK JOBCDP,1     ;(UNUSED)POINTER TO COMMAND STRING
        BLK JOBTRP,2Ø    ;(UNUSED)TRAP LOCATIONS
        BLK JOBSYM,1     ;SYMBOL TABLE POINTER
        BLK JOBUSY,1     ;(UNUSED)POINTER TO UNDEFINED SYMBOL TABLE
        BLK JOBSA,1      ;C(RH)=STARTING ADDRESS
                         ;C(LH)=HIGHEST LOCATION USED
        BLK JOBFF,1      ;FIRST FREE LOCATION
JOBDAT=14Ø       ;LENGTH OF JOB DATA AREA
JOBHGH=JOBDAT-1          ;HIGHEST LOC IN JOB DATA AREA
JOBPDL=JOBPRO-1          ;FIRST LOC. -1 OF PD LIST
JOBPD1=JOBPDL+1          ;FIRST LOC. IN PD LIST
MJOBPD=JOBPDL-JOBHGH     ;- LENGTH OF PD LIST
JOBS41=JOBPRO            ;PLACE WAHERE SAVE-GET SAVES USER LOC 41
JOBDA=JOBDAT
ENTRY JOBPFI,JOBPFU,JOBPDL,JOBPRO,MJOBPD,JOBHGH
ENTRY JOBDHI,JOBPRT,JOBPR1,JOBDA,JOBS41,JOBPD1

        END,
```

```
        XLIST


DEFINE  XP(A,B)         ;SYSTEM PARAMETER
        <A=B
        INTERNAL A
>
.ACCUMULATORS
        XP IOS,0;               IO STATUS WORD
        XP TAC,1;               TEMPORARY
        XP TAC1,2;              TEMPORARY
        XP PDP,3;               C(LH)=NUMBER OF LOCATIONS LEFT IN PD LIST
,                       C(RH)=PUSHDOWN POINTER
REPEAT 0,<      XP SUB,10               RETURN (PC) FOR ONE LEVEL SUBROUTINES
>
        XP TEM,10
        XP ITEM,4;      BYTE OR WORD POINTER,  ITEM COUNT
        XP DAT,5;               DATA OR TEMPORARY
        XP DEVDAT,6;    C(LH)=UNUSED,
,                       C(RH)=ADDRESS OF DEVICE DATA BLOCK
        XP JBUF,DAT;    ADDRESS OF JOB BUFFER AREA=JBFADR
        XP PROG,7;              C(LH)=UNUSED
,                       C(RH)=ADDRESS OF USER'S PROGRAM AREA
        XP JDAT,11;     C(RH)=ADDRESS OF JOB DATA
,ONLY 0 TO 10 NORMALLY REQUIRED
,FOR INTERRUPT SERVICE
        XP BUFPNT,12;   C(JBFADR 18-35)
        XP BUFWRD,13;   BIT 0=IOUSE
,                       BITS 1-17=BUFFER SIZE
,                       BITS 18-35=NEXT BUFFER ADDRESS
        XP UUO,14;      LAST UUO PROCESSED
        XP AC1,15       ;TEMPORARY ACS(MORE TEMPORARY
        XP AC2,16       ;THAN TAC,TAC1)
        XP AC3,17       ;
```

```
, DEVICE DATA BLOCK NAMES
        XP DEVNAM,0;    NAME IN SIXBIT ASCII
,                       C(LH)=DEVICE MNEMONIC
,                       C(RH)=DEVICE NUMBER, LEFT JUSTIFIED
        XP DEVCHR,1;    CHARACTERISTIC
,                       BITS 0-8=JOB NUMBER
,                            ZERO VALUE IMPLIES NOT ASSIGNED
,                       BITS 9-11=PI PRIORITY CHANNEL FOR DEVICE
        XP PICHN,100    ;RIGHTMOST BIT OF PI CHAN. FIELD
,                       BIT 12=UNUSED
,                       BIT 13=IORET. 0 IF DEVICE HAS A SHORT
,                           DISPATCH TABLE.  1 IF LONG
        XP IORET,20     ;FOR RETRIEVABLE DEVICES.
                ;SHORT DISPATCH TABLE =0, LONG = 1
,                       BITS 14-17=JOB DEVICE CHANNEL NUMBER
,                       BITS 18-23=DEVICE NUMBER, BINARY
,                       BITS 24-35=BUFFER SIZE
        XP DEVIOS,2;    STATUS WORD.  SEE BELOW
        XP DEVSER,3;    C(LH)=NEXT DEVICE DATA BLOCK
,                       C(RH)=DEVICE SERVICE DISPATCH TABLE
,    DEVICE SERVICE DISPATCH TABLE ASSIGNMENTS
        XP DRL,0;                   XXXDSP   :  RELEASE
        XP DCL,1;                   XXXDSP +1:  CLOSE
        ;IMMEDIATE ADDRESS PART OF CLOSE UUO
                XP CLSOUT,1     ;INHIBIT CLOSING OUTPUT
                XP CLSIN,2      ;INHIBIT CLOSING INPUT
        XP DOU,2;                   XXDSP +2:  OUTPUT
        XP DIN,3;                   XXXDSP +3:  INPUT;SHORT DISPATCH TABLE
        XP DEN,4;                   XXXDSP +4:  ENTER
        XP DLK,5;                   XXXDSP +5:  LOOKUP
        XP DDO,6;                   XXXDSP +6:  DMPO
        XP DDI,7;                   XXXDSP +7:  DMPI
        XP DSO,10;              XXXDSP+10:  SETO
        XP DSI,11;              XXXDSP+11:  SETI
        XP DGF,12;              XXXDSP+12:  GETF;LONG DISPATCH TABLE
        XP DEVMOD,4     ;BITS 6-11=LEFT HALF OF IMAGE MODE BYTE POINTER
                        ;BIT 35-J=1 IF MODE J IS LEGAL FOR THIS DEVICE
                        ;BIT 18 DEVICE ASSIGNED BY CONSOLE COMMAND
                        ;BIT 19 DEVICE ASSIGNED BY PROGRAM (INIT)
;IRGHT HALF OF DEVICE MODE WORD
        XP ASSCON,400000        ;ASSIGNED BY CONSOLE COMMAND ASSIGN
        XP ASSPRG,200000        ;ASSIGNED BY PROGRAM(INIT UUO)
;LEFT HALF DEVICE CHARACTERISTICS(DEVCHR UUO)
        XP DVOUT,1      ;OUTPUT DEVICE
        XP DVIN,2       ;INPUT DEVICE
        XP DVDIR,4      ;HAS A DIRECTORY
        XP DVTTY,10     ;IS A TTY
        XP DVMTA,20     ;IS A MAG TAPE(REWIND)
        XP DVAVAL,40    ;1 IF DEVICE IS AVAILABLE TO THIS JOB
                        ;SET BY DEVCHR UUO
        XP TTYUSE,10000 ;TTY DDB IN USE FLAG
        XP TTYATC,20000 ;TTY ATTACHED TO JOB IF 1
        XP DVLPT,40000  ;IS A LPT (CARRIAGE CONTROL IN FORTRAN)
        XP DVCDR,100000 ;IS A CARD READER(TRAILING SPACES FOR MACRO)
        XP DVDIRIN,400000       ;DIRECTORY IN CORE IF 1

        XP DEVLOG,5;    LOGICAL NAME FOR JOB DEVICE
        XP DEVBUF,6;    C(LH)=OUTPUT BUFFER AREA DDRESS
                        C(RH)=INPUT BUFFER AREA ADDRESS (JBFADR)
        XP DEVIAD,7;    C(LH)=PROG IN INDEX FIELD
```

```
XP DEVADR,DEVIAD
                C(RH)=CURRENT INPUT BUFFER ADDRESS
XP DEVOAD,10;   C(LH)=PROG IN INDEX FIELD
XP DEVPTR,DEVOAD
                C(RH)=CURRENT OUTPUT BUFFER ADDRESS
XP DEVCTR,11;
```

```
, I/0 STATUS WORD ASSIGNMENTS
,DATA MODES:  BITS 32-35
        XP A,0;          ASCII
        XP AL,1;                 ASCII LINE
        XP I,10;                 IMAGE
        XP IB,13;        IMAGE BINARY
        XP B,14;                 BINARY
        XP DR,16                 ;DUMP BY RECORDS
        XP D,17          ;DUMP ACROSS RECORDS
, STATUS BITS
,RIGHT HALF (USER)
        XP IOWC,20;      DON'T COMPUTE WORD COUNT
        XP IOCON,40;     CONTINUOUS (CONT=0)
        XP IORDEL,100;   READ AND DELETE
        XP IONRCK,100;   READ WITH NO. REREAD CHECK
        XP IODEND,20000;         DATA END ENCOUNTERED
        XP IODERR,200000;        DEVICE ERROR
        XP IODTER,100000;        DATA ERROR
        XP IOIMPM,400000;        IMPROPER MODE REQUESTED
        XP IOBKTL,40000;         BLOCK TOO LARGE
        XP IOACT,10000; DEVICE ACTIVE
  LEFT HALF (SYSTEM)
        XP IOW,1;                I/O WAIT
        XP IOBEG,2;      VIRGIN DEVICE
        XP IODISC,400000;        DISCONNECT REQUEST
        XP IOFST,4;      NEXT ITEM WILL BE THE FIRST ITEM OF A BUFFER
        XP IOSTRT,10;    IO READY TO START
        XP IO,20;                OUT=1, IN=0
        XP IOEND,40;     SERVICE ROUTINE HAS TRANSMITTED LAST DATA


;LEFT HALF USRJDA (JOB DEVICE ASSIGNMENTS) UUO'S FOR THIS CHANNEL SINCE LAST INIT
        XP INITB,400000;         INIT
        XP IBUFB,200000;         INIT WITH INPUT BUFFER SPECIFIED
        XP OBUFB,100000;         INIT WITH OUTPUT BUFFER SPECIFIED
        XP LOOKB,40000; LOOKUP
        XP ENTRB,20000; ENTER
        XP INPB,10000;  INPUT
        XP OUTPB,4000;  OUTPUT
        XP CLOSB,2000;  CLOSE
                                 ;RELEASE CLEARS THEM ALL
;CLOSE UUO BITS
        XP CLSOUT,1      ;INHIBIT CLOSING OUTPUT
        XP CLSIN,2       ;INHIBIT CLOSING INPUT
```

```
; JOB BUFFER AREA HEADER
        XP JBFADR,0;     BIT 0=1 IF THIS BUFFER RING HAS NEVER BEEN
;                            REFERENCED FROM THE USER'S PROGRAM BY
;                            AN INPUT OR OUTPUT COMMAND.
;                        BITS 1-17=UNUSED
;                        BITS 18-35=CURRENT BUFFER ADDRESS
        XP JBFPTR,1;     BYTE POINTER TO NEXT BYTE -1
        XP JBFCTR,2;     POSITIVE ITEM COUNT
; JOB BUFFER HEADER              BIT 0=IOUSE
        XP IOUSE,400000;                1 IF BUFFER IS FULL (OR BEING EMPTIED)
;                            0 IF BUFFER IS EMPTY (OR BEING FILLED)
;                       BITS 1-17=BUFFER SIZE
;                       BITS 18-35=NEXT BUFFER ADDRESS
```

```
.SYSTEM PARAMETERS
        XP USRMOD,10000;        USER MODE BIT IN ARP FLAGS, PC WORD
        XP IODONE,400000;
.JOB STATUS WORD (JBTSTS)
        XP IOWS,400000; IO WAIT SATISFIED
        XP RUN,200000;  READY TO RUN OR IS RUNNING
        XP JIOW,100000; JOB IN IO-WAIT STATE
        XP DTW,40000;   DECTAPE CONTROL WAIT
        XP DCW,20000;   DATA CONTROL WAIT
        XP MTW,10000;   MAG TAPE CONTROL WAIT
        XP JNA,4000;    JOB NUMBER ASSIGNED
        XP JERR,2000;   JOB ERROR(ILLEGAL MEM. ETC.)

        XP RUNABLE,RUN+JNA;     STATUS BIT PATTERN FOR A JOB TO STAY RUNABLE
```

```
;SPECIAL ABSOLUTE LOCATIONS IN LOWER MEMORY
;THESE ARE INTENDED TO BE EXAMINED AND CHANGED BY
;USING THE DATA SWITCHES BEFORE SYSTEM IS STARTED AFTER BEING LOADED
;ONLY TODAY AND BCDTIM NEED BE SET
;ALL OTHERS ARE ASSEMBLED IN AND SHOULD BE SET ONLY IF A CHANGE IS DESIRED

XP DDTMEM,37     ;MEMORY SIZE FOR SYSTEM DDT AND SYSMAK ONLY
XP DDTSYM,36     ;SYSTEM DDT SYSBOL TABLE POINTER (USED TO BE 77)
, SYSTEM MACROS
DEFINE  ADRCHK (A)
<       EXTERNAL ADRCK
DEFINE  ADRCHK (B)
         <PUSHJ PDP,ADRCK
         HRRZ TAC,B>
         PUSHJ PDP,ADRCK
         HRRZ TAC,A>


        LIST
```

APPENDIX 4

MONITOR OPERATING INSTRUCTIONS

PROCEDURE FOR LOADING THE MULTIPROGRAMMING MONITOR

1. Turn on the computer (POWER switch on console). This should also send power to all of the I/O devices.

2. Check to see that the power is turned on for all I/O devices.

3. Place DECtape copy of monitor on drive 0. (Set the DECtape unit selection switch to 8 to get unit 0). Make sure no other units are set to 8.

4. If machine does not have fast accumulators:

   a. Assume that the 10 instruction Read-in Mode Loader is still in locations 20 through 27. (It usually is.)

   b. Depress the READER OFF lever switch.

   c. Place a RIM paper tape copy of the DECtape RIM Loader in the reader. (See DEC 06 L DTRL UML FP ACT00 Shadow Mode Loader for a listing of this 14-instruction program.)

   d. Turn on the reader.

   e. Set the ADDRESS switches to 20 and depress the INSTRUCTION STOP, I/O RESET, and START lever switches in that order. The 10 instruction Read-in Mode Loader will load the 14-instruction DECtape RIM Loader into locations 0 through 13 and then stop.

   f. Depress the INSTRUCTION CONTINUE lever switch. The DECtape RIM Loader will then load the DECtape Block 0 Loader from block 0 of unit 0 into locations 37400-37777.

   g. If the paper tape does not move when the START lever switch is depressed, the assumption that the 10 instruction Read-in Mode Loader was intact proved

to be valid. It must be keyed into locations 20 through 27 using the DATA and ADDRESS switches and the DEPOSIT lever switch. The Read-in Mode Loader is:

```
20/     710600  000060      CONO PTR, 60
21/     710740  000010      CONSO PTR, 10
22/     254000  000021      JRST .-1
23/     710440  000026      DATAI PTR, 26
24/     710740  000010      CONSO PTR, 10
25/     254000  000024      JRST .-1
26/     irrelevant
27/     254000  000021      JRST 21
```

Go back to step 4a.

5. If machine has fast accumulators:

   a. Assume that the DECtape RIM Loader is still in locations 0 through 13 of shadow mode memory. (It usually is.)

   b. Set the ADDRESS switches to 0, depress the INSTRUCTION STOP and lift THE START lever switch.

   c. The 14-instruction DECtape RIM Loader will then load the DECtape Block 0 Loader from block 0 of unit 0.

   d. If block 0 is not loaded, the assumption that the DECtape RIM Loader was intact proved invalid. To restore it to locations 0 through 13 of shadow mode memory, flip on the RIM MAINTENANCE switch in bay 2, so that the RIM SUBR light on the panel on bay 1 is on.

   e. Load the DECtape RIM Loader from paper tape into locations 0 through 13. See steps 4a through 4e. Do not press INSTRUCTION CONTINUE (step f).

   f. Turn the RIM MAINTENANCE switch off so that the RIM SUBR light is off. Go back to step 5a.

6. Instruct the DECtape block 0 loader to load the dump file which contains the monitor by typing the file name (name of the installation) followed by carriage-return on the console Teletype. The file directory on block 1 will be searched, and the specified dump file will be loaded into core. The dump file must be a program less than 16K. If the dump file is not in the directory, a bell will be typed to indicate the error.

7. Line-feed will be typed out after the file has been loaded. Now set the ADDRESS lever switches to 140 and depress the INSTRUCTION STOP, I/O RESET, and START lever switches in that order.

8. The monitor will then enter a once-only dialogue with the operator on the console Teletype (CTY). The monitor will request various pieces of information like the date and time of day. The operator must type in this information followed by a carriage-return. A line-feed will be echoed. Rubouts work as in time-sharing and may be used to delete erroneous typing. Typing in carriage-return while monitor is typing out will terminate the line, thereby speeding up the startup procedure.

9. The once-only dialogue looks like the following with output indicated by *:

```
*DEC 1.4 10-13-65 MONITOR JUST LOADED
*TYPE TODAY'S DATE AS ABOVE.
 (operator types date)
*TYPE 4 DIGIT TIME.
 (operator types time)
*I/O CONFIGURATION
*10 TTY'S
*8 DTA'S
 etc.
*DO YOU WANT SYSMAK (TYPE Y IF YES, CR IF NO)?
 (operator types Y or CR)
*EXEC DDT?
 (operator types Y or CR)
*EXEC IS 13573 OCTAL LOCATIONS LONG.
```

10. The monitor sets its system data storage to 0.

11. The monitor examines all 256K of memory to find which are nonexistent. A memory with the power turned off will appear nonexistent and will not be used by the monitor.

12. All I/O devices are initialized

13. The PI channels are turned on and time-sharing may begin. Whenever the monitor has nothing to do, it will run the null job. The null job counts in AC0 and runs in AC1. Therefore, it can be detected by a 1 appearing in the PROGRAM COUNTER lights while the machine is in EXEC mode. The count in AC0 may be displayed by setting the ADDRESS switches to 0. The count is reset every time the null job begins to run, so that it can be used to estimate the system idle time.

14. The operator types DEASSIGN OPR which initializes job 1 and deassigns his Teletype. The operator's Teletype is given the name OPR and is defined as the first Teletype on which a character is typed in after the PI is turned on. Usually it is CTY.

15. The operator mounts the CUSP tape on drive 0 and sets it to WRITE LOCK.

16. The operator mounts console scratch tapes on all other drives, and sets them to WRITE. Make sure that all units are dialed to unique numbers starting at 0 (8). The system will hang up if a user attempts to use a unit on which no tape is mounted or the power is off.

17. Remote users may then communicate with the operator by using PIP1 in the following manner:

```
CORE 1
GET SYS:PIP1
START
OPR: ← TTY:
the message
<control> Z
```

The message should begin with a few bell characters to attract the attention of the operator. To receive an acknowledgement the message should end with GA (go ahead). Then the user should type the following PIP command string:

```
TTY: ← OPR:
```

The operator will not need to know from which Teletype the message was typed.

This interconsole message communication is most useful to request the operator to mount a tape on a drive previously ASSIGNED to the user by the ASSIGN command. It is also useful for requesting a tape to be set to WRITE or WRITE-LOCK. Example of a typical sequence of commands for remote users. Input is indicated by underline.

User's Console	Operator's Console
	DEASSIGN OPR
	JOB 1 INITIALIZED
	DEC 1.4H
ASSIGN DTA	
DEVICE DTA1 ASSIGNED	
JOB 2 INITIALIZED	
DEC 1.4H	
CORE 1	

User's Console	Operator's Console
GET SYS:PIP1	
JOB SETUP	
START	
OPR:←TTY:	
<bell><bell><bell>	
PLEASE MOUNT TAPE 39 ON UNIT 1	
SET TO WRITE GA	
↑Z	
	PLEASE MOUNT TAPE 39 ON UNIT 1
	SET TO WRITE GA
TTY:←OPR:	
	TAPE MOUNTED.
	↑Z
TAPE MOUNTED.	

DECtape Colored Label Conventions

Yellow	Console Scratch
Green	System tapes CUSP + Monitor
Red	Private tapes

ERROR RECOVERY PROCEDURES

Occasionally the system may hangup. If this happens, a software trouble report should be filled out and sent to Digital Equipment Corporation, Maynard, Mass., attention: PDP-6 Librarian. Prompt replies are sent back regardless of whether the trouble is hardware or software.

The error recovery procedure consists of restarting the monitor which will reinitialize part of itself before resuming operation. There are four restart procedures which reinitialize an increasing amount of the monitor.

1. To recover from a priority interrupt which will not dismiss (PI in PROGRESS lights on continuously):

   a. Press STOP lever switch.
   b. Fill out software trouble report.
   c. Set ADDRESS switches to 144.
   d. Press START lever switch.

This action tries to dismiss interrupt and start running null job. This affects at most one job which the user will have to restart.

2. To recover from hung I/O devices, wait until all users have finished their immediate operations. For example, all users editing should type the E editor command.

   a. Press STOP lever switch.
   b. Fill out software trouble report.
   c. Press I/O RESET lever switch.
   d. Set ADDRESS switches to 140.
   e. Press START lever switch.

This will not reenter once-only dialogue. It will not DEASSIGN any devices and will not return core to the system. All Teletypes will be in command mode, and each user must type the START monitor command to restart his job.

3. To reinitialize the monitor to the state it was in just after being loaded:

   a. Press STOP lever switch.
   b. Fill out software trouble report.
   c. Press I/O RESET lever switch.
   d. Set ADDRESS switches to 143.
   e. Press START lever switch.

Step 10, 11, and 12 of the monitor loading procedure must be repeated and all jobs reloaded.

4. To recover in case the monitor itself has been destroyed:

   a. Press STOP lever switch.
   b. Fill out software trouble report.
   c. Press I/O RESET lever switch.
   d. Go back to step 1 of monitor loading procedure.

LIST OF STARTING LOCATIONS IN MONITOR

140 First time - Once-only dialogue, initializes system data storage and I/O devices and moves EXEC DDT's symbol table near to the top of existant memory but leaving the last 200 locations free.

Succeeding Times - initializes I/O devices only.

141 Enters EXEC DDT (3000 octal locations) if the operator has indicated that he wants it in the once-only code. This is used to debug the monitor and is useful for installations making modifications to the monitor for their own special purpose.

142 Enters SYSMAK routine (36 locations) which moves job 1 down on top of the monitor and moves the symbol table to top of memory as set in absolute location 37. This routine is useful to load a program which is to replace the monitor, either a new monitor or a nontime-sharing program.

143 Always examines memories to see which exist, resets system data storage, and kills all jobs.

144 Dismisses highest interrupt in progress and runs null job.

145 Reenters once-only dialogue. This may not be possible since the lowest job may be placed on top of part of the once-only code.

146 Starts up system for first time bypassing the once-only code. It is assumed that SYSMAK and EXEC DDT are not needed. This is useful if the console Teletype is not working.

Monitor Storage Layout

monitor itself
SYSMAK
EXEC DDT
once-only dialogue code

user jobs may overlay SYSMAK, EXEC DDT, and once-only dialogue code

APPENDIX 5

EXAMPLES OF USING A USER'S CONSOLE

Underlined text is typed by the user.

```
IJOB
JOB 3 INITIALIZED

CORE 1

GET SYS PIP1
JOB SETUP

START


MTA0:/W←                                    (rewind mag tape)

MTA0:←PTR:/S

MTA0:/W←

TTY:←MTA0:
00010
00020    THIS IS A DEMONSTRATION FILE.
00030    THESE LINES WERE READ IN FROM THE PAPER TAPE READER.
00040    HERE IS THE LAST LINE OF THE FILE.
00050
00060
↑C                                          ("CTRL" key and "C" pressed simultaneously)
ASSIGN MTA0 WXYZ
DEVICE MTA0 ASSIGNED

START


MTA0:/W←

TTY:/N←WXYZ:

THIS IS A DEMONSTRATION FILE.
THESE LINES WERE READ IN FROM THE PAPER TAPE READER.
HERE IS THE LAST LINE OF THE FILE.


↑C
CORE 10
8 FREE 1K BLOCKS LEFT, NONE ASSIGNED

KJOB
```

APPENDIX 6

PDP-6 FORTRAN II COMPILER OPERATING INSTRUCTIONS

The PDP-6 FORTRAN II Compiler contains two basic sections: a compiler which generates assembly code from the FORTRAN source statements and an assembler which generates relocatable binary programs. The 22K compiler contains both parts in one program. The 9K compiler, however, prepares an intermediate file for input to the assembler which is a separate program (FOLA).

COMMAND STRING FOR 22K COMPILER

The command string is used to specify the input and output file designations for the compiler. The 22K compiler expects up to two output files and one input file. The general form of the command string is

FILE1, FILE2←FILE3

FILE1 will contain the relocatable binary output; FILE2 will contain the listing of the compiled output (source, assembly, binary, errors), and FILE3 is the source or input file. Each file may have one of the following forms:

DEVICE:
DEVICE: FILENAME
DEVICE: FILENAME.EXTENSION

where DEVICE may be any device mnemonic acceptable to the PDP-6 executive system, FILENAME may be up to six letters and/or digits, and EXTENSION may be up to three letters and/or digits.

The file name extensions REL and LST are assumed for FILE1 and FILE2 unless specified otherwise.

Example:

PTP: ,DTA3:LIST←DTA1:SORC.TXT

If FILE1 is not desired, the command string should be of the form:

,FILE2←FILE3

If FILE2 is not desired, either of the following command strings is valid:

FILE1←FILE3
FILE1,←FILE3

If neither output file is desired, the valid command strings are:

←FILE3
,←FILE3

SWITCHES FOR THE 22K COMPILER

Switches are letters specifying optional and extra functions to be performed by the compiler. These letters may appear within parentheses or after a forward slash. Only a single letter may follow a slash, while more than one letter may appear within parentheses. The switches are as follows:

- K Skip one file on the device (magnetic tape only).
- M Do not print storage map.
- N Do not list errors on Teletype console if there is a listing file.

APPENDIX 7

LIMITATIONS ON 9K FORTRAN II COMPILER

1. Boolean statements are not allowed.
2. Use of * in IF statements is not allowed.

MULTIPROGRAMMING MANUAL INDEX

For the convenience of the reader, the letters 'A', 'B', and 'C' in the Index are arbitrary references to the top, middle, and bottom sections, respectively, of the pages.

EQUIPMENT
CORPORATION

MAYNARD, MASSACHUSETTS

9087

PRINTED IN U.S.A.

3-4/66

DEC-6-0-EX-SYS-UM-IP-PREOO

MULTIPROGRAMMING SYSTEM MANUAL

PDP-6

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

M U L T I P R O G R A M M I N G T I M E S H A R I N G S Y S T E M

M A N U A L I N D E X

CHAPTER IV - USER INPUT OUTPUT COMMANDS

CONSOLE STATE DIAGRAM

COMMON USER SERVICE PROGRAM (CUSP)

JOB AREA STORAGE

SYSTEM DIAGRAM

I/O SERVICE BLOCK DIAGRAM (SIMPLIFIED)

I/O SERVICE BLOCK DIAGRAM

CHAPTER I

INTRODUCTION TO THE PDP-6 MULTIPROGRAMMING TIME SHARING SYSTEM

THE PROGRAMMED DATA PROCESSOR 6 TIME SHARING SYSTEMS ARE DESIGNED TO PROVIDE INTEGRATED SOFTWARE AND HARDWARE FOR THE SIMULTANEOUS USE OF A SINGLE SYSTEM BY MANY USERS AT CONSOLES. A CONSOLE CONSISTS OF A TELETYPEWRITER WHICH IS USED TO COMMUNICATE WITH THE SYSTEM. A USER MAY ALSO HAVE REQUESTED FACILITIES OR PROGRAMS. USER'S FILES OR DATA ARE KEPT SEPARATELY ON A DECTAPE FOR EACH USER, A SEGMENT OF THE DRUM OR DISC. THESE FILES ARE REQUIRED WHEN PERMANENT STORAGE OF RESULTS IS DESIRED.

A USER MAY HAVE ONE OR MORE ACTIVE JOBS. A JOB IS CONSOLE INITIATED AND RUNS IN AN ASSIGNED AND PROTECTED MEMORY AREA.

DESIGN OBJECTIVES

THE BASIC SYSTEM OBJECTIVES ARE TO PROVIDE A SIMPLE, MODULAR, TIME SHARING SYSTEM.

CONSOLE USE OF THE SYSTEM FOR PROGRAMMING
A USER MAY CREATE AND EDIT A PROGRAM (OR TEXT), TRANSLATE THE PROGRAM TO MACHINE LANGUAGE (USING FORTRAN, MACRO, OR SOME OTHER) IN THE FORM OF RELOCATABLE LINKING BINARY FOR THE LINKING LOADER. LOAD AND RUN THE PROGRAM, OR IF NECESSARY DIRECTLY COMMUNICATE WITH THE PROGRAM THROUGH A DEBUGGING LANGUAGE.

CONSOLE USE OF THE COMMON USER SERVICE PROGRAMS
THE ABILITY OF A TIME SHARED SYSTEM IS NEARLY OPEN ENDED SINCE FACILITIES ARE IMMEDIATELY AVAILABLE TO ALL USERS WHEN A COMMON USER SERVICE PROGRAM (CUSP) IS ADDED. THIS CLASS OF PROGRAMS INCLUDES FILE MANIPULATION, EDITING, SPECIAL DESK CALCULATING, SIMULATION, AND USER PROGRAMS (UP), OF GENERAL INTEREST.

CONSOLE USE OF THE COMMON USER SERVICE PROGRAMS FOR DATA CONVERSION
THE PERIPHERAL INTERCHANGE PROGRAMS ACCOMPLISH GENERAL DATA CONVERSION FROM MEDIA TO MEDIA, EG. CARD TO TAPE, TAPE TO PRINTER, ETC.

UNATTENDED STACK OR BATCH PROCESSING OF JOBS
A STACK IS RUN AS THOUGH IT WERE CONSOLE CONTROLLED. JOBS ARE STACKED IN A CARD READER OR OTHER INPUT DEVICE AND PROCESSED IN SEQUENCE. THE CONTROL LANGUAGE (CONTROL STATEMENTS OR CARDS) IS IDENTICAL TO THAT OF A CONSOLE USER, AND THE CONTROL STATEMENTS ARE PLACED WITHIN A SPECIAL FILE.

OPEN-ENDED SYSTEM WHICH CAN SERVE SPECIAL CONSOLES
THE EXECUTIVE AND IO SUBROUTINES MAY BE MODIFIED SO THAT CONSOLES REQUIRING A SPECIAL SERVICE OR LANGUAGE MAY BE INCLUDED.

OPEN-ENDED SYSTEM WHICH CAN SERVE REAL TIME PROCESSES
THE NECESSARY INPUT/OUTPUT ROUTINES TO CONNECT THE SPECIAL DEVICE WITH THE PROGRAMMING SYSTEM WOULD BE ADDED. A USER OR JOB WOULD ISSUE SYSTEM INPUT/OUTPUT COMMANDS FOR HIS SPECIAL DEVICE IN THE

SAME MANNER AS FOR CONVENTIONAL DEVICES. IN SOME SPECIAL APPLICATIONS, IT MAY BE NECESSARY TO MODIFY THE SCHEDULER PORTION OF THE MONITOR THAT THE OCCURRENCE OF I/O DEVICE CONDITIONS WOULD CAUSE SPECIAL SERVICE FOR THE USER.

COMMON I/O SUB-PROGRAMS

THE I/O ROUTINES PROVIDE A BASIC SET OF DEVICE INDEPENDENT INPUT/OUTPUT COMMANDS WHICH MAY BE GIVEN TO ALL DEVICES. THE SYSTEM CONTROLS THE ASSIGNMENT, INITIALIZATION, AND CLOSING FOR ALL I/O DEVICES, AND THE RENAMING FOR A USER IF NECESSARY. PROGRAMS MAY BE EASILY WRITTEN WHICH COMMUNICATE WITH ANY I/O DEVICE+ ALL I/O PROGRAMS USE THE HARDWARE PRIORITY INTERRUPT SO THAT A DEVICE'S NULL TRANSMISSION DOES NOT REQUIRE THE SYSTEM TO WAIT.

COMMON SYSTEM SUB-PROGRAMS

THESE SYSTEM ROUTINES ARE CALLED FOR SPECIAL SYSTEM FUNCTIONS, SUCH AS TIME, DATE, EXTENSION OF CORE ALLOTTMENT, ETC.

TIME SHARING SYSTEM STAGES

THE PDP-6 TIME SHARING SYSTEMS ARE AVAILABLE IN FOUR CONFIGURATIONS. THESE ARE DISTINGUISHED BY VARIOUS HARDWARE-SOFTWARE CONFIGURATIONS. THESE ALLOW A DIFFERENT BACKUP STORAGE MEDIUM TO STORE PROGRAMS WHILE OTHERS ARE RUNNING, AND THIS DETERMINES THE NUMBER OF SIMULTANEOUS ACTIVE SYSTEM USERS AND THE SYSTEM RESPONSE TIME TO A USER.

SYSTEM I - MULTIPROGRAMMING TIME SHARING

THE MULTI-PROGRAMMING TIME SHARING SYSTEM USES MULTI-PROGRAMMING TO PROVIDE SIMULTANEOUS SERVICE FOR THE USER9 NO BACKUP STORAGE IS USED TO STORE PROGRAMS AND ACTIVE PROGRAMS MUST RESIDE IN CORE MEMORY. DECTAPE IS USED TO RETAIN ALL THE LIBRARY PROGRAMS, AND EACH USER'S INACTIVE PROGRAMS. MULTI-PROGRAMMING MEANS THE SIMULTANEOUS EXISTENCE OF MULTIPLE PROGRAMS IN CORE MEMORY, WITH A CONTROL MECHANISM TO SWITCH FROM PROGRAM TO PROGRAM AS THE ACTIVE PROGRAMS COME TO AN INPUT/OUTPUT WAITING CONDITION, OR COMPLETE THEIR ALLOTTED SHARE OF COMPUTATION TIME.

SYSTEM II - DRUM TIME SHARING

THE DRUM TIME SHARING SYSTEM USES THE PDP-6 ONE MILLION WORD DRUM TO PROVIDE THE BACKUP STORAGE FOR MULTIPLE USERS. BOTH MULTI-PROGRAMMING AND PROGRAM SWAPPING TECHNIQUES FOR TIME SHARING CONTROL ARE USED TO PROVIDE SERVICE TO MANY ACTIVE USERS. WITH PROGRAM SWAPPING, A PROGRAM IS BROUGHT INTO CORE MEMORY AND RUN WHEN ITS TURN FOR COMPUTATION IS AVAILABLE8 OR INPUT/OUTPUT ASSOCIATED WITH IT IS COMPLETE, AND THE PROGRAM IS NOT IN CORE. A PROGRAM IS RETURNED TO THE DRUM WHEN IT MUST WAIT FOR INPUT/OUTPUT, OR HAS FINISHED ITS SHARE OF COMPUTING, AND SPACE IN CORE IS REQUIRED. AS IN SYSTEM I, DECTAPE IS USED TO STORE USER'S INACTIVE PROGRAMS, AND A USER BEGINS BY FIRST READING HIS DECTAPE FILES UNTO THE DRUM. AT THE COMPLETION OF CONSOLE USE, THE USER'S FILES ON THE DRUM MAY BE COPIED TO A DECTAPE, AND WITH THIS MODE OF OPERATION, AUTOMATIC DAY BY DAY BACKUP IS PROVIDED.

SYSTEM III - DRUM-DISC TIME SHARING

THE DRUM-DISC TIME SHARING SYSTEM REQUIRES THE ADDITION OF A DISCFILE WITH AT LEAST FIVE MILLION WORDS OF STORAGE, AND A TYPE 516 MAGNETIC TAPE SYSTEM. THE DISCFILE IS USED AS A LONG TERM STORAGE FOR INACTIVE FILES OR PROGRAMS. WITH THIS SYSTEM A USER MAY REQUEST A DUPLICATE OF A FILE ON THE DISC TO BE PLACED

ON THE DRUM, OR HE MAY WORK WITH FILES DIRECTLY ON EITHER THE DRUM OR DISC. THE DRUM RETAINS COPIES OF PROGRAMS BEING SWAPPED. THE MAGNETIC TAPE SYSTEM IS REQUIRED FOR DAY TO DAY BACKUP IN CASE OF SYSTEM FAILURES.

SYSTEM IV - MULTI-PROCESSOR TIME SHARING

THE MULTIPROCESSOR SYSTEM GIVES ADDITIONAL COMPUTATION POWER BY ALLOWING TWO ADDITIONAL TYPE 166 ARITHMETIC PROCESSORS TO BE CONNECTED TO THE SYSTEM.

HARDWARE FOR TIME SHARING

EXECUTIVE MODE HARDWARE

WHEN UN-DEBUGGED PROGRAMS SHARE MEMORY WITH OTHER PROGRAMS, PROTECTION MUST BE PROVIDED AGAINST ILLEGAL REFERENCES TO OTHER AREAS OF MEMORY. TO PROVIDE THIS PROTECTION, PROGRAMS PREPARED AND RUN BY THE USER ARE RUN IN THE USER MODE. THEY ARE PLACED IN MEMORY, AND RELOCATED, BY THE RELOCATION REGISTER. MEMORY REFERENCES OUTSIDE THE AREA SPECIFIED BY THE MEMORY PROTECTION REGISTER ARE ILLEGAL. AN ILLEGAL MEMORY REFERENCE, AN ATTEMPT TO STOP THE MACHINE, OR AN I/O INSTRUCTION, CALL THE EXECUTIVE.

LARGE STORAGE

SINCE A NUMBER OF USERS ARE TO OCCUPY MEMORY AT ONCE, A TIME SHARING SYSTEM MUST HAVE A LARGE CORE MEMORY AND/OR THE ABILITY FOR A FAST INTERCHANGE BETWEEN CORE MEMORY AND AUXILIARY FILE STORES, TO PROVIDE MULTI-PROGRAMMING SERVICE.

A PROCESSOR IN A PDP-6 SYSTEM MAY DIRECTLY ADDRESS UP TO 262,144-36 BIT WORDS OF CORE STORAGE IN MEMORY MODULES OF VARYING SIZE AND SPEED. THE POWERFUL INSTRUCTION SET ALSO PROVIDES FOR ADDRESSING THE SAME STORAGE AS CHARACTERS, PROVIDING A STORE OF UP TO 1.6 MILLION CHARACTERS. MAGNETIC DRUMS, DISC, OR DECTAPE CAN BE CONNECTED FOR AUXILIARY FILE STORAGE AND FOR PROGRAM SWAPPING.

DATA REPRESENTATION

TO AVOID UNNECESSARY CONVERSION PROGRAMS ALL DATA IN THE PDP-6 IS REPRESENTED BY THE AMERICAN STANDARDS ASSOCIATION INFORMATION INTERCHANGE [ASCII] 7 BIT CODE. LINE PRINTER AND TELEPRINTER CODES ARE ASCII, AND OTHER CODES ARE CONVERTED TO ASCII AS THEY ENTER AND LEAVE THE SYSTEM.

PRIORITY INTERRUPT FOR IO DEVICES

THE 7 LEVEL PRIORITY INTERRUPT SYSTEM ALLOWS ALL I/O DEVICES OF A SYSTEM TO OPERATE CONCURRENTLY. EACH PRIORITY LEVEL MAY HAVE A NUMBER OF DEVICES CONNECTED TO IT WHICH REQUEST SERVICE. HIGHEST LEVEL CHANNELS GENERALLY CONNECT WITH ONLY ONE MEDIUM SPEED DEVICE FOR DATA TRANSMISSION [E.G. DISC, MAGNETIC TAPE]. LOWER SPEED DEVICES CONNECT TO LOW PRIORITY CHANNELS AND THE EXECUTIVE SYSTEM MAY POLL MULTIPLE DEVICE REQUESTS OCCURRING ON A SINGLE CHANNEL.

TABLE OF MULTIPROGRAMMING SYSTEM HARDWARE

REQUIRED HARDWARE	
1-166	ARITHMETIC PROCESSOR WITH 626 TELEPRINTER
1-161 OR 163C	(16,384 MEMORY WORDS)
1-136	DATA CONTROL
1-551	DECTAPE CONTROL
1-555	DUAL DECTAPE (
1-760	PAPER TAPE READER (HARDWARE MAINTENANCE)
OPTIONAL HARDWARE	
1-162	16 WORD MEMORY
OPTIONAL HARDWARE WITH SOFTWARE SUPPORT	
N-161 OR 163C	MEMORIES (FORMING UP TO 262,144 WORDS)
3-555	DUAL DECTAPES (1-555/2 USERS)
1-630	DATA COMMUNICATIONS SYSTEM WITH HALF OR FULL DUPLEX 8 LEVEL TELETYPEWRITTERS
1-516	MAG TAPE CONTROL WITH 520,521,522 INTERFACE AND UP TO 8 TRANSPORTS.
1-346	DISPLAY
1-461	CARD READER
1-646	LINE PRINTER (300,600,1000 LPM)

SOFTWARE MONITOR COMPONENTS

THE MULTIPROGRAMMING SYSTEM DIAGRAM DEPICTS THE FLOW OF DATA AT THE USER CONSOLE TO EXTERNAL FILES, AND THE SOFTWARE-HARDWARE INTERACTION.

CONSOLES

THE USER'S CONSOLE IS HIS COMMUNICATION WITH THE SYSTEM, AND HIS FILES. ALL COMMANDS TO INITIATE, CARRY-OUT, AND TERMINATE A JOB ARE AT THE CONSOLE. EACH TIME A SYSTEM PROGRAM, OR SERVICE IS REQUESTED, THIS COMMUNICATION IS CARRIED OUT AT THE CONSOLE. THE CONSOLE IS ALSO THE MEDIA THROUGH WHICH A USER REQUESTS PERSONAL PROGRAMS, DATA FILES, ETC. CONTAINED IN THE SYSTEM TO BE MANIPULATED.

THE TELETYPE DATA IS TRANSMITTED ON A STANDARD TELEGRAPH LINE TO THE DATA COMMUNICATIONS EQUIPMENT AT THE COMPUTER. UP TO 64 LINES ENTER A SINGLE DATA COMMUNICATION DEVICE, AND THE CHARACTERS ARE PROCESSED IN SEQUENCE AS THEY ARRIVE. THE CHARACTER IS TRANSMITTED TO A USER'S OWN BUFFER AREA THROUGH THE TELETYPE IO SERVICE ROUTINE.

AS INITIAL CHARACTERS ARE RECEIVED, THE I/O SERVICE ROUTINE FIRST LOOKS TO SEE IF A BUFFER AREA HAS BEEN ASSIGNED TO THE USER STATION CAUSING THE INTERRUPT. IF NO BUFFER HAS BEEN ASSIGNED AND THERE IS ONE AVAILABLE, THE STATION IS ASSIGNED THE OPEN BUFFER BY MEANS OF A WORD POINTER. IF ALL AVAILABLE BUFFERS ARE BEING USED, THE OPERATOR OF THE STATION IS APPROPRIATELY NOTIFIED.

THE NORMAL QUANTUM OF DATA HANDLED IN THE INTERRUPT SERVICE BUFFERS IS A LINE TERMINATED BY A CARRIAGE RETURN (OR UP TO 75 CHARACTERS). THEREFORE, CHARACTERS ARE RECEIVED OR TRANSMITTED UNTIL THE CONSOLE BUFFER IS EMPTY, FULL OR TERMINATED BY AN APPROPRIATE CHARACTER. WHEN A BUFFER BECOMES EMPTY IN OUTPUT MODE, THE I/O SERVICE SEES IF THE USER PROGRAM HAS MORE DATA IT WISHES TO TRANSMIT. THIS IS DETERMINED BY THE USER PROGRAM SPECIFYING EITHER CONTINUOUS OPERATION, IN WHICH ALL FULL BUFFERS ARE SERVICED, OR NON-CONTINUOUS, IN WHICH ONLY ONE BUFFER IS SERVICED.

WHEN A BUFFER BECOMES FULL OR TERMINATED IN INPUT MODE, THE I/O SERVICE TRANSFERS THE BUFFER INTO THE BUFFER AREA OF THE USER PROGRAM THAT REQUESTED THE DATA. IF THE DATA HAS NOT BEEN REQUESTED BY ANY USER PROGRAM, IT IS TRANSFERRED TO THE COMMAND DECODER IN THE MONITOR FOR COMMAND ANALYSIS. THUS, INPUT IS EITHER TO A USER'S OWN PROGRAM, OR A COMMAND TO THE SYSTEM.

EXECUTIVE OR MONITOR STORAGE (EXECUTIVE MODE)

THE TIME SHARING MONITOR IS A COLLECTION OF PROGRAMS THAT ARE PERMANENTLY IN CORE MEMORY DURING NORMAL DATA PROCESSING AND DEBUGGING OPERATIONS AND SUPERVISE THE OPERATION OF THE USER'S PROGRAM. THE TIME SHARING MONITOR PERFORMS SEVERAL FUNCTIONS.

FIRST, IT PERMITS SEVERAL USER'S PROGRAMS TO BE LOADED INTO CORE MEMORY SIMULTANEOUSLY. THE MONITOR MAKES USE OF THE PDP-6 TIME SHARING HARDWARE TO PREVENT EACH USER'S PROGRAM FROM INTERFERING WITH OTHER USER'S PROGRAMS. EACH PROGRAM IS RUN FOR A CERTAIN LENGTH OF TIMES THEN THE MONITOR SWITCHES CONTROL TO ANOTHER PROGRAM IN A ROUND-ROBIN FASHION. SWITCHING IS FREQUENT ENOUGH SO THAT ALL PROGRAMS APPEAR TO RUN SIMULTANEOUSLY.

THE SECOND FUNCTION OF THE TIME SHARING MONITOR IS TO PROCESS INPUT/OUTPUT COMMANDS. ONLY ONE USER AT A TIME IS PERMITTED TO OPERATE EACH PARTICULAR DEVICE. THE INPUT/OUTPUT SERVICE ROUTINES PREPROCESS INPUT/OUTPUT DATA SO THAT ALL DEVICES APPEAR IDENTICAL TO THE USER'S PROGRAM THUS SIMPLIFYING INPUT/OUTPUT CODING. THE MONITOR MAKES USE OF THE PDP-6 PROGRAM INTERRUPT SYSTEM TO OVERLAP INPUT/OUTPUT OPERATIONS WITH COMPUTATION. IF A USER'S PROGRAM MUST WAIT FOR THE COMPLETION OF AN INPUT OR OUTPUT OPERATION, THE MONITOR AUTOMATICALLY SWITCHES TO ANOTHER USER'S PROGRAM.

THE RESULT OF THESE FUNCTIONS IS THAT THE PDP-6 COMPUTER ALWAYS IS PROCESSING DATA, SINCE AS SOON AS ONE PROGRAM HAS NOTHING TO DO, TIME SHARING MONITOR SWITCHES CONTROL TO ANOTHER PROGRAM. TIME SHARING PERMITS THE ECONOMICAL USE OF THE PDP-6 TO PERFORM ON-LINE DEBUGGING OR PROGRAM EDITING.

MULTI-PROGRAMMING EXECUTIVE

THE EXECUTIVE PROVIDES OVERALL CO-ORDINATION AND CONTROL OF THE TOTAL OPERATING SYSTEM. THE MULTI-PROGRAMMING SYSTEM STORAGE DIAGRAM DEPICTS THESE COMPONENTS. IT MUST:

1. ALLOCATE FACILITIES

2. HANDLE ALL COMMANDS ADDRESSED TO THE SYSTEM FROM THE USER CONSOLE

3. CO-ORDINATE I/O BETWEEN THE USER AND THE I/O SERVICE.

4. PROVIDE COMMUNICATION TO A USER'S CONSOLE.

IT SCHEDULES JOBS. ONCE A USER PROGRAM HAS BEEN GIVEN CONTROL, IT CAN BE REMOVED FROM RUN STATUS BY ANY OF THE FOLLOWING MEANS:

1. THE PROGRAM IS COMPLETED AND DOES NOT REQUIRE FURTHER PROCESSING.

2. THE PROGRAM HAS RUN FOR ITS ALLOTTED AMOUNT OF TIME AND MUST BE RE-SCHEDULED.

3. THE PROGRAM HAS ISSUED AN I/O COMMAND AND ISSUES A "WAIT" REQUEST.

4. A PROGRAM OF HIGHER PRIORITY MUST BE RUN.

AS A PROGRAM IS REMOVED FROM RUN STATUS, THE EXECUTIVE MUST TRANSFER CONTROL TO ANOTHER PROGRAM AS QUICKLY AS POSSIBLE.

A DESCRIPTION OF THE EXECUTIVE OR MONITOR COMPONENTS IS GIVEN IN AN APPENDIX.

INPUT-OUTPUT INTERRUPT SERVICE MODULES

EACH ACTIVE IO DEVICE IN THE SYSTEM HAS AN INPUT/OUTPUT SERVICE ROUTINE MODULE.
THE EFFECTIVENESS OF THE TIME SHARING SYSTEM DEPENDS LARGELY ON THE EFFICIENCY OF THE I/O INTERRUPT SERVICE. TO IMPLEMENT IT PROPERLY, TWO HARDWARE FEATURES WERE INCORPORATED INTO THE PDP-6.

1. A MULTIPLE LEVEL PRIORITY INTERRUPT SYSTEM THAT CAN BE PROGRAM ASSIGNED.

2. A FULLY BUFFERED I/O SYSTEM TO ALLOW THE CENTRAL

PROCESSOR TO SERVICE I/O AT A RATE THAT APPROACHES MEMORY SPEEDS.

THE INTERRUPT SERVICE IS COMPLETELY MODULAR TO ALLOW A SMALL SYSTEM TO CARRY ONLY THOSE ROUTINES DIRECTLY RELATED TO ITS PERIPHERAL EQUIPMENT. THUS THE MEMORY CAPACITY OF A SMALL SYSTEM IS NOT OVER-TAXED, YET EXPANSION OF THE SYSTEM IS NOT RESTRICTED. AS MORE I/O IS ADDED, THE PROPER SERVICE ROUTINES ARE ADDED INTO THE SYSTEM AND BUFFER AREAS ARE EXTENDED.

THUS SERVICE IS IMPLEMENTED IN SUCH A WAY THAT THE PROGRAMS ARE ABLE TO RECEIVE AND TRANSMIT A SUFFICIENTLY LARGE QUANTUM OF DATA IN THE ASCII FORMAT, COMMON TO ALL I/O EQUIPMENT. THEREFORE, I/O SERVICE RELIEVES THE PROGRAMMER OF THE BURDEN OF FORMAT TRANSLATIONS AND ALSO INSURES THAT WHEN A USER RECEIVES OR TRANSMITS A QUANTUM OF DATA, IT WILL BE LARGE ENOUGH TO BE USEFUL.

JOB STORAGE (USER MODE)

A SIMPLIFIED DIAGRAM OF JOB AREA STORAGE DEPICTS A JOB OR USER'S AREA OF MEMORY. THE USER AREA COMPOSES THE LARGEST BULK OF MEMORY AND IS UNDER COMPLETE CONTROL OF THE MONITOR. SINCE MANY USER PROGRAMS WILL NOT HAVE BEEN DEBUGGED, SEVERAL SAFEGUARDS ARE USED TO INSURE THERE WILL BE NO UNWANTED USER-USER OR USER-MONITOR INTERACTION, INCLUDING

1. TRYING TO HALT THE PROCESSOR

2. MAKING MEMORY REFERENCES OUTSIDE A MONITOR DEFINED AREA IN MEMORY

3. TRYING TO CONTROL OR INFLUENCE THE I/O OR PRIORITY INTERRUPT SYSTEM OUTSIDE MONITOR CONTROL.

FOR CONVENIENCE AND FLEXIBILITY, HOWEVER, THE USER MUST BE ABLE TO INPUT, OUTPUT, CONTROL, AND SELECT PERIPHERAL EQUIPMENT THROUGH THE MONITOR. THE USER MUST BE ABLE TO LOAD SEVERAL OF HIS PROGRAMS TOGETHER WITH SYMBOLS, HAVING THEM LINKED AND RELOCATED TO FIT PROPERLY INTO HIS ASSIGNED AREA.
CHAPTER III DESCRIBES THE JOB OR USER'S AREA IN DETAIL, BUT THIS AREA IS WHERE A COMMON USER SERVICE PROGRAM (CUSP) OR A USER PROGRAM (UP) IS LOADED AND RUN.
AREA IS WHERE A CUSP OR UP IS LOADED AND RUN.

THE MAJOR AREAS OF A JOB OR USER'S AREA ARE:

1. SYSTEM USEAGE - TEMPORARY STORAGE IS REQUIRED WHERE THE SYSTEM IS DOING WORK ON THE BEHALF OF A USER. ALSO MOST SYSTEM DATA REGARDING A JOB IS KEPT WITH THE JOB. THIS TEMPORARY WHICH THE SYSTEM REQUIRES IS LOADED AND/OR UNLOADED AS THE JOB IS STARTED OR STOPPED.

2. PROGRAM - THE PROGRAM ITSELF REMAINS IN THE JOB AREA, INCLUDING ITS FORTRAN COMMON, SYMBOL TABLE, COPY OF DDT, LIBRARY FILES, FUNCTION, ETC.

3. I/O BUFFERS - AS A JOB DOES INPUT/OUTPUT, BUFFERS ARE DYNAMICALLY ASSIGNED IN A JOB'S OWN AREA, AND DATA ARE TRANSMITTED DIRECTLY TO THESE BUFFERS FROM THE IO DEVICE SERVICE PROGRAMS.

INPUT/OUTPUT DATA FILES ORGANIZATION

THIS PORTION OF THE SYSTEM CONTAINS THE FILES OR DATA STORAGE FOR ALL JOBS AND USERS.

A FILE IS A LINEAR STRING OF DATA WORDS OR DATA CHARACTERS. FILES HAVE FILE NAMES WHICH APPEAR IN ASSOCIATED FILE DIRECTORIES, OR FILES NAMES ARE NAMED IMPLICITLY BY THE DEVICE ON WHICH THEY APPEAR.

EACH FILE HAS A DATA MODE WHICH DENOTES THE TYPE OF DATA WITHIN THE FILE.

A FILE NAME EXTENSION IS OFTEN INCLUDED TO FURTHER DISTINGUISH TWO FILES.

FILE DIRECTORY OR LOCATION DESIGNATION

1-3 OR 1-6	CHARACTER DEVICE NAME
1-6	CHARACTER FILE NAME
1-3	CHARACTER FILE NAME EXTENSION
1 CHARACTER	(4 BITS) FILE MODE
FILE ACTIVITY OR STATUS INFORMATION	
POINTER TO DATA	

DATA

DATA STRING (CHARACTERS, WORDS, ETC.)

THE ALLOWABLE FILE NAME FORMATS FROM A CONSOLE ARE DESCRIBED IN CHAPTER II, FILE COMMAND FORMATS.

THE FILE DATA MODE SPECIFICATIONS ARE GIVEN IN CHAPTER IV.

BRIEFLY, THE FILE DATA STRUCTURE IS:

1. DEVICES ARE DENOTED PTR (PAPER TAPE READER), PTP, CDR, DTAØ,...7, ETC.

2. FILE NAMES (ARBITRARY)

3. FILE NAME EXTENSIONS TXT (TEXT), MAC, FOR, ETC.

4. FILE DATA MODES

   A ASCII MODE DATA PACKED 5-7 BIT CHARACTERS, EITHER PACKED IN SMALL PHYSICAL RECORDS (IF TAPE) OR ON A SINGLE PAPER TAPE OR CARD DECK.

   B BINARY MODE DATA 36 BIT WORDS. USUALLY PACKED IN SMALL PHYSICAL RECORDS.

   D DUMP MODE, CONTIGUOUS BLOCK OF BINARY WORDS, GENERALLY USED TO STORE A CORE IMAGE, OR PROGRAM, IN A SINGLE FILE.

   I IMAGE MODE, RECEIVED DIRECTLY FROM DEVICE

SYSTEM LIBRARY FILE

THIS LIBRARY CONTAINS ALL THE ACTIVE PROGRAMS WHICH ARE USED WITHIN THE SYSTEM. A TABLE OF THE LIBRARY, ITS COMPONENT

MNEMONICS, ETC. IS GIVEN IN THE APPENDIX.

C H A P T E R I I

C O N S O L E U S E O F T H E S Y S T E M

INTRODUCTION

THE CONSOLE COMMANDS GIVE THE SYSTEM USER COMPLETE ACCESS TO ALL SYSTEM FUNCTIONS AND PROGRAMS, AS WELL AS THE ABILITY TO CREATE, EDIT, LOAD, RUN, AND DEBUG HIS OWN PROGRAMS. THE MONITOR MAKES NO DISTINCTION BETWEEN COMMON USER SERVICE PROGRAMS AND USER PROGRAMS. IT HANDLES BOTH KINDS OF PROGRAMS IN THE SAME WAY.

PHYSICAL DECTAPE 1 (DTA1) CONTAINS THE COMPLETE SYSTEM LIBRARY. THE SYSTEM IS OPEN ENDED SINCE A USEFUL USER PROGRAM MAY BECOME A CUSP SIMPLY BY ADDING IT TO THE SYSTEM LIBRARY TAPE. HENCEFORTH, THE TERM "PROGRAM" SHALL MEAN BOTH COMMON USER SERVICE PROGRAMS AND USER PROGRAMS.

THE PHILOSOPHY OF THE MULTIPROGRAMMING TIME SHARING SYSTEM IS THAT USERS MUST COOPERATE WITH EACH OTHER TO A CERTAIN EXTENT. ENOUGH SAFEGUARDS HAVE BEEN IMPLEMENTED TO PREVENT MISTAKES BY ONE USER FROM HARMING A FELLOW USER. THE SYSTEM DOES NOT PREVENT A USER FROM DISTURBING OTHER USERS. FOR EXAMPLE, A USER MAY RESERVE THE LINE PRINTER FOR AN INDEFINITE PERIOD WITHOUT USING IT, OR AN EARLY USER MAY BE GRANTED AN EXCESSIVE AMOUNT OF CORE PREVENTING OTHER USERS FROM USING THE SYSTEM.

CONSOLE STATES

THE CONSOLE STATE DIAGRAM SHOWS THE "STATES" OR "MODES" OF A CONSOLE. EACH LINE INDICATES A COMMAND OR ACTION WHICH WILL CHANGE THE STATE OF THE USER'S PROGRAM, OR HIS JOB. SOME STATE TO STATE TRANSITIONS ARE AUTOMATIC, WHILE OTHERS ARE CONSOLE INITIATED. A CONSOLE IS THUS EITHER CONNECTED TO THE MONITOR, OR TO A PROGRAM RUNNING IN A JOB AREA. A JOB MAY BE EITHER A CUSP (LOADER, EDITOR, TRANSLATOR, ETC) OR A USER PROGRAM (A RUNNING PROGRAM, OR THE DDT ASSOCIATED WITH A USER'S PROGRAM.)

THE MONITOR DOES NOT DIFFERENTIATE BETWEEN A CUSP AND A UP.

MONITOR LEVEL CONTROL

THERE ARE TWO MONITOR CONSOLE STATES, DORMANT, OR MONITOR COMMAND. THE DORMANT STATE EXISTS WHERE NO JOB IS PRESENT, AND THE MONITOR COMMAND STATE INDICATES A JOB IS PRESENT. FROM THIS STATE, A JOB MAY BE GOTTEN FROM A LIBRARY FILE (USING "GET") AND PROGRAMS CAN BE LOADED.

COMMON USER SERVICE PROGRAM LEVEL CONTROL

THE MAJOR STATE ASSOCIATED WITH A CUSP, IS WHEN THE CUSP IS GOTTEN FROM THE SYSTEM LIBRARY USING "GETS". THE CONSOLE THEN COMMUNICATES WITH CUSP. A USER'S PROGRAM IS LOADED USING THE LOADER.

USER PROGRAM LEVER CONTROL

TWO MAJOR STATES EXIST WITH A USER PROGRAM WHICH HAS BEEN LOADED WITH A USER DDT RELOCATABLE BINARY FILE. ONE STATE IS WHEN THE USER PROGRAM IS BEING RUN AT THE USER'S SPECIFIED STARTING ADDRESS, AND THE CONSOLE MAY BE USED BY THE JOB OR IS DISCONNECTED.

THE SECOND STATE IS ENTERED WHEN DDT IS REQUESTED. DDT STATE IS ENTERED BY FIRST ENTERING THE MONITOR COMMAND LEVEL FOLLOWED BY A REQUEST FOR DDT.

MONITOR COMMANDS

A USER MAY RUN MORE THAN ONE PROGRAM OR JOB AT A TIME. A JOB IS IDENTIFIED BY A JOB NUMBER STARTING WITH 1. ALTHOUGH EACH JOB MAY HAVE ONLY ONE PROGRAM RUNNING UNDER IT, THE USER MAY INITIATE MORE THAN ONE JOB. IN ORDER TO COMMUNICATE WITH A PROGRAM, THE CONSOLE MUST BE ATTACHED TO THE APPROPRIATE JOB. INITIALLY A CONSOLE IS IN COMMAND MODE AND IS NOT ATTACHED TO ANY JOB. WHEN THE FIRST COMMAND IS TYPED, THE CONSOLE IS AUTOMATICALLY ATTACHED TO AN UNUSED JOB AND THE JOB IS INITIALIZED (SEE IJOB COMMAND TO SEE WHAT INITIALIZED MEANS). IF THERE ARE NO UNUSED JOBS AVAILABLE THE MONITOR WILL IGNORE THE COMMAND AND TYPE OUT:

N JOB CAPACITY EXCEEDED

WHERE N IS THE JOB CAPACITY WRITING THE NULL JOB.
THIS WILL HAPPEN EACH TIME A COMMAND IS TYPED UNTIL A JOB BECOMES FREE

INITIALLY THE CONSOLE IS IN COMMAND MODE AND EACH INPUT LINE IS INTERPRETED AS A COMMAND TO THE MONITOR. SOME OF THESE COMMANDS INSTRUCT THE MONITOR TO START OR CONTINUE EXECUTION OF A PROGRAM WHICH HAS PREVIOUSLY BEEN LOADED. ALL SUCCEEEDING CONSOLE INPUT IS THEN CONSIDERED AS INPUT TO THE PROGRAM UNTIL THE PROGRAM RETURNS CONTROL TO THE MONITOR BY CALLING EXIT, (CALL SIXBIT /EXIT/)
THE MONITOR RESPONDS BY TYPING:

EXIT

AND THE CONSOLE IS ONCE AGAIN IN COMMAND MODE.
ALL TYPEOUT WILL BE SUSPENDED IF THE USER TYPES IN A CHARACTER WHILE HIS PROGRAM IS TYPING OUT. THE NEXT CHARACTER TYPED IN IS INTERPRETED IN A SPECIAL WAY. IF IT IS <CONTROL> C, THE MONITOR WILL STOP THE JOB, TYPE OUT ↑C, AND RETURN THE CONSOLE TO COMMAND MODE. IF IT IS <CONTROL>O, THE MONITOR WILL CONTINUE RUNNING THE PROGRAM, TYPE OUT ↑O AND SUPPRESS THE PRINTING OF THE CURRENT CONTENTS OF THE CONSOLE OUTPUT BUFFER. ANY OTHER CHARACTER WILL CAUSE BOTH THE OUTPUT AND THE EXECUTION TO CONTINUE.

COMMANDS ARE COMPOSED OF ONE OR MORE FIELDS OF ALPHANUMERIC CHARACTERS SEPARATED BY A NON-ALPHANUMERIC CHARACTER AND TERMINATED BY A CARRIAGE RETURN (OR A LINE-FEED, A FORM FEED, OR A VERTICAL TAB).

LEADING SPACES AND TABULATES ARE IGNORED IN ANY FIELD. ONLY THE FIRST 6 CHARACTERS OF THE FIRST FIELD ARE USED FOR THE COMMAND NAME AND THE FIRST 6 CHARACTERS OF EVERY FIELD THEREAFTER AS ARGUMENTS. A FIELD SEPARATION CHARACTER MAY BE SELECTED FOR MNEUMONIC VALUE. BLANK LINES ARE IGNORED.

THE SYSTEM WILL RESPOND IMMEDIATELY WITH A LINEFEED AND WILL TYPE A SECOND CARRIAGE RETURN-LINE FEED, WHEN THE COMMAND IS FINISHED.

NO OTHER ACKNOWLEDGEMENT WILL BE PRINTED UNLESS THERE IS AN ERROR. IF AN UNKNOWN COMMAND IS TYPED, THE MONITOR WILL TYPE THE COMMAND NAME BACK WITH A QUESTION MARK APPENDED. IF A FIELD IS SUPPOSED TO BE ENTIRELY NUMERIC OF A PRESCRIBED RADIX AND AN ILLEGAL ALPHANUMERIC CHARACTER OCCURS, THE MONITOR WILL TYPE OUT THE ENTIRE LINE UP TO AND INCLUDING THE ERRONEOUS CHARACTER WITH A QUESTION MARK APPENDED.

AS A GENERAL RULE ALL NUMERIC ARGUMENTS ARE DECIMAL EXCEPT WHERE OCTAL IS MORE CONVENIENT TO THE USER. EXTRA ARGUMENTS TO COMMANDS WILL BE IGNORED AND NO MESSAGE PRINTED. IF A COMMAND IS TYPED WITH AN INSUFFICIENT NUMBER OF ARGUMENTS IT WILL CAUSE THE MONITOR TO RESPOND WITH:

NOT ENOUGH ARGUMENTS

AND TO IGNORE THE COMMAND.

THE FOLLOWING USER COMMANDS ARE INDICATED BY QUOTES, ". IF ERRORS MAY OCCUR WHEN THE COMMANDS ARE GIVEN, THEY ARE INDICATED. THE COMMANDS ARE:

"IJOB" INITIALIZE JOB COMMAND
INITIALIZES THE JOB TO WHICH THE CONSOLE IS ATTACHED. THIS MEANS THE JOB HAS NO CORE ASSIGNED TO IT (SEE "CORE" COMMAND), NO DEVICES ASSIGNED TO IT EXCEPT THE TELETYPE, TTY IS SET AS THE ONLY LOGICAL/PHYSICAL DEVICE EQUIVALENCE (SEE "ASSIGN" COMMAND).

"PJOB" PRINT JOB NUMBER COMMAND
PRINTS THE JOB NUMBER TO WHICH THE CONSOLE IS CURRENTLY ATTACHED.

"KJOB" KILL THE JOB COMMAND
KILLS THE JOB AND RETURNS TO THE SYSTEM ALL RESOURCES ASSIGNED TO IT. THE CONSOLE IS ALSO DETACHED (SEE "DETACH" COMMAND) AND THE JOB IS THEN AVAILABLE FOR SOME OTHER USER. THIS COMMAND SHOULD BE TYPED WHENEVER A USER IS THROUGH WITH A JOB, IN ORDER TO ALLOW SOMEONE ELSE TO USE THE JOB NUMBER.

"CORE N" GET CORE FOR JOB COMMAND
SETS THE TOTAL NUMBER OF 1024 WORD (DECIMAL) BLOCKS OF CORE TO N (DECIMAL) FOR THE JOB TO WHICH THE CONSOLE IS CURRENTLY ATTACHED. IF THE JOB ALREADY HAS CORE ASSIGNED, IT IS EXTENDED (IF POSSIBLE) OR CONTRACTED TO SATISFY THE REQUEST. IF NO CORE IS ASSIGNED, THE MONITOR SEARCHES THE UNUSED BLOCKS OF CORE AND ASSIGNS N (IF POSSIBLE). IN EITHER CASE, IF THERE IS NOT ENOUGH FREE CORE

TO SATISFY THE REQUEST, THE MONITOR WILL ASSIGN AS MUCH AS POSSIBLE AND TYPE :

K BLOCKS ASSIGNED

JOBS WILL NOT BE SHUFFLED IN MEMORY TO TRY TO SATISFY A PARTICULAR REQUEST. ALSO ANY NEW CORE ASSIGNED WILL NOT BE CLEARED. THE COMMAND "CORE Ø" WILL RETURN ALL OF A JOB'S CORE TO THE SYSTEM. IF A USER ATTEMPTS TO EXTEND HIS CORE ASSIGNMENT TO K BLOCKS AND FINDS THAT THERE IS NOT ENOUGH ROOM, HE MAY SAVE HIS CORE IMAGE (SEE "SAVE" COMMAND), RETURN ALL OF HIS CORE TO THE SYSTEM ("CORE Ø"), REQUEST K BLOCKS ("CORE K"), AND RESTORE HIS CORE IMAGE (SEE "GET" COMMAND). IT IS UP TO EACH COMPUTER INSTALLATION TO DEVELOP GROUND RULES FOR USE OF "CORE".

"GET DEVICE:FILE" GET A FILE COMMAND
GETS A PREVIOUSLY SAVED PROGRAM FROM DEVICE "DEVICE" AND PLACES IT IN CORE, WHERE THE FILE NAME OF THE PROGRAM IS FILE DMP. THIS IS THE MECHANISM FOR LOADING SYSTEM PROGRAMS SUCH AS THE ASSEMBLER. HOWEVER, IT IS ALSO USEFUL FOR LOADING USER PROGRAMS WHICH HAVE BEEN PREVIOUSLY SAVED BY THE "SAVE" COMMAND. THE PROGRAM IS NOT STARTED.

ERROR MESSAGES:

NO CORE ASSIGNED
DEVICE NOT AVAILABLE
FILE NOT FOUND
NOT ENOUGH CORE ASSIGNED
DEVICE ERROR
DATA ERROR

"SAVE DEVICE:FILE" SAVE THE JOB COMMAND
SAVES THE CORE IMAGE ON DEVICE "DEVICE" AND GIVES IT A FILE NAME OF FILE.DMP. ALL DEVICES ARE RELEASED AND THE PROGRAM COUNTER IS SET TO THE PROGRAM'S STARTING ADDRESS BEFORE THE FILE IS WRITTEN. ERROR MESSAGES:

NO CORE ASSIGNED
DEVICE NOT AVAILABLE
DIRECTORY FULL
DEVICE ERROR
DATA ERROR

"START LOC" START AT LOC COMMANDS
STARTS EXECUTION OF THE PROGRAM AT RELATIVE OCTAL LOCATION "LOC" IN THE JOB AREA OR AT THE PROGRAM STARTING ADDRESS IF "LOC" IS NOT SPECIFIED. NO CHECK IS MADE TO SEE IF A PROGRAM HAS BEEN LOADED. ALL SUCCEEDING INPUT WILL BE DIRECTED TO THE USER'S PROGRAM. THIS COMMAND WILL BE OPERATIVE EVEN AFTER A SYSTEM ERROR MESSAGE HAS BEEN PRINTED OR "EXIT" CALLED WHILE EXECUTING

THE PROGRAM.
ERROR MESSAGES:

NO CORE ASSIGNED

"CONT" CONTINUE JOB COMMAND
CONTINUOUS EXECUTION FROM WHEREVER THE PROGRAM WAS STOPPED BY <CONTROL>C. ALL SUCCEEDING INPUT WILL BE DIRECTED TO THE USER'S PROGRAM. THIS COMMAND WILL NOT BE OPERATIVE AFTER A MONITOR DETECTED ERROR MESSAGE HAS BEEN PRINTED OR "EXIT" CALLED WHILE EXECUTING THE PROGRAM.
ERROR MESSAGES:

NO CORE ASSIGNED

"DDT" START USER DDT COMMAND
STARTS EXECUTION OF USER DDT IN THE JOB AREA. ALL SUCCEEDING INPUT WILL BE DIRECTED TO "DDT" AND THE USER'S PROGRAM. THIS COMMAND WILL BE OPERATIVE EVEN AFTER A SYSTEM ERROR MESSAGE HAS BEEN PRINTED OR "EXIT" CALLED.
ERROR MESSAGES:

NO CORE ASSIGNED
NO DDT LOADED

"STARTM LOC" START JOB, MONITOR, COMMAND
WORKS EXACTLY LIKE "STARTM LOC" EXCEPT THE CONSOLE REMAINS IN MONITOR MODE. THIS COMMAND IN CONJUNCTION WITH THE "DETACH" AND "ATTACH" COMMANDS ALLOW THE USER TO EXECUTE MORE THAN ONE PROGRAM AT THE SAME TIME.
ERROR MESSAGES:

NO CORE ASSIGNED

"CONTM" CONTINUE JOB, MONITOR, COMMAND
WORKS LIKE "CONT" EXCEPT THE CONSOLE REMAINS IN MONITOR MODE.
ERROR MESSAGES:

NO CORE ASSIGNED
CAN'T CONTINUE

"DETACH" CONSOLE FROM JOB COMMAND
DETACHES THE CONSOLE FROM THE JOB TO WHICH IT IS ATTACHED. IF THE JOB IS STILL RUNNING IT WILL CONTINUE TO RUN. THIS LEAVES THE CONSOLE IN THE STATE IT WAS IN BEFORE THE FIRST COMMAND WAS TYPED.

THE NEXT COMMAND TYPED-IN WILL AUTOMATICALLY ATTACH THE CONSOLE TO AN UNUSED JOB.

"ATTACH N" CONSOLE TO JOB COMMAND
DETACHES THE CONSOLE FROM THE JOB TO WHICH IT IS CURRENTLY ATTACHED AND ATTACHES IT TO JOB N. THE LOGICAL-PHYSICAL DEVICE EQUIVALENCE IS SETUP SO THAT A PROGRAM MAY COMMUNICATE WITH THE CONSOLE BY REFERRING TO IT SIMPLY AS TTY. (SEE "ASSIGN" COMMAND). SINCE THE CONSOLE IS IMPLICITLY ATTACHED TO A JOB WHENEVER THE FIRST COMMAND IS TYPED (INCLUDING "ATTACH"), "ATTACH" SHOULD NOT BE THE FIRST COMMAND TYPED. IN FACT, THIS COMMAND SHOULD BE USED ONLY TO RECONNECT THE CONSOLE TO A JOB WHICH HAS PREVIOUSLY BEEN "DETACH"ED.
ERROR MESSAGE:

ANOTHER CONSOLE ALREADY ATTACHED

IN THIS CASE THE CONSOLE REMAINS ATTACHED TO THE ORIGINAL JOB.

"ASSIGN DEV:NAME" TO JOB COMMAND
ASSIGNS PHYSICAL DEVICE DEV TO THE JOB. NO OTHER JOB MAY USE A DEVICE ONCE IT HAS BEEN ASSIGNED.

IF DEV IS THE FIRST 3 CHARACTERS OF ONE OF THE MULTIPLE DEVICES (TTY, DTA, MTA) THE MONITOR WILL SEARCH FOR A FREE DEVICE. IN ANY CASE, THE MONITOR WILL INDICATE THE DEVICE ASSIGN BY TYPING:

DEVICE DEVN ASSIGNED

ANY NUMBER OF DEVICES MAY BE ASSIGNED BY REPEATED USES OF ASSIGN. THIS COMMAND ALLOWS A USER TO PROTECT HIS PRIVATE DECTAPES FROM BEING REFERENCED BY ANOTHER USER. IT ALSO ALLOWS A USER AT A REMOTE CONSOLE TO TEST THE AVAILABILITY OF A PARTICULAR DEVICE. THE ASSIGN COMMAND DOES NOT HAVE TO BE TYPED BEFORE A PROGRAM CAN USE THE DEVICE AND IT MAY BE TYPED AFTER THE PROGRAM HAS BEGUN TO USE THE DEVICE.
ERROR MESSAGE:

DEVICE NOT AVAILABLE

IF THE SECOND ARGUMENT IS PRESENT, THE MONITOR WILL NAME "NAME" EQUIVALENT TO DEVICE "DEV" WHERE "NAME" MAY BE ANY ALPHA-NUMERIC STRING (FIRST SIX CHARACTERS USED). PHYSICAL DEVICES NAMES ARE ALL THREE LETTERS LONG SUFFIXED BY A NUMBER IF THE SYSTEM HAS MORE THAN ONE OF THE DEVICES. THE POSSIBLE PHYSICAL DEVICES ARE

WHEN A JOB IS INITIALIZED AN EQUIVALENCE IS SET UP BETWEEN TTY AND TTYN WHEN TTYN IS THE PHYSICAL DEVICE NAME FOR THE CONSOLE ATTACHED TO THE JOB. THUS PROGRAMS MAY REFERENCE THE ATTACHED CONSOLE BY REFERRING TO THE LOGICAL DEVICE NAME TTY.

A TYPICAL USE FOP THIS COMMANU WOULD BE WHERE A PROGRAM IS KNOWN TO REFERENCE A SPECIFIC DEVICE WHICH A PREVIOUS "ASSIGN" COMMAND HAS REVEALED IS NOT AVAILABLE.

THIS COMMAND ALLOWS USER PROGRAMS TO BE WRITTEN WITHOUT KNOWING IN ADVANCE WHICH PHYSICAL TAPE DRIVE A TAPE IS TO BE MOUNTED. IT IS RECOMMENDED NEVER TO REFER DIRECTLY TO A MULTIPLE DEVICE BY ITS PHYSICAL NAME. INSTEAD THE FIRST 3 LETTERS OF THE DEVICE BY A LETTER (STARTING WITH "A" E.G). THUS IF A USER PROGRAM REQUIRES TWO DECTAPES THEY WOULD BE DTAA AND DTAB. USING THE APPROPRIATE "ASSIGN" COMMAND, A PROGRAM MAY ALWAYS BE RUN, AND NEED NOT WAIT UNTIL PHYSICAL DEVICES ARE FREE, OR FOR CERTAIN ASSIGNMENT CONVENTIONS FOR PHYSICAL DEVICES.

IN A SYSTEM WHERE REMO"E CONSOLES ARE USED, THE SEARCH FEATURE OF THE "ASSIGN" COMMAND ALLOWS THE USER TO ASSIGN A FREE DECTAPE OR MAGTAPE BEFORE REQUESTING THE OPERATOR TO MOUNT A PARTICULAR TAPE ON THE ASSIGNED PHYSICAL DEVICE.

THIS COMMAND IS NOT REQUIRED FOR ANY OF THE SYSTEM PROGRAMS SINCE THE USER TYPES IN THE DEVICES TO BE USED DIRECTLY TO THE PROGRAM. USER PROGRAMS WHICH ARE CANDIDATES TO BECOME SYSTEM PROGRAMS SHOULD ALSO BE WRITTEN TO ACCEPT THE DEVICE NAME FROM THE CONSOLE.

"DEASSIGN DEV" FROM JOB COMMAND
DEASSIGNS DEVICE "DEV". THIS MAY BE DONE EVEN THOUGH THE PROGRAM IS CONTINUING TO USE THE DEVICE. "DEVICE" MAY BE EITHER A PHYSICAL OR LOGICAL DEVICE. A SEARCH IS FIRST MADE FOR A LOGICAL DEVICE - FOLLOWED BY A SEARCH FOR A PHYSICAL DEVICE.

THE CONSOLE WILL ALSO BE RETURNED TO COMMAND MODE WHENEVER THE MONITOR DETECTS AN ERROR. TO HELP DISTINGUISH BETWEEN MONITOR DETECTED ERRORS AND PROGRAM DETECTED MESSAGES, THE MONITOR WILL TYPE:

MONITOR DETECTED ERROR
ERROR IN JOB N

BEFORE TYPING THE ERROR MESSAGE.

THE MONITOR ERROR MESSAGES:

MONITOR ERROR AT N; MONITOR CALLED FROM USER/MONITOR LOC M

WHERE N IS THE LOCATION OF THE ERROR WITHIN THE MONITOR AND M IS THE LOCATION IN THE USER PROGRAM FROM WHICH THE MONITOR WAS CALLED.

OUTPUT DEVICE, DEV, CANNOT DO INPUT AT USER LOC

WHERE DEV IS THE DEVICE REQUESTED AND M IS THE LOCATION IN THE JOB AREA OF THE ILLEGAL INPUT PROGRAMMED OPERATOR.

INPUT DEVICE, DEV, CANNOT DO OUTPUT; AT USER LOC M

WHERE DEV IS THE DEVICE REQUESTED AND M IS THE LOCATION IN THE JOB AREA OF THE ILLEGAL OUTPUT PROGRAMMED OPERATOR.

ILL INST AT USER LOC M

WHERE M IS THE LOCATION IN THE JOB AREA OF THE ILLEGAL INSTRUCTION. A ZERO OPCODE IS CONSIDERED TO BE AN ILLEGAL INSTRUCTION.

ILL DEVICE DATA MODE AT USER LOC M

WHERE M IS THE LOCATION IN THE JOB AREA OF THE INIT PROGRAMMED OPERATOR SPECIFYING AN ILLEGAL DEVICE DATA MODE.

IO TO UNASSIGNED CHANNEL AT USER LOC M

WHERE M IS THE LOCATION IN THE JOB DATA AREA OF A PROGRAMMED OPERATOR REFERRING TO A USER CHANNEL WHICH HAS NOT BEEN ASSIGNED BY A PREVIOUS INIT PROGRAMMED OPERATOR.

ILLEGAL PRO OPE USED AT USER/MONITOR LOC M

WHERE M IS THE LOCATION IN THE JOB AREA OF THE ILLEGAL PROGRAMMED OPERATOR. THE SYSTEM USES THE PROGRAMMED OPERATORS FROM 40 TO 77 LEAVING 1-37 FOR THE USER'S OWN USE. IF A PROGRAM EXECUTES A PROGRAMMED OPERATOR BETWEEN 1 AND 37 AND LOCATION 41 IS NOT SET UP WITH A JSR INSTRUCTION (SEE PDP-6 INSTRUCTION MANUAL F-65), THIS ERROR MESSAGE WILL OCCUR. OTHERWISE THE MONITOR WILL SIMULATE THE EXECUTION OF THE PROGRAMMED OPERATOR INCLUDING THE JSR INSTRUCTION IN LOCATION 41 IN THE JOB AREA.

ADDRESS CHECK FOR DEVICE DEVICE; MONITOR CALLED FROM USER LOC M

WHERE DEVICE IS THE DEVICE FOR WHICH THE MONITOR ATTEMPTED TO MAKE A REFERENCE OUTSIDE THE JOB AREA AND M IS WHERE THE MONITOR WAS CALLED.

ADDRESS CHECK FOR DEVICE, DEV DURING INTERRUPT SERVICE

WHERE DEV IS THE DEVICE FOR WHICH THE INTERRUPT I/O SERVICE ROUTINE ATTEMPTED TO MAKE A REFERENCE OUTSIDE OF THE JOB AREA.

ILL MEM REF FROM MONITOR/USER LOC M

WHERE M IS THE LOCATION OF THE INSTRUCTION WHICH ATTEMPTED AN ILLEGAL MEMORY REFERENCE OUTSIDE ITS PROTECTED AREA.

PC EXCEEDS MEM BOUND AT MONITOR/USER LOC M

WHERE M IS THE LOCATION WHICH THE PROGRAM ATTEMPTS TO EXECUTE.

NON EX MEM AT MONITOR/USER LOC M

WHERE M IS THE LOCATION OF AN INSTRUCTION ATTEMPTING TO REFERENCE NON-EXISTENT MEMORY.

PDL OV AT MONITOR/USER LOC M

WHERE M IS THE LOCATION OF THE INSTRUCTION CAUSING THE PUSH DOWN OVERFLOW.

ILL BLOCK NO BUFFER AT USER LOC M

WHERE M IS THE LOCATION IN THE USER AREA OF THE BUFFER WHICH CONTAINS THE ILLEGAL BLOCK NUMBER (GREATER THAN 1100 OCTAL).

CUSP COMMAND FORMATS

THE GENERAL CONSOLE INPUT COMMAND STRING IS:

DEST FILE 1, DEST FILE 2 ← SOURCE FILE 1,...SOURCE FILE N

THE COMMAND STRING CONSISTS OF ONE OR MORE DESTINATION FILE DESCRIPTORS SEPARATED FROM ONE OR MORE SOURCE FILE DESCRIPTORS BY AN "←", WHERE THE ALLOWABLE COMPONENTS OF A FILE DESCRIPTOR ARE A DEVICE NAME, A FILE NAME, A FILE NAME EXTENSION, THE MODE, AND SWITCHES, WHICH APPLY FOR THE FILE DESCRIPTOR CONTAINING THE SWITCHES PLUS ALL DESCRIPTORS TO THE RIGHT, SEPARATED FROM EACH OTHER BY SPECIAL DELIMITERS WHICH IDENTIFY THEM.

FILE DELIMITERS

()	ENCLOSE PROGRAM SWITCHES.
:	FOLLOWS A DEVICE NAME.
,	FOLLOWS A FILE DESCRIPTOR.
/	PRECEDES A MODE.
.	PRECEDES A FILE NAME EXTENSION
←	SEPARATES DESTINATION FILES FROM SOURCE FILES
(CR)	DELIMITS A FILE DESCRIPTOR

DEVICE NAMES

LPT	LINE PRINTER
CDR	CARD READER
PTR	PAPTER TAPE READER
PTP	PAPER TAPE PUNCH
TTYØ...N	TELETYPE
DTAØ...N	DECTAPE
MTAØ...N	MAGNETIC TAPE
DIS	DISPLAY

FILE NAME EXTENSIONS (NEED NOT BE SPECIFIED)

MAC	MACRO SOURCE
FOL	FOLA SOURCE
FOR	FORTRAN SOURCE
LIB	LIBRARY
REL	RELOCATABLE BINARY FILE

DMP	DUMP FILE
LST	LISTING
MAP	STORAGE MAP

FILE MODES

A	ASCII
B	CHECKSUM BINARY
S	ASCII SEQUENCED
L	ASCII LINE
D	DUMP FILE

C H A P T E R I I I

J O B A R E A U S E O F S Y S T E M

GENERAL

THE USER'S PROGRAM AREA IS THE PORTION OF CORE MEMORY AVAILABLE TO THE USER WHEN HIS PROGRAM IS RUNNING. THE FIRST ADDRESS OF THIS AREA IS DEFINED BY THE CONTENTS OF THE MEMORY RELOCATION REGISTER WHEN THE USER'S PROGRAM RUNS, AND THE LAST ADDRESS IS DEFINED BY THE CONTENTS OF THE MEMORY PROTECTION REGISTER. THE USER'S AREA IS SHOWN IN THE DIAGRAM, JOB AREA STORAGE

IN ORDER TO MAKE THE FOLLOWING DISCUSSION CLEARER, IT IS NECESSARY TO DEFINE SOME TERMS AND TO REVIEW BRIEFLY THE OPERATION OF THE PDP-6 TIME SHARING HARDWARE. THE USER'S PROGRAM RUNS WHILE THE PDP-6 IS IN A SPECIAL MODE CALLED "USER'S MODE". IN THIS MODE, THE CONTENTS OF A REGISTER IN THE ARITHMETIC PROCESSOR CALLED THE MEMORY RELOCATION REGISTER IS ADDED TO EACH MEMORY ADDRESS BEFORE THE ADDRESS IS SENT TO THE MEMORY SYSTEM. THE ADDRESS BEFORE THIS ADDITION TAKES PLACE IS CALLED THE RELATIVE ADDRESS, AFTER THE ADDITION THE ADDRESS IS CALLED THE ABSOLUTE ADDRESS. ALL ADDRESSES IN THE USER'S AREA ARE RELATIVE ADDRESSES.

TO TAKE ADVANTAGE OF THE FAST ACCUMULATORS, ACCUMULATOR ADDRESSES (Ø THROUGH 17) ARE NOT RELOCATED. THUS, RELATIVE LOCATIONS Ø THROUGH 17 CANNOT BE REFERENCED BY THE USER'S PROGRAM. (THE TIME SHARING MONITOR SAVES THE USER'S ACCUMULATORS IN THIS AREA WHILE THE SYSTEM IS SERVICING A PROGRAMMED OPERATOR REQUEST.) THE CONTENTS OF THE MEMORY PROTECTION REGISTER ARE COMPARED WITH THE EIGHT HIGH-ORDER BITS OF EACH RELATIVE ADDRESS AS THE USER'S PROGRAM RUNS. IF THE RELATIVE ADDRESS EXCEEDS THE CONTENTS OF THE MEMORY PROTECTION REGISTER, THE MEMORY VIOLATION FLAG IS SET AND CONTROL TRAPS TO THE TIME SHARING MONITOR. SOME INSTRUCTIONS TRAP TO ABSOLUTE LOCATION 4Ø WHEN EXECUTED IN THE USER MODE. THESE INSTRUCTIONS, JRST INSTRUCTIONS WHICH ATTEMPT TO HALT, AND ALL UNDEFINED OPERATION CODES. THE ACTION ON TRAPS TO LOCATION 4Ø IS THE SAME AS THE ACTION FOR PROGRAMMED OPERATORS IN EXECUTIVE MODE, WITH THE ADDITION THAT THE USER MODE IS TURNED OFF WHEN THE TRAP OCCURS. PROGRAM INTERRUPTS ALSO TRAP TO THE ABSOLUTE LOCATIONS NORMALLY USED IN EXECUTIVE MODE AND TURN OFF THE USER MODE.

SYSTEM PARAMETERS IN JOB AREA

THE FIRST 14Ø (OCTAL) REGISTERS OF THE USER JOB AREA COMPRISE THE JOB DATA AREA WHICH IS RESERVED FOR STORING SPECIFIC INFORMATION CONCERNING THE JOB, SUCH AS THE STARTING ADDRESS OF THE USER'S PROGRAM AND THE IN/OUT DEVICE ASSIGNMENTS MADE FOR THIS PROGRAM. ALSO IN THIS AREA ARE REGISTERS FOR SAVING ACCUMULATORS DURING REQUESTS FOR SERVICE TO THE EXECUTIVE SYSTEM USING THE PROGRAMMED OPERATORS OR WHILE SOME OTHER USER'S PROGRAM IS RUNNING. AN UNDERSTANDING OF THE FUNCTIONS OF THE SPECIFICALLY RESERVED LOCATIONS IS REQUIRED FOR EFFICIENT USE OF THE SERVICES OF THE TIME SHARING MONITOR

THESE LOCATIONS ARE RESTORED TO THE ORIGINAL VALUES WHENEVER THE USER'S PROGRAM BECOMES INACTIVE (DUE TO WAITING FOR INPUT/OUTPUT OR TO RUNNING ANOTHER USER'S PROGRAM). THESE REGISTERS ARE LABELED "PROTECTED" IN THE DESCRIPTIONS BELOW. THE REGISTERS MNEMONIC ASSIGNMENTS ARE MADE ON THE FILE , JOBDEF, AND JOBDEF SHOULD BE EITHER ASSEMBLED WITH PROGRAMS WHICH REFERENCE THIS AREA, OR EXTERNALS SHOULD BE USED TO NAME THE REGISTERS. A LISTING OF JOBDEF, AND THE ASSIGNMENTS OF UUOS TO SYSTEM FUNCTIONS, OPDEF, IS GIVEN IN THE APPENDIX.

JOBAC

THE BLOCK OF CORE MEMORY FROM USER'S LOCATION Ø THROUGH 17 CANNOT BE REFERENCED BY THE USER'S PROGRAM BECAUSE THE ARITHMETIC PROCESSOR SUBSTITUTES ABSOLUTE REGISTERS Ø THROUGH 17. THE BLOCK OF 2Ø REGISTERS STARTING AT JOBAC, DEFINED AS USER'S Ø, STORES THE USER'S ACCUMULATORS WHILE SERVICING EXECUTIVE REQUESTS VIA THE PROGRAMMED OPERATORS.

JOBDAC

A SECOND BLOCK OF 2Ø REGISTERS STORES THE ACCUMULATORS WHILE A JOB IS INACTIVE. SINCE A JOB MAY BECOME INACTIVE EITHER WHILE THE USER'S PROGRAM IS COMPUTING OR WHILE THE EXECUTIVE PROGRAM IS PROCESSING A USER'S REQUEST, THE INFORMATION SAVED MAY BELONG TO TO EITHER THE USER'S PROGRAM OR THE EXECUTIVE PROGRAM.

JOBUUO

TWO REGISTERS AT USER'S 4Ø AND 41 ARE USED FOR PROCESSING USER'S PROGRAMMED OPERATORS (ØØ THROUGH 37). THE TIME SHARING MONITOR RESERVES PROGRAMMED OPERATORS 4Ø THROUGH 77 FOR VARIOUS PURPOSES. THE REMAINING PROGRAMMED OPERATORS ARE EXECUTED AS DESCRIBED IN THE PDP-6 PROGRAMMING HANDBOOK USING USER'S 4Ø AND 41 INSTEAD OF ABSOLUTE 4Ø AND 41.

JOBPC

THE LOCATION JOBPC SAVES THE PROGRAM COUNTER AND ARITHMETIC PROCESSOR FLAGS WHILE THE JOB IS INACTIVE.

JOBPDP

PROTECTED. THE EXECUTIVE PROGRAM MAINTAINS A PUSHDOWN LIST IN THE LAST 15 (OCTAL) LOCATIONS OF THE USER'S JOB DATA AREA. JOBPDP CONTAINS THE PUSHDOWN POINTER FOR THIS PUSHDOWN LIST.

JOBREL

PROTECTED. THE LEFT HALF OF JOBREL IS UNUSED. THE RIGHT HALF CONTAINS THE HIGHEST RELATIVE LOCATION AVAILABLE TO THE USER; IN PARTICULAR, THE CONTENTS OF THE MEMORY PROTECTION REGISTER.

JOBUXT

PROTECTED. JOBUXT CONTAINS THE PROGRAM COUNTER AND PROCESSOR FLAGS FOR RETURNING TO THE USER'S PROGRAM AFTER THE EXECUTIVE IS CALLED BY A PROGRAMMED OPERATOR.

JOBSAV

PROTECTED. JOBSAV IS A GENERAL PURPOSE TEMPORARY REGISTER FOR USE BY THE PROGRAMMED OPERATOR PROCESSOR.

JOBJDA

PROTECTED. A BLOCK OF 2Ø REGISTERS STARTING AT JOBJDA CONTAINS THE CORRELATION BETWEEN INPUT/OUTPUT DEVICES AND THE DEVICE CHANNEL ASSIGNMENTS REQUESTED BY THE USER'S PROGRAM BY MEANS OF THE INIT COMMAND.

JOBDDT

JOBDDT CONTAINS THE STARTING ADDRESS OF DDT, THE USER'S DEBUGGING PROGRAM. IN THE EVENT OF A SYSTEM OR USER'S ERROR (FOR EXAMPLE, A USER'S MEMORY BOUNT VIOLATION) THE EXECUTIVE SYSTEM PRINTS AN ERROR MESSAGE ON THE USER'S TELETYPE CONSOLE AND RETURNS CONTROL TO THE USER'S DEBUGGING PROGRAM.

JORSYM

JOBSYM CONTAINS A POINTER TO THE SYMBOL TABLE CREATED BY THE LINKING LOADER. DDT USES THIS SYMBOL TABLE FOR PRINTING AND INTERPRETING SYMBOLIC ADDRESSES. THE LEFT HALF OF JOBSYM CONTAINS A NEGATIVE COUNT OF THE LENGTH OF THE SYMBOL TABLE AND THE RIGHT HALF CONTAINS THE LOWEST REGISTER USED. SEE THE DESCRIPTION OF THE LINKING LOADER FOR THE EXACT FORMAT OF THE SYMBOL TABLE.

JOBSA

JOBSA CONTAINS TWO ADDRESSES. THE LEFT HALF CONTAINS THE HIGHEST REGISTER USED BY THE PROGRAM. THE SYSTEM PUSHDOWN LIST IS PLACED IN THE AREA IMMEDIATELY FOLLOWING THIS HIGHEST REGISTER. THE RIGHT HALF CONTAINS THE STARTING ADDRESS OF THE USER'S PROGRAM. INITIALLY, THIS ADDRESS IS SET TO THE LATEST STARTING ADDRESS SEEN BY THE LOADER.

JOBFF

JOBFF CONTAINS THE ADDRESS OF THE FIRST FREE LOCATION FOLLOWING THE USER'S PROGRAM. THIS REGISTER IS UPDATED BY THE INPUT/OUTPUT SERVICE ROUTINES WHENEVER AN INPUT/OUTPUT BUFFER IS CREATED. THE USER'S PROGRAM SHOULD RESTORE THE CONTENTS OF JOBFF AFTER FINISHING ALL USE OF A BUFFER IN ORDER TO RECOVER THE CORE MEMORY USED.

THE REMAINING LOCATIONS IN THE JOB DATA AREA ARE RESERVED FOR FUTURE EXPANSION AND DEVELOPMENT. PROPOSED USAGE FOR SOME OF THESE LOCATIONS ARE:

JOBTEM

THREE LOCATIONS BEGINNING WITH JOBTEM TO BE RESERVED FOR GENERAL PURPOSE TEMPORARY SYSTEM STORAGE.

JOBINF

THE LOCATION JOBINF WOULD CONTAIN THE JOB NUMBER ASSIGNED TO THE USER IN BITS 0 THROUGH 11.

JOBCDP

JOBCDP WOULD CONTAIN A BYTE POINTER THAT POINTS TO A COMMAND STRING. THE TIME SHARING EXECUTIVE WOULD ACCEPT A COMMAND TYPED ON THE USER'S TELETYPE AND DETERMINE WHICH SYSTEM PROGRAM MUST BE CALLED TO PROCESS THE COMMAND. THE EXECUTIVE WOULD THEN LOAD THE SYSTEM PROGRAM AND SET UP JOBCDP TO POINT TO THE COMMAND STRING. THE SYSTEM PROGRAM THEN EXAMINES THE COMMAND STRING, ONE CHARACTER AT A TIME, AND PERFORMS THE COMMAND OPERATION.

JOBTRP

A BLOCK OF 20 (OCTAL) LOCATIONS BEGINNING WITH JOBTRP WILL CONTAIN INSTRUCTIONS TO BE EXECUTED WHEN AN INPUT/OUTPUT ERROR OR UNUSUAL CONDITION OCCURS. ONE LOCATION IS ASSOCIATED WITH EACH DEVICE CHANNEL. IF THE TRAP LOCATION CONTAINS 0, THEN THE PROGRAM IS DISMISSED IF AN ATTEMPT IS MADE TO TRAP ON THAT CHANNEL. OTHERWISE

A JSR INSTRUCTION IS PRESUMED TO BE IN THE TRAP LOCATION AND CONTROL IS TRANSFERRED TO THE SPECIFIED TRAP SUBROUTINE AS IF BY A JSR REGARDLESS OF WHAT INSTRUCTION IS ACTUALLY AT THE TRAP LOCATION.

JOBUSY

JOBUSY WILL CONTAIN A POINTER TO A LIST OF SYMBOLS THAT ARE UNDEFINED AT THE END OF THE LOADING PROCESS. THE USER WILL BE ABLE TO DEFINE ALL SUCH SYMBOLS BY MEANS OF DDT. DDT WILL AUTOMATICALLY ADJUST EACH LOCATION WHERE THE NEWLY DEFINED SYMBOLS ARE CALLED FOR.

IN ADDITION TO THE NAMED LOCATIONS ABOVE, ALL LOCATIONS BETWEEN JOBFF AND 140 (OCTAL) (NOT INCLUSIVE) ARE RESERVED FOR THE SYSTEM PUSHDOWN LIST. THE SYSTEM PROGRAM LOADER ALWAYS STARTS LOADING AT USER'S LOCATION 140.

LOADING USER PROGRAMS

A COPY OF THE LOADER FROM THE SYSTEM TAPE IS READ INTO A JOB AREA BY THE "GET" COMMAND AND IS STARTED BY THE "START" COMMAND, WHICH TRANSFERS COMMAND CONTROL TO THE LOADER. THE LOADER DESCRIPTION LISTS ITS COMMANDS. THE LOADER REMAINS IN CORE UNTIL ALL FILES ARE LOADED. ONCE EXECUTION IS REQUESTED, THE LOADER CALLS THE SYSTEM FUNCTION "CALL" SIXBIT /LDRBLT/ WHICH TRANSFERS THE LOADED PROGRAM DOWN OVER THE LOADER.

THE LOADER FIRST LOADS FORTRAN COMMON AT LOCATION 140, FOLLOWED BY HIS PROGRAMS, LIBRARY FILES, DDT, ETC.

THE SYMBOL TABLE IS LOADED FROM THE TOP OF MEMORY DOWN TOWARD THE PROGRAM. THE SYMBOL TABLE FORMAT, DATA FORMAT FOR THE LOADER, ETC. ARE GIVEN IN THE LOADER DESCRIPTION.

SYSTEM PUSHDOWN LIST

IN ADDITION TO SPECIFIC-PURPOSE LOCATIONS DESCRIBED BELOW, THE TIME SHARING MONITOR MAINTAINS A PUSHDOWN LIST IN THE USER'S AREA FOR GENERAL PURPOSE TEMPORARY STORAGE. ALL THE TEMPORARY STORAGE REQUIRED FOR SERVING EACH USER IS CONTAINED WITHIN THE USER'S AREA AND THUS, THE TIME SHARING MONITOR CAN SWITCH TO SERVING ANOTHER USER AT ANY TIME WITHOUT FORGETTING WHAT IT WAS DOING FOR THE ORIGINAL USER. SOME OF THE SPECIAL PURPOSE LOCATIONS DESCRIBED BELOW ARE SAVED IN REGISTERS WITHIN THE TIME SHARING MONITOR SO THAT THE USER'S PROGRAM CANNOT MODIFY THE VALUES.

MONITOR INDEPENDENCE

AS FUTURE IDEAS AND FEATURES ARE ADDED TO THE TIME SHARING MONITOR, EVERY EFFORT WILL BE MADE TO AVOID OBSOLETING OLDER PROGRAM; HOWEVER, SOME CHANGES WILL INEVITABLY REQUIRE USERS TO MODIFY THEIR PROGRAMS. THEREFORE, EACH PROGRAM SHOULD BE WRITTEN IN TWO SEPARATE SECTIONS. ONE SECTION IS THE COMPUTATIONAL PROGRAM THAT NEED NEVER CHANGE, REGARDLESS OF CHANGES IN THE TIME SHARING MONITOR. THE OTHER SECTION IS THE INITIALIZATION AND SYSTEM INTERFACE SECTION; THE CODING THAT COMMUNICATES DIRECTLY WITH THE TIME SHARING MONITOR. WITH ALL SYSTEM-DEPENDENT CODING ISOLATED INTO ONE SECTION OF THE PROGRAM, UPDATING

PROGRAMS TO RUN IN FUTURE SYSTEMS WILL BE A RELATIVELY EASY CHORE.

JOBDEF AND OPDEF TAPES SHOULD BE USED, RATHER THAN ABSOLUTE ASSIGNMENTS FOR NAMING SYSTEM REGISTERS OR FUNCTIONS.

C H A P T E R I V

U S E R I N P U T O U T P U T I N S T R U C T I O N S

GENERAL

THE INPUT/OUTPUT SERVICE ROUTINES PROVIDE THE INPUT/OUTPUT FACILITIES FOR PROGRAMS RUNNING UNDER THE PDP-6 TIME-SHARING MONITOR SYSTEM. THE OBJECT OF THESE ROUTINES IS TO SIMPLIFY CODING FOR THE VARIOUS INPUT/OUTPUT DEVICES AS MUCH AS POSSIBLE AND YET TO PROVIDE FLEXIBILITY FOR PERFORMING THE MOST SOPHISTICATED FUNCTIONS OF INPUT/OUTPUT. FULL ADVANTAGE IS TAKEN OF THE PROGRAM INTERRUPT FACILITY OF THE PDP-6 TO OVERLAP IN/OUT OPERATIONS WITH COMPUTATION.

THE USER'S PROGRAM COMMUNICATES WITH THE I/O SERVICE ROUTINES BY MEANS OF THE PROGRAMMED OPERATORS. A PROGRAMMED OPERATOR AND THE MACHINE INSTRUCTION SET, UUO (UNUSED OPERATION CODE) ARE SYNONOMOUS. TYPICAL COMMANDS ARE TO SET UP A BUFFER AREA, TO TRANSMIT A BUFFER FULL OF DATA TO AN OUTPUT DEVICE, OR TO RELEASE A DEVICE FOR USE BY OTHER USERS.

DEVICE DEPENDENT OPERATIONS

SOME I/O DEVICES REQUIRE SPECIAL COMMANDS IN addition TO THOSE DESCRIBED BELOW. FOR EXAMPLE, MAGNETIC TAPE REQUIRES A REWIND COMMAND, DECTAPE REQUIRES BLOCK NUMBER COMMANDS. THESE COMMANDS AND OTHER INDIVIDUAL CHARACTERISTICS ARE DESCRIBED SEPARATELY FOR EACH DEVICE. COMMANDS IRRELEVANT TO A PARTICULAR DEVICE ARE IGNORED. FOR EXAMPLE, A FILE DIRECTORY MANIPULATION COMMAND (REQUIRED FOR DECTAPE OPERATIONS) WHEN GIVEN TO THE LINE PRINTER IS IGNORED. THUS, TO CHANGE A PROGRAM FROM OUTPUTTING TO THE LINE PRINTER, THE USER NEED ONLY CHANGE THE DEVICE NAME IN THE INITIALIZATION COMMAND. THE MNEMONICS FOR THE FOLLOWING COMMANDS AND THEIR CODE ASSIGNMENTS ARE ON THE SYSTEM FILE, OPDEF. OPDEF IS IN THE APPENDIX.

THE COMMANDS CAN BE SUMMARIZED AS FOLLOWS:

FILE INITIALIZATION

RESET	SYSTEM RESET FOR JOB
ASSIGN	GET A DEVICE, AND RENAME IT
INIT	ATTACH FREE DEVICE TO A JOB CHANNEL
INBUF,OUTBUF	SETUP BUFFERS (OPTIONAL)
LOOKUP,ENTER	FIND OR ASSIGN FILE NAMES

FILE DATA TRANSMISSION (NON-DUMP)

INPUT, OUTPUT	

FILE MANIPULATION (SPECIAL DEVICES)

INPUT,OUTPUT	(DUMPMODE)
USETO,USETI	SET IO FILE BLOCKS
UGETF	GET FREE FILE NUMBER
MTAPE	PERFORM AN ACTION ON MAGTAPE

FILE STATUS EXAMINATION

STATUS,STATZ,STATO	CHECK DEVICE STATUS

FILE TERMINATION

CLOSE	TERMINATES CURRENT FILE

RELEAS	RETURNS DEVICE TO SYSTEM, UNLESS ASSIGNED
DEASSIGN	RETURNS DEVICE TO SYSTEM
EXIT	

THE ABOVE COMMAND PAIRS, RESET-EXIT, ASSIGN-DEASSIGN, INIT-RELEASE, AND LOOKUP/ENTER-CLOSE, EACH WORK AT A DEVICE, CHANNEL-DEVICE, OR FILE LEVEL RESPECTIVELY.

RING BUFFERS

CORE MEMORY SERVES AS AN INTERMEDIATE BUFFER BETWEEN THE USER'S PROGRAM AND THE IN/OUT DEVICE. AN I/O BUFFER CONSISTS OF A HEADER BLOCK (THREE WORDS FOR BOOKKEEPING) AND A DATA STORAGE AREA. THE STORAGE AREA IS SUBDIVIDED INTO ONE OR MORE INDIVIDUAL BUFFERS ARRANGED IN A RING. TO BE SPECIFIC, ASSUME THAT THE DEVICE IS AN INPUT DEVICE. ONCE INPUT OPERATIONS ARE UNDER WAY, THE I/O SERVICE ROUTINES FILL A BUFFER, MAKE THE BUFFER AVAILABLE TO THE USER'S PROGRAM, ADVANCE TO THE NEXT BUFFER, AND BEGIN TO FILL IT IF IT IS FREE. AFTER FILLING THE LAST BUFFER IN THE STORAGE AREA, THE NEXT BUFFER TO BE FILLED IS THE FIRST BUFFER IN THE AREA; HENCE THE NAME RING BUFFER.

THE USER'S PROGRAM FOLLOWS ALONG BEHIND, EMPTYING THE NEXT BUFFER IF IT IS FULL, OR WAITING FOR THE NEXT BUFFER TO FILL UP IF IT IS NOT YET FULL.

FOR OUTPUT, THE USER'S PROGRAM AND THE I/O ROUTINES EXCHANGE ROLES, THE USER FILLING THE BUFFERS AND THE I/O ROUTINES EMPTYING THEM.

BUFFER STRUCTURE

THE BUFFER STRUCTURE CONSISTS OF TWO SEPARATE SEGMENTS, THE HEADER AND THE STORAGE AREA. THE HEADER CONSISTS OF THE FOLLOWING THREE WORDS:

1) THE ADDRESS OF THE SECOND WORD OF THE BUFFER CURRENTLY IN USE BY THE USER'S PROGRAM IN THE RIGHT HALF. BIT Ø=1 MEANS NO INPUT OR OUTPUT HAS OCCURRED FOR THIS RING.

2) A BYTE POINTER TO THE LAST ITEM REFERENCED IN THE BUFFER.

3) A COUNT OF THE NUMBER OF ITEMS REMAINING IN THE BUFFER.

FOR MOST APPLICATIONS, THE USER'S PROGRAM FINDS ALL THE INFORMATION IT REQUIRES ABOUT THE STATE OF THE BUFFER AREA IN THE HEADER. WHEN REFERRING TO A BUFFER BY NAME, THE ADDRESS OF THE BUFFER HEADER, NOT THE STORAGE AREA, IS IMPLIED.

THE STORAGE AREA IS SUBDIVIDED INTO A NUMBER OF INDIVIDUAL, EQUAL-LENGTH BUFFERS. THE SIZE OF EACH BUFFER DEPENDS ON THE REQUIREMENTS OF THE DEVICE SERVICED BY THE BUFFER. THE FIRST TWO WORDS OF EACH BUFFER ARE RESERVED FOR SPECIAL FUNCTIONS; THE REMAINING WORDS ARE FOR DATA STORAGE. THE FIRST TWO WORDS ARE:

1) RESERVED FOR A BLOCK NUMBER FROM FIXED ADDRESS DEVICES SUCH AS DECTAPE OR DRUM.

2) IN THE RIGHT HALF, THE ADDRESS OF THE SECOND WORD OF THE NEXT BUFFER IN THE RING. IN THE LEFT HALF, BIT Ø IS A FLAG AS EXPLAINED BELOW; BITS 1 THROUGH 17 CONTAIN THE BUFFER SIZE IN WORDS (EXCLUDING THE TWO WORDS OF OVERHEAD).

BIT Ø OF THE SECOND WORD OF THE BUFFER, CALLED THE USE BIT, IS A FLAG THAT INDICATES WHETHER THE BUFFER CONTAINS ACTIVE DATA. THIS BIT IS SET TO 1 BY THE I/O ROUTINES WHEN THE BUFFER IS FULL ON INPUT (OR BEING EMPTIED ON OUTPUT) AND SET TO Ø WHEN THE BUFFER IS EMPTY ON OUTPUT (OR BEING FILLED ON INPUT). THE USE BIT PREVENTS THE I/O ROUTINES AND THE USER'S PROGRAM FROM INTERFERING WITH THE USER'S PROGRAM BY ATTEMPTING TO USE THE SAME BUFFER SIMULTANEOUSLY.

IN ALL DATA PROCESSING MODES, THE FIRST DATA WORD OF THE BUFFER IS RESERVED FOR A COUNT OF THE NUMBER OF DATA WORDS IN THE BUFFER (EXCLUDING ITSELF). THE LEFT HALF OF THIS WORD IS RESERVED FOR OTHER BOOKKEEPING PURPOSES DEPENDING ON THE PARTICULAR DEVICE.

RESET COMMAND

TO CLEAR ALL PREVIOUS DEVICE ASSIGNMENTS AND PREPARE THE I/O SERVICE ROUTINES FOR RECEIVING FURTHER INITIALIZATION COMMANDS, EXECUTE THE INSTRUCTION:

```
CALL [SIXBIT /RESET/]
```

THIS MUST BE THE FIRST INSTRUCTION IN EACH PROGRAM.

ASSIGN COMMAND

THE ASSIGN COMMAND IS GIVEN TO ATTACH A PHYSICAL DEVICE TO A JOB, AND AT THIS TIME RENAME THE DEVICE BY A LOGICAL NAME, AN ARBITRARY SIXBIT NAME. IF MORE THAN ONE PHYSICAL DEVICE MAY BE USED (E.G. ANY 1 OF N TELETYPES). THE NUMBER OF THE DEVICE IS SUPPLIED BY THE SYSTEM. AN EXAMPLE IS:

```
ASSIGN  D, ADR
ERROR RETURN ; NOT AVAILABLE
SUCCESSFUL RETURN


ADR:      SIXBIT /DESIRED PHYSICAL NAME/
ADR+1:    SIXBIT /DESIRED LOGICAL NAME/
C(ADR) MAY CONTAIN A 3 CHARACTER DEVICE NAME WHICH THE
;SYSTEM SUPPLIES.  THE FIRST FREE PHYSICAL DEVICE IS USED.
;AT COMPLETION THE C(AC-D) CONTAIN A ;SIXBIT PHYSICAL NAME
```

IF A PHYSICAL DEVICE IS ALREADY ATTACHED TO THE JOB, AND AN ASSIGN IS GIVEN, THE NEW NAME IS USED. IF NO RENAMING IS TO OCCUR, C(ADR+1)=Ø.

INITIALIZATION

INIT COMMAND

EACH DEVICE IS INITIALIZED AND BUFFER HEADERS SPECIFIED BY EXECUTING THE COMMAND:

```
INIT D, MODE
SIXBIT /DEVICE NAME/
XWD OBUF,IBUF
ERROR RETURN
NORMAL RETURN
```

INIT COMMAND PERFORMS THE FOLLOWING FUNCTIONS:

1) THE 4 BIT CHANNEL NUMBER D IS ASSIGNED TO THE DEVICE NAME APPEARING IN THE SIXBIT STATEMENT. HENCEFORTH, THE I/O SERVICE ROUTINES INTERPRET THE NUMBER D AS REFERRING TO THAT DEVICE.

2) THE DATA PROCESSING MODE IS SELECTED BY MODE (SEE NEXT SECTION).

3) THE BUFFER AREA HEADERS ARE SPECIFIED TO THE DEVICE SERVICE ROUTINES. EACH HEADER MUST BE THREE LOCATIONS RESERVED BY THE USER IN HIS PROGRAM. ONLY THOSE HEADERS WHICH ARE TO BE USED NEED BE SPECIFIED; THE OUTPUT HEADER NEED NOT BE SPECIFIED IF THE PROGRAM PERFORMS ONLY INPUT FROM A DEVICE) THE DEVICE IS EITHER A LOGICAL OR PHYSICAL NAME. THE FIRST SEARCH IS FOR A LOGICAL NAME, FOLLOWED BY A PHYSICAL NAME.

INIT DATA MODES

ONE ITEM OF INFORMATION SPECIFIED IN THE INIT COMMAND IS THE DATA PROCESSING MODE. THE FOLLOWING MODES ARE AVAILABLE:

OCTAL CODE	MNEMONIC	MEANING
0	A	ASCII, 7-BIT CHARACTERS, PACKED AS BY THE BYTE COMMANDS.
1	AL	ASCII LINE, SAME AS A EXCEPT THAT THE BUFFER IS TERMINATED BY A FORM-FEED, VERTICAL TAB, OR LINE-FEED.
5	AM	ALT MODE, SAME AS AL EXCEPT THAT THE LINE IS TERMINATED ALSO ON ALT-MODE.
10	I	IMAGE, A DEVICE DEPENDENT MODE, THE BUFFER IS FILLED WITH UNPROCESSED DATA EXACTLY AS SUPPLIED BY THE DEVICE
14	B	BINARY, A BLOCKED FORMAT CONSISTING OF A WORD COUNT (THE FIRST WORD OF THE BUFFER) FOLLOWED BY N 36-BIT DATA WORDS. CHECK-SUMMING IS DONE AUTOMATICALLY BY THE SERVICE ROUTINES OR BY THE DEVICE ITSELF.

		N IS IN RIGHT HALF OF FIRST WORD, AND IF CHECKSUM IS COMPUTED (CARDS, PAPER TAPE) IT IS IN BITS 6-17. THE 12 BIT FOLDED CHECKSUM IS COMPUTED BY SUMMING THE 36 BIT DATA (2'S COMPLEMENT ADD), FOLLOWED BY A 1'S COMPLEMENT SUM OF BITS 0-11, 12-23, 24-35.
15, 16, 17	D	DUMP, DATA IS TRANSMITTED BETWEEN ANY CONTIGUOUS BLOCK OF CORE AND THE I/O DEVICE. IN DUMP MODE, RING BUFFERS ARE NOT USED. SEE SPECIFIC DUMP DESCRIPTIONS FOR EACH DEVICE. THE CONTROL FOR THE SIZE, LOCATION IS FROM A COMMAND LIST.

INIT DATA CONTROL MODES
THE USER MAY SPECIFY CHARACTERISTICS THAT FURTHER DEFINE THE DATA TRANSFER.

BIT	NAME	ACTION
31	IOWC	IO WORD COUNT. IF A 1, COMMANDS THE SERVICE ROUTINES TO ACCEPT THE WORD COUNT IN THE FIRST DATA WORD OF THE BUFFER (AS COMPUTED BY THE USER) INSTEAD OF COMPUTING A WORD COUNT FROM THE CONTENTS OF THE BYTE POINTER IN THE BUFFER HEADER.
30	IOCON	IO CONTINUOUS MODE. IF A 1, SPECIFIES STOPPING THE DEVICE AFTER EACH BUFFER IS FILLED (INPUT ONLY).
29	IORDEL	IO READ AND DELETE A FILE IF A 1. (DECTAPE, DRUM, DISC.)
29	IORCK	IO NO READ CHECK. SUPPRESS THE ERROR CORRECTING IF A 1. I.E. NORMAL MODE IS TO RE-READ A RECORD 3 TIMES BEFORE SETTING ERRO← OR TO CONTINUE TO RE-WRITE UNTIL A GOOD RECORD IS WRITTEN. (MAG TAPE, DISC, DECTAPE.)
27-28	IODENS	IO DENSITY FOR MAG TAPE. 0,2=556 BPI. 1=200 BPI. 3=800 BPI.
26	IOPAR	IO PARITY CHECKING FOR MAG TAPE. 0 FOR ODD, OR NORMAL PARITY. 1 FOR BCD OR EVEN PARITY.

INBUF COMMAND
TO CAUSE AN INPUT BUFFER AREA TO BE SET, ISSUE THE COMMAND:

```
INBUF  D,N
```

WHERE N SPECIFIES AN N BUFFER RING. THE STORAGE IS TAKEN FROM THE FREE STORAGE FOLLOWING THE PROGRAM. THE LOCATION JOBFF (A PERMANENT GLOBAL SYMBOL DEFINED BY IN SYSTEM TAPE, JOBDEF) CONTAINS

THE ADDRESS OF THE FIRST FREE LOCATION FOLLOWING THE PROGRAM. IF THE USER WISHES TO ABANDON A BUFFER FOR ONE DEVICE AND SET UP BUFFER FOR ANOTHER DEVICE, THE PROGRAM SHOULD RESTORE THE CONTENTS OF JOBFF TO THE VALUE CONTAINED BEFORE THE FIRST BUFFER WAS SET UP. THEN THE I/O SERVICE ROUTINES CAN REUSE THE OLD BUFFER AREA IN CORE MEMORY FOR SETTING UP THE NEW BUFFER.

OUTBUF COMMAND

AN OUTPUT BUFFER AREA MAY BE SET UP BY:

```
OUTBUF  D,N
```

IF NO BUFFER AREA IS SET UP WHEN THE FIRST INPUT OR OUTPUT COMMAND IS ISSUED, A ONE BUFFER RING IS SET UP AUTOMATICALLY FROM THE FREE STORAGE FOLLOWING THE PROGRAM.

A DEVICE MAY BE INITIALIZED SEPARATELY ON TWO DEVICE CHANNELS, IF ONE INIT COMMAND SPECIFIES THE OUTPUT BUFFER HEADER AND THE OTHER SPECIFIES THE INPUT BUFFER HEADER. ALL FURTHER COMMANDS MAY THEN REFER TO EITHER CHANNEL NUMBER ; THE OUTPUT COMMAND CAN USE THE CHANNEL THAT WAS INITIALIZED FOR INPUT FOR EXAMPLE.

LOOKUP, ENTER COMMANDS

THESE COMMANDS ARE GIVEN TO EACH DEVICE, THEY SPECIFY A FILE NAME, FILE NAME EXTENSION, MODE FOR A FILE. THE COMMANDS HAVE NO EFFECT FOR SINGLE FILE DEVICES, E.G. PTR, PTP, TTY, LPT, CDR, BUT ARE USED WITH DTA, MTA, DRUM, DISC, ETC. WHICH HAVE MULTIPLE FILES/DEVICE AND A DIRECTORY TO HOLD FILE NAMES. AN ENTER MUST BE GIVEN BEFORE OUTPUT, AND LOOKUP BEFORE INPUT. A LOOKUP COMMAND FORMAT EXAMPLE IS:

```
        LOOKUP D,E          ;SEARCH DIRECTORY FOR
        ITEM NOT FOUND      ;ENTRY SPECIFIED BY C(E).
        OK RETURN           ;

    ;SEARCHES C(E,E+1, E+2, E+3,) FOR FILE NAME, (6-SIXBIT)
    ;FILE NAME EXTENSION (3-SIXBIT).
    ;AND FILLS IN BLANK INFORMATION NEEDED
    ;TO MANIPULATE THE FILE
```

AN ENTER COMMAND FORMAT EXAMPLE IS :

```
        ENTER D,E       ;ENTER DIRECTORY INFO. FROM
        FULL DIRECTORY  ;C(E)
        OK RETURN       ;
    ;SEARCHES DIRECTORY AND REPLACES OLD ITEM
    ;WITH NEW.  IF NONE IS PRESENT AN ENTRY
    ;IS MADE.
```

NON DUMP INPUT, OUTPUT

TWO PROGRAMMED OPERATORS ARE REQUIRED FOR NORMAL I/O OPERATION. THESE ARE:

INPUT D,ADR AND OUTPUT D,ADR

THE FUNCTIONS OF THESE TWO COMMANDS ARE SIMILAR, ONE SERVING FOR INPUT OPERATIONS AND THE OTHER FOR OUTPUT OPERATIONS. USUALLY, THE ADDRESS ADR IS Ø AND THE COMMANDS OPERATE AS FOLLOWS:

INPUT COMMAND
RELEASES THE CURRENT BUFFER TO THE I/O ROUTINES FOR REFILLING AND ADVANCES THE BUFFER HEADER POINTERS TO THE NEXT BUFFER. IF THE NEXT BUFFER IS NOT FULL WHEN THE INPUT COMMAND IS GIVEN, THE I/O ROUTINES DELAY UNTIL THE BUFFER IS FULL OR AN END OF DATA INDICATION IS RECEIVED. SINCE INPUT OPERATIONS NORMALLY CONTINUE UNTIL ALL BUFFERS ARE FULL, THE INPUT COMMAND SERVES TO SYNCHRONIZE THE USER AND THE INPUT DEVICE.

OUTPUT COMMAND
COMPUTES A WORD COUNT FROM THE POSITION OF THE BYTE POINTER, STORES IT IN THE RIGHT HALF OF THE FIRST DATA WORD, AND MAKES THE BUFFER AVAILABLE TO THE DEVICE FOR OUTPUTTING. THE BUFFER HEADER POINTERS ARE ADVANCED TO THE NEXT BUFFER AND THE ITEM COUNT IS SET TO THE MAXIMUM NUMBER THAT CAN FIT IN THE BUFFER. THE SERVICE ROUTINES ASSURE THAT THE BUFFER IS EMPTY BY WAITING UNTIL IT IS OUTPUT, IF NECESSARY, BEFORE RETURNING AND CLEAR THE BUFFER TO ZERO.
WHEN THE INIT COMMAND IS GIVEN, THE ITEM COUNT IN EACH BUFFER HEADER IS SET TO ZERO AND THE BYTE SIZE IS SET ACCORDING TO THE DATA PROCESSING MODE. THE INBUF AND OUTBUF COMMANDS SET THE POINTERS TO THE FIRST BUFFER IN THE RING. THE REMAINING POINTERS ARE SET UP WHEN THE FIRST INPUT OR OUTPUT COMMAND IS GIVEN. IF THE ITEM COUNT IS IGNORED BY THE USER'S PROGRAM DURING OUTPUT OPERATIONS, THE OUTPUT COMMAND MUST BE GIVEN ONCE AFTER THE INIT COMMAND TO SET UP THE OUTPUT BUFFER HEADER. IF THE USER'S OUTPUT SUBROUTINE CHECKS THE ITEM COUNT BEFORE PUTTING DATA INTO THE OUTPUT BUFFER, THIS INITIAL OUTPUT COMMAND HAPPENS AUTOMATICALLY.

A TYPICAL GET ONE CHARACTER SUBROUTINE FROM AN INPUT DEVICE IS :

```
GET:    Ø
        SOSG IBUF+2          ;DECREMENT ITEM COUNT AND TEST
        INPUT D,             ;IF NO DATA, CALL INPUT
        ILDB AC,IBUF+1       ;GET NEXT CHARACTER IN AC
     JRST @GET               ;EXIT
```

A TYPICAL OUTPUT ONE CHARACTER SUBROUTINE WOULD BE:

```
PUT:    Ø
        SOSG OBUF+2          ;DECREMENT ITEM COUNT AND TEST
        OUTPUT D,            ;IF NO ROOM LEFT CALL OUTPUT
        IDPB AC,OBUF+1       ;DEPOSIT CHARACTER FROM AC
```

JRST @PUT

IF ADR IS NON ZERO, IT IS TAKEN AS A POINTER TO THE SECOND WORD OF THE NEXT BUFFER. THE SEQUENCE OF BUFFERS IN THE RING THUS CAN BE ALTERED. THE BUFFER POINTED TO BY ADR CAN BE IN AN ENTIRELY SEPARATE RING FROM THE PREVIOUS BUFFER. ONCE A NEW BUFFER POSITION IS ESTABLISHED, FOLLOWING BUFFERS ARE TAKEN FROM THE RING STARTING AT ADR. WHEN USING THIS FEATURE IN OUTPUT OPERATIONS, THE I/O ROUTINES COMPUTE AN IMPROPER WORD COUNT UNLESS ONE OF THE FOLLOWING PRECAUTIONS ARE TAKEN. EITHER 1) IN THE INIT COMMAND HAVE BIT 31 A ONE TO PREVENT THE SERVICE ROUTINES FROM COMPUTING A WORD COUNT (USEFUL IF THE USER'S PROGRAM COMPUTES ITS OWN WORD COUNT AND PLACES IT IN THE RIGHT HALF OF THE FIRST DATA WORD OF THE BUFFER OR IF OUTPUTTING DIRECTLY FROM AN INPUT BUFFER) OR 2) SET THE BYTE POINTER OF THE BUFFER HEADER TO POINT TO THE LAST ITEM IN THE NEW BUFFER.

STATUS , STATZ, STATO, COMMANDS

CERTAIN ERRORS, SUCH AS PARITY OR CHECKSUM ERRORS, CAN OCCUR DURING DATA PROCESSING. FOR EACH 4-BIT DEVICE NUMBER, A STATUS WORD IS PROVIDED WHICH CONTAINS ERROR INDICATION BITS AND OTHER USEFUL CONDITIONS. THE STATUS WORD IS EXAMINED OR TESTED BY THE COMMANDS:

STATUS D,ADR OR STATZ D, MASK OR STATO D,MASK

THESE THREE COMMANDS ARE EXACTLY ANALOGOUS TO THE COMMANDS CONI, CONSZ, AND CONSO EXCEPT THAT D IS THE 4-BIT DEVICE NUMBER.

STATUS BITS

THE BITS OF THE STATUS WORD HAVE THE FOLLOWING MEANINGS:

BIT	MEANING
18	IOIMPM, IMPROPER MODE, THE SELECTED MODE IS UNOBTAINABLE ON THE SELECTED DEVICE.
19	IODERR, DEVICE ERROR, THE DEVICE'S SELF-CHECKING CIRCUITS INDICATE AN ERROR (PARITY FAILURE(ETC.)
20	IODTER, DATA ERROR, THE COMPUTED CHECKSUM FAILED OR INVALID DATA WAS RECEIVED.
21	IOBKTL, BLOCK TOO LARGE, A BLOCK OF DATA FROM A HIGH-SPEED DEVICE (DECTAPE, MAG TAPE, DRUM) WAS TOO LARGE TO FIT IN ONE BUFFER.
22	IODEND, DATA END ENCOUNTERED.
23	IOACT, DEVICE ACTIVE AND CURRENTLY TRANSMITTING DATA.
24	IOBOT, BEGINNING OF TAPE
25	IOTEND, END OF TAPE
26	IOPAR, WRITE EVEN PARITY

27,28 IODENS, DENSITY OF TAPE

29 IONRCK, SUPPRESS ERROR CHECKING IF =1. NORMAL MODE.Ø. RE-READS RECORD 3 TIMES IN ATTEMPT TO GET CORRECT DATA.

29 IORDEL, READ AND DELETE FILE.

3Ø IOCON, NON-CONTINUOUS OPERATION.

31 IOWC, DO NOT COMPUTE A WORD COUNT ON OUTPUT.

32-35 DATA PROCESSING MODE.

BITS 26 THROUGH 35 READ BACK THE MODE TRANSMITTED TO THE I/O SERVICE ROUTINES BY THE MOST RECENT INIT COMMAND.

DUMP MODE INPUT-OUTPUT

THE DUMP MODE FORMAT IS A CONTIGUOUS GROUP OF ARBITRARY SIZED DATA BLOCKS WITHIN A SINGLE FILE. DUMP MODE IS USED TO STORE A COMPLETE PROGRAM IN BLOCKED CORE AREAS, AND REQUIRES NO INTERMEDIATE DATA BUFFERING. IT IS AVAILABLE FOR HIGH SPEED DEVICES (DECTAPE, MAGTAPE, DRUM, OR DISC) AND IS USED TO STORE THE OPERATING EXECUTIVE SYSTEM, THE CUSPS, AND USER PROGRAMS.

THE CONTROL OF INFORMATION IS THROUGH A COMMAND LIST. THE LOCATION OF THE COMMAND LIST IS GIVEN BY THE INPUT OR OUTPUT INSTRUCTION. COMMANDS EXIST TO TRANSFER A BLOCK, JUMP, OR STOP.

COMMAND LIST FORMAT

THE COMMANDS FORMAT VARIES FOR THE SPECIFIC DEVICE BUT ARE IN GENERAL:

```
        ---------------------

IOWD    A,B         ;PLACE "A" WORDS, STARTING AT "B" ON THE
                    ;FILE.

XWD     0, A        ;GO TO A FOR THE NEXT COMMAND.

        0           ;TERMINATE THE FILE

        ---------------------
```

OUTPUT D, ADR COMMAND

DUMP MODE IS WRITTEN IN A FILE BY GIVING AN OUTPUT D,ADR COMMAND. THE CORE LOCATION "ADR" CONTAINS A DUMP FILE COMMAND LIST, WHICH SPECIFIES THE BLOCKS OF CORE THAT ARE TO BE WRITTEN IN THE FILE.

INPUT D,ADR COMMAND

INPUTS A FILE AS SPECIFIED BY COMMAND LIST AT "ADR".

INPUT D, COMMAND

INPUTS A FILE USING THE FIRST WORD OF THE FILE AS THE COMMAND LIST.

CLOSE COMMAND

WHEN FINISHING INPUT AND OUTPUT OPERATIONS FOR A FILE, THE USER ISSUES THE COMMAND:

```
CLOSE D
```

CLOSE FINISHES OUTPUT OPERATIONS ON DEVICE D BY DUMPING THE LAST BUFFER IF IT IS PARTIALLY FULL AND, PERHAPS, BY WRITING AN END CHARACTER IF THE DEVICE REQUIRES ONE. FOR EITHER INPUT OR OUTPUT, THE DEVICE IS RETURNED TO THE SAME STATE AS BEFORE THE FIRST INPUT OR OUTPUT COMMAND.

RELEASE COMMAND

RELEAS MAY PERFORM A CLOSE OPERATION AND THEN RELEASES THE DEVICE, D, FOR USE BY OTHER USERS UNLESS AN ASSIGN HAD BEEN GIVEN TO REQUEST THE DEVICE. A NEW INIT COMMAND MUST BE GIVEN IF FURTHER OPERATIONS WITH THE DEVICE ARE REQUIRED. THE COMMAND IS:

```
RELEASE D,
```

IF A DEVICE IS INITIALIZED ON MORE THAN ONE CHANNEL, ALL SUCH CHANNELS, BOTH INPUT AND OUTPUT ARE AFFECTED BY RELEAS COMMANDS.

DEASSIGN COMMAND

THIS COMMAND RETURNS A DEVICE BACK TO THE SYSTEM. THE DEASSIGNMENT IS ON A NAMED BASIS, AND THE SYSTEM FIRST SEARCHES TO FIND A CORRECT LOGICAL NAME FOR A DEVICE TO DEASSIGN, FOLLOWED BY A SEARCH FOR A PHYSICAL DEVICE. THE COMMAND IS:

```
DEASSIGN,   ADR
ADR- Ø  SIXBIT /LOGICAL OR PHYSICAL NAME/
```

EXIT COMMAND

TO RELEASE ALL DEVICES AND STOP EXECUTION, ISSUE THE COMMAND:

```
CALL [SIXBIT/EXIT/]
```

CONSOLE OR USER TELETYPEWRITTERS

EQUIPMENT

A MODEL 33 OR 35 MAY BE USED, WITH EITHER HALF OR FULL DUPLEX COMMUNICATION WITH THE 630 DATA COMMUNICATIONS SYSTEM AND THE 626 CONSOLE TELEPRINTER (FULL DUPLEX). HORIZONTAL TAB, VERTICAL TAB, AND FORM FEED ARE SYNTHETICALLY GENERATED USING SPACES AND LINE FEEDS WHEN A MODEL 33 TELETYPE IS USED. THE SETTING FOR TABULATION IF A MODEL 35 IS USED ARE: HT = 8 SPACES, VT = 4 LINE FEEDS, AND FF = 8 LINE FEEDS (1 PAGE).

INPUT CONTROL CHARACTERS

CR SYSTEM RESPONDS BY TYPING A LINE FEED.

RUB-OUT - DELETES PREVIOUS CHARACTER TYPED, SYSTEM RESPONDS BY TYPING\.

ALT MODE,

LINE FEED (TYPED BY USER OR SYSTEM). VERTICAL TAB, OR FORM FEED. DELIMIT A LINE, COMPLETE AN INPUT COMMAND, OR TRANSFERS CONTROL TO THE SYSTEM COMMAND DECODER.
CONTROL Z =S2 - DATA END.

PAPER TAPE READER

A, AL MODE CONTROL

IGNORE RUBOUT, BLANK TAPE, NULL (200).

B MODE CONTROL

BLOCKED DATA WITH BLOCK WORD COUNT AND CHECKSUM IN THE FIRST WORD OF THE BLOCK. (SEE PAPER TAPE PUNCH)

MANUAL OPERATION

TURNING THE READER SWITCH OFF GIVES THIS DATA END CONDITION. AN S2 CHARACTER (32) IS ALWAYS GENERATED AS THE LAST CHARACTER.

PAPER TAPE PUNCH

SPECIAL CLOSE

INITIAL AND FINAL TAPE FEED OCCURS AT FIRST OUTPUT, AND AT CLOSE.

A,AL MODE CONTROL

200 IS ADDED TO ALL CHARACTERS). TAPE FEED IS INSERTED AFTER FORM FEED. A RUBOUT IS INSERTED AFTER A TAB.

B MODE CONTROL

PUNCHES EACH BUFFER AS A BINARY BLOCK OF DATA. THE FIRST WORD IN EACH BLOCK IS THE WORD COUNT IN THE RIGHT HALF AND THE CHECKSUM IN THE LEFT HALF, FOLLOWED BY THE DATA. TAPE FEED IS INSERTED AFTER EACH BLOCK.

LINE PRINTER

SPECIAL CLOSE

FORM FEED OCCURS PRIOR TO ACTUAL FIRST OUTPUT, AND AFTER CLOSE.

SPECIAL CONTROL CHARACTERS
SPECIAL CODES HAVE MEANING TO THE LINE PRINTER. THESE ARE:

HORIZONTAL TAB (Ø11) = 1Ø SPACES
FORM FEED (Ø14) = 1 PAGE
EOT (ØØ4) = STOPS PRINTER
DC0 (Ø2Ø) = 1 LINE
DC1 (Ø21) = 2 LINES
DC2 (Ø22) = 3 LINES
DC3 (Ø23) = 6 LINES
DC4 (Ø24) = 11 LINES
VERTICAL TAB (Ø13) = 22 LINES

CARD READER

GENERAL
EACH BUFFER CONTAINS THE CONTENTS OF ONE CARD FOR ALL MODES.

THE BINARY CARD FORMAT WILL CONTAIN A 7-9 PUNCH IN COLUMN ONE. ROWS 2 THROUGH 6 OF COLUMN 1 WILL CONTAIN THE NUMBER OF DATA WORDS ON THE CARD. ROWS 11, 12, Ø, 1 MAY BE USED FOR MODE AND TYPE INFORMATION. ZERO IN THOSE ROWS WILL MEAN A NORMAL BINARY CARD. COLUMN 2 WILL CONTAIN A FOLDED 12 BIT CHECKSUM. COLUMN 3 THROUGH 8Ø MAY CONTAIN DATA WORDS. FOR DENSELY PACKED BINARY CARDS, THIS PERMITS 26 WORDS PER CARD. FOR SEQUENCED NUMBERED CARDS, THIS PERMITS 24, IF 6 COLUMNS ARE USED FOR SEQUENCING INFORMATION. THE CHECKSUM WILL INCLUDE ONLY THE DATA WORDS.

A, AL MODE OPERATION
BLANKS ARE REPLACED BY SPACES. A CARRIAGE RETURN AND LINE FEED IS APPENDED TO EACH CARD. BUFFER CONSISTS OF 8Ø CHARACTERS, CR, LF. ALL CHARACTERS ARE TRANSFORMED TO ASCII. (SEE THE MACRO-6 MANUAL FOR A TABLE GIVING THE TRANSLATION BETWEEN HOLLERITH AND ASCII)

B MODE
BINARY READS FIRST TWO COLUMNS AS CONTROL INFORMATION. COLUMN 1 MUST CONTAIN A 7-9 PUNCH. COLUMN 2 IS A 12-BIT CHECKSUM AS DESCRIBED FOR THE PAPER TAPE PUNCH.

MANUAL CONTROL
EOF SWITCH CAUSES IODEND TO BE SET.

DECTAPE FORMAT

1103 ADDRESSABLE BLOCKS OF 200 WORDS EACH WITH BLOCK 0-A LOADER, BLOCK 1-THE DIRECTORY, AND BLOCKS 2-1101-DATA.

BLOCK 1: DIRECTORY FORMAT

WORD 0, C(LH):	LAST BLOCK OF TAPE USED (2 INITIALLY)
WORD 0, C(RH):	LOCATION ON BLOCK 1 OF THE FIRST DIRECTORY ENTRY. (5 INITIALLY)
DIRECTORY ENTRIES:	4 WORDS PER ENTRY
WORD 0:	6 SIXBIT NAME OF FILE
WORD 1, C(LH)- 0	3 SIXBIT FILE EXTENSION
WORD 1, C(RH):	FIRST BLOCK OF FILE
WORD 2, BITS 0-2:	ACTIVITY MODES (UNUSED)
WORD 2, BITS 3-6:	DATA MODE (A,AS,B,ETC.)
WORD 2, BITS 8-35:	DATE
WORD 3, C(LH):	DUMP FILE WORD COUNT
WORD 3, C(RH):	DUMP FILE ADDRESS

BLOCKS 2-1101:
NON DUMP FILE DATA:

EACH BLOCK CONTAINS	UP TO 177 DATA WORDS.
WORD 0, C(LH):	LINK TO NEXT BLOCK OF FILE.
WORD 0, C(LH)=0:	LAST BLOCK OF FILE
WORD 0, C(RH):	NUMBER OF ITEMS IN BLOCK
WORDS 1-177:	DATA

DUMP FILE DATA - NON CONTROL WORDS IN THE DATA:
WORD COUNT WORDS ARE PLACED IN CONTIGUOUS BLOCKS, AND DATA ARE STORED ACROSS PHYSICAL BLOCK BOUNDARIES. THE FINAL BLOCK MAY NOT BE FILLED.

DUMP FILE DATA - CONTROL WORDS IN THE DATA:

WORD 0:	COMMAND WORD. NEXT COMMAND WORD IS TAKEN FROM DECTAPE FILE

LOOKUP COMMAND

SEARCHES DIRECTORY FOR FILENAME AND FILENAME EXT (OR BLANK), AND FILLS DIRECTORY INFO INTO C(E,E+1,E+2,E+3). THE BLOCK NUMBER C(E+1 RIGHT) IS PLACED IN THE INPUT BLOCK NUMBER. CLEARS ALL ERROR BITS. AN EXAMPLE IS:

```
LOOKUP D,E          ;E=LOCATION OF 4 WORD
JRST ERROR          ;DIRECTORY BLOCK. ERROR
OK RETURN           ;= NOT FOUND
```

ENTER COMMAND

SEARCHES DIRECTORY AND REPLACES OLD ITEM WITH NEW. A NEW ENTRY IS INSERTED IF NONE PRESENT. A FREE BLOCK IS PLACED IN C(E+1 RIGHT), AND OUTPUT BLOCK NUMBER CLEARS ALL ERROR BITS. AN EXAMPLE IS:

```
ENTER D,E           ;E=LOCATION OF 4 WORD
JRST ERROR          ;DIRECTORY BLOCK. ERROR=
OK RETURN           ;NO DIRECTORY SPACE.
```

MISCELLANEOUS COMMANDS

```
    USETI D,E       ;SETS DECTAPE INPUT BLOCK NUMBER TO E.

    USETO D,E       ;SETS DECTAPE OUTPUT BLOCK NUMBER TO E

    UGETF D,E       ;GET THE NEXT FREE BLOCK NUMBER, SET THE OUTPUT
                    ;BLOCK NUMBER (USETO), SET BLOCK
                    ;IN C(E RIGHT), AND INCREMENT THE NEXT FREE BLOCK.

    CLOSE D,X       ;X=1(CLOSE OUTPUT),X=2(CLOSE INPUT),X=3(DO NOTHING)
                    ;CLOSE PRESENT FILE OF DECTAPE
                    ;WRITE FINAL BUFFER IF IT WERE LINKED TO A FREE
                    ;BLOCK TERMINATE FILE.

    RELEAS D,X      ;RELEASES DEVICE, THE DIRECTORY
                    ;IS REWRITTEN IF NECESSARY
```

NON DUMP OUTPUT COMMANDS

THE BUFFER LINK VALUE DETERMINES BLOCK ADDRESSES FOR OUTPUT AS FOLLOWS:

VALUE	ACTION
Ø	REPLACE LINK WITH NUMBER OF NEXT FREE BLOCK ON TAPE, WRITE AT CURRENT OUTPUT BLOCK NUMBER AND SET NEW CURRENT OUTPUT BLOCK NUMBER TO NEXT FREE BLOCK (RE-LINK).
1	LAST BLOCK OF FILE. SET LINK TO Ø AND WRITE
N	ACTUAL LINK REQUESTED BY USER. SET NEXT OUTPUT BLOCK TO N. (N=2-1100.)

USETO MAY MODIFY BLOCK NUMBER.

NON DUMP INPUT COMMANDS

SUCCESSIVE BLOCKS OF THE TAPE ARE READ INTO BUFFER(S), USING LINK TO LOCATE NEXT BLOCK.

USETI MAY MODIFY NEXT BLOCK NUMBER.

DUMP MODE INPUT OUTPUT COMMANDS

DUMP MODE IS CODE 17. THE COMMAND LIST FOR INPUT MAY BE EITHER ON CORE OR ON TAPE. THE COMMANDS ARE SHOWN BY EXAMPLE:

```
        INPUT D,E       ;INPUT AS DUMP FILE ACCORDING TO COMMAND LIST
                        AT E.

        INPUT D,        ;INPUT ON DEVICE, TAKE COMMANDS FROM THE FILE.
                        ;FIRST COMMAND IS FIRST WORD OF FILE.

        OUTPUT D,E      ;OUTPUT DUMP FILE ACCORDING TO COMMAND
                        ;LIST AT E.
```

MAGNETIC TAPE DATA FORMAT

```
REFLECTIVE STRIP (BEGINNING OF TAPE, BOT)
DIRECTORY ENTRY RECORD (4 WORDS)
FIRST RECORD, FIRST FILE
          .         .
          .         .
LAST RECORD, FIRST FILE
END OF FILE
          .         .
          .         .
LAST RECORD, FIRST FILE
          .         .
          .         .
LAST RECORD, LAST FILE
END OF FILE
END OF FILE (SIGNIFIES LOGICAL END OF TAPE, EOT)
```

DIRECTORY FORMAT

```
WORD 0:        FILE NAME (6-SIXBIT CHARS)
WORD 1:        C(LH)=FILE EXTENSION (3-SIXBIT)
WORD 2:        BIT 3-6=MODE
WORD 3:        C(LH)=WORD COUNT FOR DUMP FILE, C(RH)= DUMP ADDRESS
```

PHYSICAL RECORDS FORMAT

ONE PHYSICAL RECORD/DUMP FILE COMMAND IN DUMP FILES
FIRST WORD OF RECORD (128 WORDS) CONTAINS NUMBER OF ITEMS IN RECORD.

ENTER COMMAND - ENTER D, E

SET TAPE FOR OPERATION ON A PARTICULAR FILE

```
ENTER D,E
```

ENTER CAUSES THE TAPE TO BE BACKSPACED ONE FILE OR TO THE BEGINNING OF TAPE (BOT). THE FOUR WORD FILE HEADER STARTING AT E IS WRITTEN ON TAPE AS A RECORD AND THE TAPE IS POSITIONED FOR WRITING DATA.

LOOKUP COMMAND - LOOKUP D,E

LOOKUP CAUSES A BACKSPACE TO THE BEGINNING OF THE NEAREST FILE AND A FORWARD SEARCH FROM THAT POINT FOR THE FILE NAME CONTAINED IN THE FILE HEADER AT E. THE ERROR RETURN IS TAKEN IF A DOUBLE END OF FILE IS REACHED. IF THE SEARCH IS SUCCESSFUL, THE TAPE IS POSITIONED FOR READING OR WRITING THE DATA IN THE FILE.

AN EXAMPLE OF LOOKUP IS:

```
LOOKUP D,E          ;E=HEADER TO LOOKUP
JRST ERROR          ;NOT FOUND ON TAPE
FOUND EXIT
```

DATA TRANSMISSION COMMANDS - DUMP MODE

DUMP MODE =(16). DUMP DOES NOT RESTART USER UNTIL DATA TRANSFER IS COMPLETE.

INPUT D,E — INPUT ACCORDING TO THE COMMAND LIST AT E.

OUTPUT D,E — OUTPUT ACCORDING TO THE COMMAND LIST AT E.

MTAPE MOVEMENT COMMAND - MTAPE D,FCN

FCN VALUE	TAPE FUNCTION
1	REWIND
11	REWIND AND UNLOAD
7	BACKSPACE RECORD
17	BACKSPACE FILE
3	WRITE END OF FILE
6	SKIP RECORD
13	WRITE BLANK TAPE (3")
16	SKIP FILE
10	SPACE TO LOGICAL END OF TAPE

CLOSE COMMAND - CLOSE,D

INPUT — (NONDUMP) SET TO NEXT BUFFER

INPUT — (DUMP) SPACE TO EOF IF DUMP

OUTPUT — DUMP CURRENT BUFFER, WRITE TWO EOFS, BACKSPACE ONE EOF.

GET DUMP WORD COUNT COMMAND - GETN D,E

PLACE WORD COUNT FOR FILE IN C(E).

COMMAND LIST OPERATION

THE DISPLAY HAS A COMPLETE INSTRUCTION SET FOR DISPLAYING PICTURES AND ITS COMMANDS ARE EXECUTED INTERRUPTIVELY BY THE DISPLAY SERVICE ROUTINE. THE COMMANDS PROVIDE FOR DISPLAYING DATA TRANSMISSION OF CONTROL WORD (OR INDEX VALUES), INSTRUCTIONS TO ALLOW SUBROUTINES TO BE WRITTEN IN THE COMMAND LANGUAGE, LIGHT PEN TRACKING, ETC. AND TO DISPLAY A COMPLETE PICTURE.

A COMMAND TO DISPLAY A LIGHT PEN POSITIONAL TRACKING CROSS. EACH TIME A LIGHT PEN TRACKING INSTRUCTION IS GIVEN, THE CO-ORDINATES OF THE LIGHT PEN ARE UPDATED.

THE DISPLAY IS STARTED BY ISSUING AN OUTPUT D,ADR INSTRUCTION WITH THE ADDRESS, ADR, OF A DATA BLOCK OF DISPLAY STATUS REGISTERS. THESE DISPLAY REGISTERS COMPLETELY HOLD THE STATE OF THE USER'S DISPLAY. THE DISPLAY AND USER'S PROGRAMS (WHICH INITIATED THE USER'S DISPLAY) OPERATED CONCURRENTLY.

THE DISPLAY REGISTERS

DISLCT	DISPLAY LOCATION COUNTER WHICH POINTS TO USER'S DISPLAY INSTRUCTION CURRENTLY BEING EXECUTED.
DISBLK	CURRENT IO POINTER WORD TO BLOCK OF DATA BEING DISPLAYED.
LPPOS	LIGHT PEN TRACKING CROSS POSITION, BIT Ø=1 IF SEEN.
LPDS	LIGHT PEN DATA STATUS WORD.
DISPD	PUSH DOWN POINTER FOR LOCATION COUNTER.

DISPLAY COMMANDS

```
IOWD N, A       ;DISPLAY N DATA WORDS BEGINNING AT C(A).  LOAD

XWD C, B        ;DISBLK WITH C(A).
                ;THE DATA FORMAT IS THE 346 DISPLAY WORD DATA.
                ;PLACE THE FIRST FOUR WORDS OF THE DISPLAY REGISTERS
                ;IN C(B,B+1,B+2,B+3).  JUMP TO C IF LIGHT PEN IS SEEN.

XWD 0, A        ;JUMP TO ADDRESS A.

Ø               ;STOP

XWD 2, 0        ;UPDATE LIGHT PEN TRACKING POSITION CROSS.
                ;LPPOS BITØ IS RESET TOO.

XWD 4, A        ;TRANSFER A SOURCE WORD IN MEMORY TO A DESTINATION.
                ;C(A LEFT) = SOURCE MEMORY WORD LOCATION.
                ;C(A RIGHT) = DESTINATION MEMORY WORD LOCATION.

XWD 5, A        ;TRANSFER RIGHT HALF OF WORD ONLY.

XWD 6, A        ;SUBTRACT 1 FROM C(A).  SKIP NEXT INSTRUCTION
                ;IF C(A) IS LESS THAN OR EQUAL TO Ø.

XWD 1Ø, A       ;SKIP IF C(A) IS GREATER THAN OR EQUAL TO Ø.
```

```
     XWD 11, A       ;ADD ONE TO C(A).

     XWD 12, A       ;PUSH DOWN JUMP TO A.  C(DISPD)+1←C(DISPD).
                     ;C(DISLCT)←C(DISPD). C(DISLCT)←A.

     XWD 13, 0       ;RETURN JUMP FROM PUSHDOWN C(DISLCT)←C(DISPD).
                     ;C(DISPD)←C(DISPD)-1.

                     --------------------

LPPOS - LIGHT PEN TRACKING STATUS WORD
     BIT Ø           IF 1, LIGHT PEN WAS SEEN.

     BITS 8-13       X-POSITION

     BITS 24-35      Y-POSITION

LPDS - LIGHT PEN DATA STATUS WORD
     BIT Ø           IF 1, LIGHT PEN WAS SEEN

     BITS 8-17       X POSITION OF PEN.
     BITS 24-35      Y POSITION OF PEN

CLOSE COMMAND
     STOPS THE DISPLAY
```

C H A P T E R V

M I S C E L L A N E O U S M O N I T O R F U N C T I O N S

THESE FUNCTIONS ARE AVAILABLE AS EITHER "CALL UUOS" OR ARE GENERALLY PERTINENT TO OPERATION OF THE SYSTEM.

"CALL" PROGRAM OPERATORS OR UUOS.

THESE FUNCTIONS HAVE THE FORM :

```
        ---------------------

        CALL    AC, [SIXBIT /FUNCT/]

        ---------------------
```

WHERE "FUNCT" IS THE NAME OF THE MONITOR ROUTINE WHICH PERFORMS THE ACTION, THE AC, OR REGISTERS FOLLOWING THE "CALL" MAY CARRY ARGUMENTS, ETC.

CALL "DATE"

```
        ---------------------

        CALL    AC, [SIXBIT.DATE.]

        ---------------------
```

RETURNS WITH THE DATE IN ACCUMULATOR AC, IN THE FORM:

```
        (((Y-1964)   12+(M-1))31+(D-1)
```

WITH A 12 BIT NUMBER.

C H A P T E R V I

T H E L O A D E R

GENERAL

1. INITIALLY RESIDENT IN USER'S AREA. CONTROLLED BY A USER STATION.

2. LOADS AND LINKS RELOCATABLE BINARY FORMAT PROGRAMS ASSEMBLED BY MACRO6 OR FOLA (1 PASS FORTRAN ASSEMBLER), PROVIDING SPACE FOR FORTRAN COMMON. SEARCHES LIBRARY TAPES, AND LOADS OTHER FILES, INDEPENDENT OF IO INPUT MEDIA.

3. PROVIDES LOADED PROGRAM TO RUN, SYMBOL TABLE IN CORE FOR DDT, AND CREATES STORAGE MAP OF PROGRAMS FOR USER.

4. SIZE: 2K (OCTAL) WITH IO BUFFERS. (STORAGE IS RECOVERABLE JUST BEFORE RUNNING)

5. EQUIPMENT: USER TELETYPE, IO DEVICE FOR FILES FROM WHICH TO LOAD, IO DEVICE FOR LOADER MAP.

COMMANDS TO LOADER FROM TELETYPE

CONTROL MODE SWITCHES (ENCLOSED IN PARENTHESES)

S-	LOAD LOCAL SYMBOLS
W	DO NOT LOAD LOCAL SYMBOLS
L-	SEARCH FILES IN LIBRARY SEARCH MODE
N-	STOP LIBRARY SEARCH MODE
A-	IGNORE ALL SUBSEQUENT STARTING ADDRESSES
M-	PRINT STORAGE MAP
R-	RESTART LOADING
I-	SET LOADER TO CURRENT STATE. THIS STATE WILL BE RETURNED TO IF R IS TYPED.
	(OTHER CONTROL CHARACTERS)
'-	ENTER LIBRARY SEARCH FOR THE PREVIOUS FILE
ALT MOD	GO INTO EXECUTION

COMMAND FORMAT

```
STORAGE MAP FILE!(SWITCHES) SOURCE FILE 1,...(SWITCHES) SOURCE FILE N
```

EXAMPLE

```
TTY:←DTA4:(S)PG1,PG2,(WL)LB1,ARTLIB,(N)TSTDAT,(M)
```

LOADS FROM DTA4 FILES

PG1,PG2 WITH LOCALS
LB1,ARTLIB WITHOUT LOCALS AS LIBRARY FILES
TSTDAT IS LOADED WITHOUT LOCAL SYMBOLS
DESTINATION FILE IS STORAGE MAP.
(NO DESTINATION FILE IMPLIES TTY)

ERROR MESSAGES

ILLEGAL FORMAT
INDICATES THAT THE BINARY INPUT TO THE LOADER IS NOT IN PROPER FORM, AND THEREFORE CANNOT BE LOADED.

DATA ERROR
INDICATES THAT DATA CANNOT BE READ BECAUSE OF ILLEGAL CHECKSUM, BAD PARITY, ETC.

CORE EXCEEDED
THIS MESSAGE IS TYPED WHEN THE HIGHEST NUMBERED LOCATION LOADED IS GREATER THAN OR EQUAL TO THE LOWEST NUMBERED LOCATION IN THE LOADER SYMBOL TABLE. NO FURTHER LOADING IS POSSIBLE. PROGRAM EXECUTION MAY BE POSSIBLE IF THE AREA OVERLOAPPING THE SYMBOL TABLE DOES NOT CONTAIN CONSTANTS OR INSTRUCTIONS AND DDT IS NOT USED.

ILLEGAL COMMON
THIS MESSAGE IS TYPED IF A PROGRAM TRIES TO INCREASE THE LENGTH OF COMMON. THIS ERROR CAN BE AVOIDED BY REORDERING THE BINARY INPUT TO THE LOADER SO THAT THE PROGRAM WITH THE LONGEST COMMON AREA IS LOADED FIRST.

COMMAND ERROR
THIS MESSAGE IS TYPED IF A FILE OR DEVICE NAME CONTAINS TOO MANY CHARACTERS, OR IF AN INPUT FILE MODE OTHER THAN BINARY IS SPECIFIED.

FILE 'X' NOT FOUND
THIS MESSAGE INDICATES THAT FILE 'X' IS NOT IN THE DECTAPE DIRECTORY.

ALL ERRORS WITH THE EXCEPTION OF FILE 'X' NOT FOUND, CAUSE THE REST OF THE INPUT COMMAND STRING TO BE IGNORED.

DEFINITIONS

1. GLOBAL SYMBOL - ANY SYMBOL ACCESSIBLE TO OTHER PROGRAMS.

2. GLOBAL REQUEST - REQUEST TO THE LOADER TO LINK A GLOBAL TO A PROGRAM. REQUEST MAY BE ADDITIVE OR CHAINED.
   A. CHAINED REQUEST:
   A REQUEST WHICH POINTS TO THE LAST REFERENCE IN THE PROGRAM TO THE GLOBAL SYMBOL REQUESTED. EACH REFERENCE IN THE PROGRAM POINTS TO THE PREVIOUS REFERENCE TO THE REQUESTED GLOBAL. SUCH A CHAIN IS TERMINATED BY A NON-RELOCATABLE ZERO ADDRESS IN THE PROGRAM. CHAINED GLOBALS ARE RESTRICTED

TO REFERENCES APPEARING IN THE ADDRESS PART OF A STORAGE WORD. SYMBOLIC REFERENCES TO THE AC OR INDEX FIELDS CANNOT BE CHAINED.

B. ADDITIVE REQUEST:
A REQUEST WHICH FOR SOME REASON CANNOT BE CHAINED. THE REQUEST POINTS TO A STORAGE WORD WHICH IS TO HAVE THE REQUESTED GLOBAL ADDED IN THE WORD AFTER APPROPRIATE SHIFTING AND MASKING. EACH REFERENCE TO A GLOBAL THAT CONNOT BE CHAINED REQUIRES A SEPARATE ADDITIVE REQUEST. THE ADDITIVE REQUEST CARRIES SUFFICIENT INFORMATION SO THAT A GLOBAL VALUE CAN BE PLACED IN THE AC OR INDEX FIELD OF AN INSTRUCTION.

3. PROGRAM ORIGIN - THE LOCATION ASSIGNED BY THE LOADER TO RELOCATABLE ZERO OF A PROGRAM.

4. THE PROGRAM BREAK - THE LENGTH OF A PROGRAM, THE FIRST LOCATION NOT USED BY A PROGRAM (BEFORE RELOCATION) THE RELOCATION CONSTANT FOR THE FOLLOWING PROGRAM (AFTER RELOCATION).

5. OFFSET - THE NUMBER OF LOCATIONS TOWARD ZERO, A PROGRAM MUST BE MOVED BEFORE IT CAN BE EXECUTED.

6. RELOCATION CONSTANT - THE NUMBER ADDED TO EVERY RELOCATABLE REFERENCE WITHIN A PROGRAM. THE RELOCATION CONSTANT IS THE RELOCATED PROGRAM BREAK OF THE PREVIOUS PROGRAM.

7. INTERNAL SYMBOL - GENERATES A GLOBAL DEFINITION WHICH IS USED TO SATISFY ALL GLOBAL REQUESTS FOR THAT SYMBOL.

8. EXTERNAL SYMBOL - GENERATES A CHAINED GLOBAL REQUEST AND ONE OR MORE ADDITIVE GLOBAL REQUESTS.

9. COMMON - AREA SET ASIDE FOR COMMON STORAGE BEFORE THE BEGINNING OF A PROGRAM.

10. THE LIBRARY SEARCH FEATURE - THE LOADER WILL, ON REQUEST, SEARCH A FILE AS A LIBRARY FILE. THIS MEANS THAT A PROGRAM IS LOADED ONLY IF ONE OF ITS ENTRIES MATCHES AN UNSATISFIED GLOBAL SYMBOL REQUEST. AN ENTRY IS SPECIFIED IN THE ASSEMBLER BY USE OF THE ENTRY PSEUDO-OP. ALL FORTRAN FUNCTION AND SUBROUTINE SUB-PROGRAMS ARE COMPILED WITH THE NAME OF THE PROGRAM AS AN ENTRY. THUS A COLLECTION OF PROGRAMS INCLUDING FORTRAN COMPILED PROGRAMS CAN BE LOADED SELECTIVELY FROM A TAPE.

THE SYSTEM LIBRARY IS CURRENTLY SEARCHED ONLY ON REQUEST. THIS FILE CAN BE SEARCHED BY REQUESTING THE FILE DECLIB FROM THE TAPE CONTAINING THE SYSTEM LIBRARY (USUALLY DTA1).

LINKING LOADER INPUT FORMAT

ALL DATA TO THE LOADER IS IN BLOCKS. THE FIRST WORD OF EACH BLOCK CONTAINS AN OCTAL CODE FOR THE BLOCK TYPE IN THE LEFT HALF AND THE WORD COUNT (POSITIVE) IN THE RIGHT HALF. FOLLOWING THE HEADER WORD ARE N SUB-BLOCKS, EACH OF WHICH CONTAINS A SUB-HEADER PLUS 18 OR FEWER DATA WORDS. THESE SUB-HEADERS ARE NOT INCLUDED IN THE WORD COUNT IN THE BLOCK HEADER. THEY CONTAIN 18 TWO-BIT BYTES WHICH ARE THE RELOCATION BITS FOR DATA WORDS WITHIN THAT SUB-BLOCK. THE RELOCATION CODES ARE:

```
          Ø - DO NOT RELOCATE
          1 - RELOCATE RIGHT HALF OF THE STORAGE WORD
          2 - RELOCATE LEFT HALF OF THE STORAGE WORD
          3 - RELOCATE BOTH HALVES

THE BLOCK FORMAT IS CONSTANT FOR INPUT FROM ALL DEVICES, IT IS THE SAME
FOR PAPER TAPE, BINARY CARDS, DECTAPE, DRUMS, ETC., AND IS INDEPENDENT
OF PHYSICAL DIVISIONS IN THE INPUT MEDIA.

BLOCK TYPE 1:       RELOCATABLE OR ABSOLUTE PROGRAMS AND DATA.
          WORD 1    THE LOCATION OF THE FIRST DATA WORD IN THE BLOCK.
          WORD 2              A CONTIGUOUS BLOCK OF PROGRAM OR DATA
                              WORDS.
            .
            .
            .
          WORD N              (N MUST BE LESS THAN 200,000 OCTAL)

BLOCK TYPE 2:       SYMBOLS
          CONSISTS OF WORD PAIRS
          1ST WORD  BITS Ø-3  CODE BITS
          1ST WORD  BITS 4-35 RADIX 50 REPRESENTATION OF SYMBOL
          2ND WORD  DATA (VALUE OR POINTER)

          CODE Ø4:  GLOBAL (INTERNAL) DEFINITION
          2ND WORD  BITS Ø-35 VALUE OF SYMBOL

          CODE 1Ø:  LOCAL DEFINITION
          2ND WORD  BITS Ø-35 VALUE OF SYMBOL

          CODE 6Ø:  CHAINED GLOBAL REQUESTS:
          2ND WORD  BITS Ø-17 = Ø
          2ND WORD  BITS 18-35 POINTER TO FIRST WORD OF CHAIN
                    REQUIRING DEFINITION

          CODE 6Ø:  GLOBAL SYMBOL ADDITIVE REQUEST:
          2ND WORD  BIT Ø = 1.
          BIT 1     SUBTRACT VALUE BEFORE ADDITION
          BIT 2     SWAP HALVES BEFORE ADDITION
          BIT 3     ROTATE LEFT 5 BEFORE ADDITION
          BIT 9     REPLACE LH WITH RESULT IN STORAGE
          BIT 1Ø    REPLACE RH WITH RESULT IN STORAGE
          BIT 11    REPLACE INDEX FIELD WITH RESULT IN STORAGE
          BIT 12    REPLACE AC FIELD WITH RESULT IN STORAGE
          BITS 18-35          POINTER TO WORD REQUIRING ADDITION

BLOCK TYPE 4:       ENTRY BLOCK
          THIS BLOCK CONTAINS A LIST OF RADIX 50 SYMBOLS, EACH OF
          WHICH MAY CONTAIN A ZERO OR ONE IN THE HIGH ORDER CODE BIT.
          EACH REPRESENTS A SERIES OF LOGICAL 'AND' CONDITIONS.  IF ALL
          THE GLOBALS IN ANY SERIES ARE REQUESTED, THE FOLLOWING PROGRAM
          IS LOADED.  OTHERWISE ALL INPUT IS IGNORED UNTIL THE NEXT END
          BLOCK.  THIS BLOCK MUST BE THE FIRST BLOCK IN A PROGRAM.

BLOCK TYPE 5:       END BLOCK
          THIS IS THE LAST BLOCK IN A PROGRAM.  IT CONTAINS ONE WORD
          WHICH IS THE PROGRAM BREAK, THAT IS, THE LOCATION OF THE
          FIRST FREE REGISTER ABOVE THE PROGRAM.  (NOTE:  THIS WORD IS
          RELOCATABLE).  IT IS THE RELOCATION CONSTANT FOR THE FOLLOWING
```

PROGRAM LOADED.

BLOCK TYPE 6: NAME BLOCK
THE FIRST WORD OF THIS BLOCK IS THE PROGRAM NAME (RADIX 50). IT MUST APPEAR BEFORE ANY TYPE 2 BLOCKS. THE SECOND WORD IF IT APPEARS DEFINES THE LENGTH OF COMMON.

BLOCK TYPE 7: STARTING ADDRESS
THE FIRST WORD OF THIS BLOCK IS THE STARTING ADDRESS OF THE PROGRAM. THE LAST BLOCK OF THIS TYPE ENCOUNTERED BY THE LOADER IS USED UNLESS THE CONTROL CHARACTER (A) HAS BEEN TYPED. THE STARTING ADDRESS FOR A RELOCATABLE PROGRAM MAY BE RELOCATED BY MEANS OF THE RELOCATION BITS.

BLOCK TYPE 10: INTERNAL REQUEST
EACH DATA WORD IS ONE REQUEST. THE LEFT HALF IS THE POINTER TO THE PROGRAM. THE RIGHT HALF IS THE VALUE. EITHER QUANTITY MAY BE RELOCATABLE.

LOADER MAP OUTPUT FORMAT

IF REQUESTED, THE LOADER WILL OUTPUT A LIST OF ALL PROGRAMS LOADED. THIS WILL BE A LISTING OF ALL DEFINED GLOBALS AND THEIR VALUES, ALL PROGRAM NAMES WITH THEIR ORIGINS AND LENGTHS, AND AT THE END. A LIST OF ALL UNDEFINED GLOBALS, THE FORMAT IS:

```
PROG NAME ORIGIN    LENGTH

GLOBAL    VALUE
  .         .
  .         .
  .         .
GLOBAL    VALUE

PROG NAME ORIGIN    LENGTH

GLOBAL    VALUE
  .         .
  .         .
  .         .
UNDEFINED GLOBALS LIST
```

USER AREA CORE STORAGE ALLOCATION FORMAT

RELOCATED PROGRAMS ARE LOADED INTO MEMORY FROM THE PROGRAM ORIGIN UPWARD. THE PROGRAM ORIGIN IS THE LOWEST ADDRESS AVAILABLE FOR LOADING PROGRAMS. IT IS NORMALLY 140 PLUS THE LENGTH OF COMMON, UNLESS OTHERWISE ASSIGNED.

THE SYMBOL TABLE IS LOADED FROM THE HIGHEST AVAILABLE LOCATION IN MEMORY DOWNWARD TOWARD THE PROGRAM AREA, WHICH IS BEING LOADED UPWARD. EACH SYMBOL REQUIRES TWO MEMORY LOCATIONS. THE LOWER NUMBERED LOCATION CONTAINS A SYMBOL IN RADIX 50, WITH ITS CLASSIFICATION CODE BITS, AND THE HIGHER LOCATION CONTAINS THE VALUE FOR THE SYMBOL.

CORE SYMBOL TABLE FORMAT

THE LOADER CREATES A SYMBOL TABLE FOR ITS USE WHILE LOADING, AND FOR THE USE OF DDT WHILE DEBUGGING.
THE SYMBOL TABLE IS ORGANIZED INTO SEGMENTS HEADED BY A WORD PAIR CONSISTING OF A PROGRAM NAME AND ITS ORIGIN. FOLLOWING THE PROGRAM NAME

ARE WORD PAIRS CONTAINING A RADIX 50 SYMBOL PLUS FOUR CODE BITS IN ONE WORD AND A VALUE IN THE OTHER WORD.

LOCAL SYMBOLS ARE NOT PLACED INTO THE SYMBOL TABLE BY THE LINKING LOADER UNLESS A REQUEST IS MADE BY THE USER. LOCAL SYMBOLS ARE USED BY DDT. GLOBAL SYMBOLS ARE AUTOMATICALLY PLACED IN THE SYMBOL TABLE BY THE LOADER SO THAT INTER-PROGRAM LINKING CAN TAKE PLACE.

LOCAL SYMBOLS, GLOBAL SYMBOLS, AND PROGRAM NAMES ARE DISTINGUISHED FROM EACH OTHER BY FOUR CODE BITS IN BIT POSITIONS Ø-3 OF THE WORD CONTAINING THE RADIX 50 SYMBOL. THEY ARE

CODE ØØ	PROGRAM NAME
CODE Ø4	GLOBAL SYMBOL (INTERNAL)
CODE 1Ø	LOCAL SYMBOL
CODE 2Ø	DELETED ON OUTPUT (DDT)
CODE 4Ø	DELETED BY DDT OR THE LINKING LOADER
CODE 6Ø	GLOBAL REQUEST (EXTERNAL)

GLOBAL SYMBOL REQUESTS HAVE THEIR CODE BITS CHANGED FROM 6Ø TO 4Ø AS THEY ARE SATISFIED. CODE 2Ø IS CREATED BY DDT ONLY. WHEN LOADING IS COMPLETE, THE LOADER LEAVES A SYMBOL TABLE POINTER IN JOBSYM (LOCATION 11Ø IN THE USER AREA) WHICH CONTAINS THE ADDRESS OF THE LOWEST NUMBERED LOCATION IN THE SYMBOL TABLE IN THE RIGHT HALF, AND THE NEGATIVE WORD COUNT OF THE SYMBOL TABLE IN THE LEFT HALF.

THE RADIX 50 REPRESENTATION OF SYMBOLS

THIS IS A SPACE SAVING DEVICE USED TO CONDENSE A SIX CHARACTER SYMBOL INTO 32 BITS. THE SIX LETTERS OF THE SYMBOL (RIGHT) JUSTIFIED) ARE TRANSLATED INTO CODE NUMBERS. THE RIGHT HAND LETTER WILL BE DESIGNATED L1, AND ITS CORRESPONDING CODE NUMBER , C1, AND ACROSS FOR THE SIX LETTERS. THE RADIX 50 REPRESENTATION OF THE SYMBOL IS CALCULATED THUS:

(((((C6*50)+C5*50+C4)*50+C3)*50+C2)*50+C1 (OCTAL NUMBERS)
THE CODE NUMBERS CORRESPONDING TO THE SYMBOL CONSTITUENTS ARE:

CODE (OCTAL)	SYMBOL CHARACTER
ØØ	BLANK
Ø1-12	Ø-9
13-44	A-Z
45	.
46	$
47	(

RESTRICTIONS FOR ONE PASS ASSEMBLY

IN A ONE PASS ASSEMBLER, SYMBOL TAGS MAY BE REFERENCED BEFORE THEY ARE ASSIGNED A VALUE. WHEN THEY ULTIMATELY BECOME DEFINED, THE REFERENCE TO THEM HAS ALREADY BEEN GENERATED, AND THEREFORE MUST BE RESOLVED AT LOAD TIME. THIS UNRESOLVED REFERENCE MAY ACTUALLY BE A SERIES OF REFERENCES. TO AVOID CREATING TABLES OF UNRESOLVED REFERENCES, THEY ARE CHAINED TOGETHER BY THE ASSEMBLER IN SUCH A FASHION THAT THE LOADER MAY EASILY FIND AND RESOLVE ALL REFERENCES TO A PARTICULAR SYMBOL TAG. THE FIRST INSTANCE IN WHICH A REFERENCE TO A YET UNDEFINED SYMBOL TAG OCCURS CAUSES A NON-RELOCATABLE Ø TO BE GENERATED

AS THE VALUE OF THE TAG AND THE ADDRESS OF THIS REFERENCE IS KEPT IN THE SYMBOL TABLE BY THE ASSEMBLER. SUCCEEDING REFERENCES TO THE SAME TAG AND THE ADDRESS OF THE REFERENCE REPLACES THE ADDRESS OF THE PREVIOUS REFERENCE IN THE ASSEMBLERS SYMBOL TABLE THEREBY CREATING A CHAIN OF REFERENCES TO BE RESOLVED BY THE LOADER.

RESTRICTIONS FOR GLOBAL REQUESTS

GLOBAL SYMBOL REQUESTS OF THE ADDITIVE TYPE WILL BE ACCUMULATED BY THE ASSEMBLER AND OUTPUT IN A SEPARATE BLOCK (TYPE 2) WHENEVER NECESSARY. THIS BLOCK CAN OCCUR ANYWHERE ON A BINARY TAPE WITHIN A PROGRAM SEGMENT HAVING THE SAME RELOCATION CONSTANT AS THE DATA WORDS POINTED TO BY THE ADDITIVE REQUESTS. A DATA WORD POINTED TO BY AN ADDITIVE REQUEST MUST OCCUR ON THE BINARY TAPE BEFORE THE ADDITIVE REQUEST. THE LOADER WILL CHECK BIT ZERO OF A GLOBAL REQUEST TO DETERMINE WHETHER IT IS ADDITIVE OR CHAINED. A CHAIN IS A SERIES OF REFERENCES, EACH OF WHICH POINTS TO THE PREVIOUS REFERENCE. A CHAIN IS TERMINATED BY A NON-RELOCATABLE ZERO.

RESTRICTIONS FOR COMMON

THE PROGRAM WITH THE LONGEST DECLARED COMMON AREA MUST BE LOCATED FIRST. COMMON STARTS AT LOCATION 140. PROGRAMS ARE LOADED STARTING AT THE COMMON BREAK (THE FIRST FREE LOCATION ABOVE THE COMMON AREA). THEREFORE, ONCE LOADING HAS STARTED, THE COMMON AREA CANNOT BE EXPANDED.

C H A P T E R V I I

T H E E D I T O R

GENERAL

THE DECTAPE EDITOR PROVIDES A MEANS OF CREATING, ADDING TO, OR DELETING FROM SEQUENCE-NUMBERED LINES IN FILES ON DECTAPE. THIS TEXT MAY BE INTENDED AS INPUT FOR THE FORTRAN COMPILER, THE MACRO6 ASSEMBLER, OR JUST AS A CONVENIENT MEANS FOR HANDLING TEXTUAL OR LINES OF INFORMATION.

THE CHARACTERISTICS OF THE EDITOR

SIZE: 2K OCTAL
EQUIPMENT: USER TTY, 1 DECTAPE
LOADING: RUN AS USER PROGRAM,

USAGE

THE EDITOR PROVIDES THE FACILITY FOR SELECTING A TAPE UNIT, EXAMINING THE DIRECTORY (WHICH LISTS ALL THE FILES CURRENTLY ON THE TAPES), CLEARING THE DIRECTORY, ADDING A NEW FILE TO THE DIRECTORY, OR SELECTING A FILE CURRENTLY ON THE DIRECTORY. WHEN A FILE HAS BEEN SELECTED, THE USER MAY RESEQUENCE IT, PRINT A LINE OR MANY LINES, ENTER NEW LINES, DELETE EXISTING LINES, OR REPLACE EXISTING LINES.

ONCE CONTROL IS TURNED OVER TO THE EDITOR, THE EDITOR OPERATES IN EITHER OF TWO MODES: COMMAND MODE OR TEXT MODE. THE COMMAND MODE IS THE MODE IN WHICH THE USER ISSUES INSTRUCTIONS TO THE EDITOR.

COMMANDS

SX CONTROL A (CR)	(SELECT UNIT X AND CLEAR THE DIRECTORY.)
SX,NAME CONTROL A ALTMODE	(SAME AS ABOVE BUT CREATES FILE NAME AFTER CLEARING.)
SX,NAME (CR)	(ALLOW THE USER TO ACCESS FILE NAME ON UNIT X. IF NO SUCH FILE EXISTS, THE ERROR *NCF* WILL BE GIVEN.)
SX,NAME ALTMODE	(CREATE A NEW FILE CALLED NAME ON UNIT X. IF NAME ALREADY EXISTS THE ERROR *FAU* WILL BE GIVEN.)
E (CR)	(END THE CURRENT FILE.)
IN1 (CR)	(INSERT AT LINE N1, WHERE N1 MAY BE A SEQUENCE NUMBER OR THE SPECIAL CHARACTER . (POINT). IF N1 IS AN EXISTING LINE, IT IS REPLACED.)
IN1,I (CR)	(INSERT AT LINE N1, (AS ABOVE) BUT SET THE INDEXING INCREMENT TO I.)

DN1	(DELETE LINE N1.)
DN1,N2 (CR)	(DELETE LINES N1 THRU N2.)
PN1 (CR)	(PRINT LINE N1.)
PN1,N2 (CR)	(PRINT LINES N1 THRU N2.)
RX (CR)	(RESEQUENCE THE CURRENT FILE, STARTING AT X AND STEPPING BY X.)
RX,Y (CR)	(RESEQUENCE THE CURRENT FILE, STARTING AT X AND STEPPING BY Y.)

THE TEXT MODE IS ENTERED AUTOMATICALLY BY THE INSERT COMMAND. THE SEQUENCE NUMBER OF THE LINE BEING REFERENCED WILL BE PRINTED FOLLOWED BY A TAB. THE LINE BEING INSERTED IS THEN TYPED IN AND TERMINATED BY A CARRIAGE RETURN. THIS LINE MAY CONTAIN ANY CHARACTER EXCEPT RUBOUT AND ALTMODE. AFTER THE LINE BEING INSERTED HAS BEEN TERMINATED BY A CARRIAGE RETURN, THE INDEXING INCREMENT IS ADDED TO THE LINE NUMBERS AND THE INCREMENTED LINE NUMBER IS THEN PRINTED, FOLLOWED BY A TAB, AND THE EDITOR IS READY TO ACCEPT THE NEXT INSERTION. THE INDEXING INCREMENT IS INITIALLY SET TO 10.
TO CORRECT A TYPING ERROR, TYPE RUBOUT TO DELETE THE OFFENDING CHARACTER. THE EDITOR WILL RESPOND BY TYPING A \. SUCCESSIVE CHARACTERS MAY BE DELETED BY TYPING MORE RUBOUTS, UNTIL THE BEGINNING OF THE LINE HAS BEEN REACHED, AT WHICH POINT THE EDITOR WILL CEASE RESPONDING WITH \.

TO RETURN FROM THE TEXT MODE TO THE COMMAND MODE, TYPE ALTMODE.

CARRIAGE RETURN	(ENDS A COMMAND AND CAUSES EXECUTION. NULL COMMANDS ARE IGNORED.
.	(POINT. HAS THE VALUE OF THE LAST LINE NUMBER TYPED.)
ALTMODE	(SPECIAL DELINEATOR FOR S; ALSO, WHEN USED ALONE, ADVANCES . AND PRINTS THE LINE.)

ERROR MESSAGES

ILC	ILLEGAL COMMAND
NLN	A P OR D HAS MADE REFERENCE TO A NONEXISTENT LINE.
ILS	THE INDEX STEP (IN INSERT) HAS CAUSED A LINE L
ILS	THE INDEX STEP (IN INSERT) HAS CAUSED A LINE ALREADY IN THE FILE TO BE SKIPPED OVER.
ILR	THE INDEX STEP HAS CAUSED AN ALREADY EXISTENT LINE TO BE REFERENCED BY INSERT.
UNA	A DECTAPE UNIT HAS BEEN SELECTED THAT IS NOT AVAILABLE TO THE USER.
DCE	NO MORE ROOM ON THE DECTAPE DIRECTORY FILE.
NFO	A COMMAND HAS BEEN INVOKED THAT REQUIRES AN ACTIVE FILE,

BUT NO FILE HAS BEEN SELECTED (BY S).

FAU FILE NAME ALREADY IN USE.

NCF NOT A CURRENT FILE.

RESTRICTIONS

ONLY EDITS SEQUENCED MODE FILES.

C H A P T E R V I I I

T H E P E R I P H E R A L I N T E R C H A N G E P R O G R A M

(P I P)

GENERAL

PIP1 AND PIP2 ARE PDP-6 PROGRAMS TO TRANSFER DATA FROM ONE I/O MEDIUM TO ANOTHER. PIP PERFORMS SIMPLE EDITING FUNCTIONS DURING DATA TRANSMISSION.

CHARACTERISTICS OF PIP1, PIP2

SIZE: PIP1=2K, PIP2=4K OR MORE OCTAL
EQUIPMENT: TELETYPE FOR CONTROL, OTHER DEVICES AS REQUIRED.
LOADING: RUN AS USER PROGRAM, PIP2 MAY BE READ INTO CORE GREATER THAN 4K

USAGE
COMMANDS ACKNOWLEDGED BY PIP ARE COMPRISED OF DESTINATION AND SOURCE INFORMATION. DESTINATION IS SEPARATED FROM SOURCE BY THE SPECIAL CHARACTER (←). MULTIPLE SOURCES MAY BE INCLUDED IN ONE COMMAND LINE.

PIP RESTRICTIONS: THE * COMMANDS BELOW ARE NOT AVAILABLE IN PIP1.

COMMANDS FORM

DESTINATION (SPECIAL OPERATORS)←SOURCE,SOURCE,...,SOURCE

THE DESTINATION MAY BE ONE OF THE FOLLOWING FORMS:

1) DEVICENAME:

2) DEVICENAME:FILENAME

3) DEVICENAME:FILENAME.EXTENSION

DEVICE NAMES ARE THE SAME FORM AS THOSE USED BY THE I/O SERVICE ROUTINES. FOR EXAMPLE, PTP IS THE PAPER TAPE PUNCH, DTA3 IS DECTAPE UNIT 3. FILE NAMES AND EXTENSIONS ARE RELEVANT ONLY FOR DEVICES THAT CONTAIN MULTIPLE FILES SUCH AS DECTAPE. FILE NAMES AND EXTENSIONS ARE CONSTRUCTED OF LETTERS AND NUMERALS. ONLY THE FIRST SIX CHARACTERS OF THE FILE NAME AND THE FIRST THREE LETTERS OF THE EXTENSION ARE USED.

SOURCES HAVE THE SAME FORM AS THE DESTINATION WITH THE EXCEPTION THAT IF THE DEVICE NAME IS OMITTED, THE LAST DEVICE NAME TYPED IS USED.

CARRIAGE RETURN TERMINATES THE COMMAND REQUEST.

RUBOUT MAY BE USED TO ERASE SUCCESSIVE CHARACTERS. FOR EACH CHARACTER ERASED, ONE REVERSE SLASH IS TYPED UNTIL ALL CHARACTERS ARE ERASED.

SPECIAL OPERATORS ARE LETTERS ENCLOSED IN PARENTHESIS(). IF ONLY ONE SPECIAL OPERATOR IS TYPED PRECEEDING SLASH

CHARACTER MAY REPLACE BOTH PARENTHESES.
IF SPECIAL OPERATORS ARE OMITTED, PIP TRANSMITS THE DATA
FROM EACH SUCCESSIVE SOURCE TO THE DESTINATION WITHOUT ALTERATION EXCEPT
THAT WHEN AN UNSEQUENCED SOURCE FILE FOLLOWS A SEQUENCED FILE,
SEQUENCE NUMBERS ARE ADDED TO THE UNSEQUENCED FILE AUTOMATICALLY.

PIP RECOGNIZES THE FOLLOWING SPECIAL OPERATORS:

SPEC. OPER.	ACTION
A	ASCII LINE MODE PROCESSING. (DO NOT BREAK UP LINES BETWEEN BUFFERS.)
B	PROCESS BINARY DATA
C*	SUPPRESS TRAILING SPACES AT THE END OF EACH LINE AND CHANGE MULTIPLE BLANKS INTO TABS. (USEFULL FOR CARD INPUT)
D*	DELETE ALL SOURCE FILE NAMES FROM THE DIRECTORY OF THE DESTINATION DEVICE. THE DESTINATION DEVICE NAME SUPERCEDES ANY OTHER DEVICE NAME THAT IS TYPED.
L*	LIST THE DIRECTORY OF THE SOURCE DEVICE. ONLY ONE SOURCE MAY BE SPECIFIED.
N	REMOVE SEQUENCE NUMBERS
R*	RENAME THE SOURCE FILE TO HAVE THE NAME OF THE DESTINATION FILE. NO SOURCE DEVICE NAME IS NECESSARY BECAUSE THE DESTINATION DEVICE IS IMPLIED. ONLY ONE SOURCE MAY BE SPECIFIED.
S	A SEQUENCE NUMBER STARTING WITH 00010 AND INCREMENTING BY 10 IS ADDED TO THE FRONT OF EACH LINE. A TAB IS INSERTED AFTER THE SEQUENCE NUMBER UNLESS THE LINE CONSISTS ENTIRELY OF FORM FEED, VERTICAL TAB, OR LINE FEED, IN WHICH CASE A CARRIAGE RETURN IS INSERTED. IF A FILE IS ALREADY SEQUENCED, THE OLD SEQUENCE NUMBERS ARE REMOVED.
T	THE SAME AS C EXCEPT DO NOT REPLACE MULTIPLE BLANKS WITH TABS.
W	REWIND MAGNETIC TAPE. SOURCE FILES ARE IGNORED.
X*	COPY AN ENTIRE DECTAPE, RECOVERING UNUSED BLOCKS. EMPTY FILES ARE NOT COPIED. DOES NOT DISTURB DATA PREVIOUSLY WRITTEN ON THE DESTINATION TAPE UNLESS FILE NAMES CONFLICT. ALTHOUGH THE SOURCE DEVICE MUST BE DECTAPE, THE DESTINATION MAY BE ANY OUTPUT DEVICE.
Z	CLEAR THE DESTINATION DECTAPE AND CREATE A FRESH DIRECTORY. SOURCES ARE IGNORED.

ERROR MESSAGES

DEVICE XXX NOT AVAILABLE	THE DEVICE NAMED XXX IS NOT AVAILABLE

	EITHER BECAUSE IT DOES NOT EXIST OR SOME OTHER USER IS USING IT.
NO FILE NAMED XXX	THE SOURCE FILE NAMED XXX WAS NOT FOUND IN THE DIRECTORY.
DIRECTORY FULL	THE DESTINATION DEVICE HAS NO ROOM IN THE DIRECTORY FOR ANOTHER FILE NAME. THE ONLY RECOURSE IS TO DELETE ONE OF THE PREVIOUS FILES ON THE DESTINATION TAPE TO MAKE ROOM IN THE DIRECTORY.
DECTAPE INPUT REQUIRED	A DIRECTORY MANIPULATION COMMAND WAS GIVEN FOR A DEVICE OTHER THAN DECTAPE.

EXAMPLES

LPT:←DTA1:ABC.TXT,XYZ	THE TWO FILES ON DECTAPE UNIT 1 NAMED ABC.TXT AND XYZ ARE LISTED ON THE LINE-PRINTER.
DTA2:SUBR/S←PTR:	A FILE NAMED SUBR IS CREATED ON DECTAPE UNIT 2 CONTAINING THE ASCII DATA FROM THE TAPE IN THE PAPER TAPE READER WITH SE-QUENCE NUMBERS ADDED.
DTA1:(D)←ABC.TXT,XYZ	DELETE THE FILES NAMED ABC.TXT AND XYZ

EXAMPLES: PIP1 AND PIP2

TO COPY THREE PAPER TAPES ONTO ONE DECTAPE FILE, TYPE:

```
DTA1:BIGFIL←PTR:,,
```

THE SAME AS ABOVE, BUT WITH SEQUENCE NUMBERS, TYPE:

```
DTA1:BIGFIL/S←PTR:,,
```

TO LIST THIS FILE, TYPE:

```
LPT:←DTA1:BIGFIL
```

THE SAME AS ABOVE, BUT WITHOUT SEQUENCE NUMBERS, TYPE:

```
LPT:/N←DTA1:BIGFIL
```

TO REWIND MAG TAPE:

```
MTA1/W←
```

TO COPY BINARY DATA:

```
DTA:1PIP2.REL/B←PTR:
```

TO CREAT A FRESH DIRECTORY:

```
DTA1:/Z←
```

PIP2 EXAMPLES:

TO CLEAN UP A DECTAPE AND RECOVER UNUSED BLOCKS, MOUNT SOURCE TAPE ON UNIT (SAY) 1 AND A SCRATCH TAPE ON UNIT (SAY) 2. THEN TYPE:

```
DTA2:/Z←
DTA2:/X←DTA1:
```

RETURN YOUR OLD TAPE TO THE SCRATCH TAPE SUPPLY AND TAKE POSITION OF THE TAPE ON UNIT 2.

TO RENAME FILES, TYPE:

```
DTA2NEWNAM.EXT/R←OLDNAM
DTA2:AFTER/R←BEFORE
```

```
                    ----------------------

TO DELETE FILES, TYPE:
                    ----------------------

               DTA2:/D←FILE1,FILE2,FILE3

                    ----------------------


TO LIST A DIRECTORY:
                    ----------------------

               TTY:/L←DTA2

                    ----------------------
```

C H A P T E R I X

B A T C H C O N T R O L P R O G R A M

(B C P)

BCP IS A 1024 WORD (DECIMAL) PROGRAM TO CONTROL A STACK OR BATCH OF PROGRAMS WHICH HAVE BEEN SUBMITTED FOR PROCESSING.

THE USER SUBMITS A DECK OF CARDS, A SEQUENCE OF PAPER TAPES, A MAGNETIC TAPE, OR DECTAPE FILES WHICH CONTAIN THE SOURCE PROGRAM AND DATA FOR THE JOBS, WITH INTERSPURSED CONTROL CARDS. THE LANGUAGE OF THE CONTROL CARDS IS IDENTICAL TO THAT OF THE CONSOLE USER COMMANDS.

BCP RUNS AS A SEPARATE JOB, INDEPENDENT OF THE STACK, AND INTERPRETS THE CONTROL CARDS ON BEHALF OF THE STACK. BCP ALSO TRANSMITS INDIVIDUAL PROGRAM ERROR CONDITIONS BACK TO THE OPERATOR WHO ATTENDS.

APPENDIX OF MULTIPROGRAMMING EXECUTIVE COMPONENTS

THE EXECUTIVE COMPONENTS RESIDE IN SEPARATE FILES WHICH MAY BE ASSEMBLED INDEPENDENTLY USING THE SYSTEM TAPE SYSPAR. SYSPAR CONTAINS THE COMMON SYSTEM DEFINITIONS, MACROS, BIT, AND AC ASSIGNMENTS FOR ALL ROUTINES. EACH ROUTINE WHICH IS REQUIRED TO LINK WITH OTHER SYSTEM ROUTINES CONTAINS THE APPROPRIATE INTERNAL AND EXTERNAL LIST OF MNEMONICS.

DIRECT JOB CONTROL ROUTINES
THESE ROUTINES ARE DIRECTLY ASSOCIATED WITH THE USER OR A JOB. THEY HANDLE THE CONSOLE COMMUNICATIONS, SCHEDULING, ETC.

UUO HANDLER
PROCESSES THE UUOS (OR PROGRAMMED OPERATORS) THAT ORIGINATE WITH THE USER. IF THE COMMANDS ARE USER DEFINED, IT RETURNS CONTROL TO THE USER AREA, OTHERWISE IT CALLS THE APPROPRIATE SYSTEM COMMAND SECTION TO HANDLE THE FUNCTION.

CALL HANDLER
PROCESSES THE SPECIAL UUO, CALL.

COMMAND DECODER
PROCESSES THE COMMANDS WHICH ORIGINATE FROM THE USER'S CONSOLE. ALL SYSTEM CALLS FROM THE CONSOLE E.G. GET, SAVE, ASSIGN, ETC. ARE DECODED AND PROCESSED BY THE COMMAND DECODER. THE COMMAND DECODER MAY CALL OTHER SYSTEM ROUTINES, E.G. CORE, TO DO SPECIFIC FUNCTIONS.

CORE
ALLOCATES CORE STORAGE FOR ALL THE USERS. IT MAY BE CALLED BY A JOB OR BY A USER CONSOLE COMMAND AND IS A REQUEST FOR A SPECIFIC AMOUNT OF CORE STORAGE. ABSOLUTE REQUESTS ARE MADE AND CORE MAY BE EXTENDED OR WITHDRAWN.

SCHEDULER
SCHEDULES THE RUNNING OF JOBS. IT IS CALLED AT REGULAR INTERVALS TO DECIDE WHICH PROGRAM IS TO BE RUN NEXT. A RUNNING PROGRAM IS TEMPORARILY TERMINATED EACH TIME ITS ALLOTED COMPUTATION TIME EXPIRES, OR WHEN A REQUEST FOR INPUT OR OUTPUT REQUIRES WAITING. A USER MAY HALT A JOB FROM A CONSOLE AND TEMPORARILY SUSPEND OPERATION. THE ALGORITH FOR CONTROL IS ON A ROUND-ROBIN BASIS, WITH JOBS WHICH HAVE JUST COMPLETED INPUT OR OUTPUT RECEIVING HIGHER PRIORITY.

RUN
CONTROLS THE ACTUAL STARTING AND STOPPING OF JOBS AND ALL THE SYSTEM DATA ASSOCIATED WITH A JOB IS PROCESSED BY RUN.

ERRCON
COMMUNICATES JOB ERRORS TO A USER THROUGH HIS CONSOLE. ILLEGAL MEMORY REFERENCES, DATA MODE TRANSMISSION ERRORS, ETC. ARE TRANSMITTED TO THE USER THROUGH THIS ROUTINE.

EXECUTIVE DATA
THE SYSTEM ROUTINES ARE EITHER PURE PROCEDURES OR GO TO COMPLETION FOR A GIVEN CALL. THE DATA FOR THE SYSTEM IS KEPT IN SPECIAL AREAS.

CLOCK QUEQUE DATA
CONTAINS ALL THE REQUESTS FOR CLOCK PROCESSING OR TIME

SERVICE. THE AMOUNT OF TIME REQUESTED TO TIME AND THE LOCATION TO TRANSFER ON COMPLETION OF TIMED INTERVAL.

CALL DATA

CONTAINS ALL THE TABLES FOR THE SYSTEM FUNCTIONS AVAILABLE THROUGH THE UUO CALL. ENTRIES FOR THE CALL NAME, AND THE LOCATION OF THE ROUTINE ARE IN CALTAB.

JOBSTS TABLE

CONTAINS THE SYSTEM DATA ASSOCIATED WITH A PARTICULAR JOB WHILE THE JOB IS RUNNING. JOBSTS ALSO CONTAINS ENTRIES FOR EACH JOB IN THE SYSTEM. THESE ARE THE ADDRESS OF THE JOB AND ITS STATUS (IO WAIT, DORMANT, AND ACTIVITY) THAT THE SCHEDULER REQUIRES.

TTYTAB

CONTAINS THE STATUS OF ALL THE TELETYPE CONSOLES. THIS IS THE INFORMATION FOR THE ASSIGNMENT OF TELETYPE BUFFERS TO SPECIFIC JOBS, THE PROGRAMS TO WHICH A TELETYPE IS COMMUNICATING, WHETHER COMMAND DECODER OR SCHEDULER SERVICE IS REQUIRED, ETC.

COMMON JOB FUNCTIONS AND INTERNAL SYSTEM FUNCTIONS

CLOCK

HANDLES REQUESTS FROM ROUTINES FOR TIMING SERVICE. FOR EXAMPLE, DECTAPE ROUTINES REQUEST CLOCK TIMING INFORMATION IF A DECTAPE IS MADE TO FREE RUN FOR A PREDETERMINED PERIOD OF TIME. THE SCHEDULER USES THE CLOCK TO TIME JOB RUNNING INTERVALS.

IOCSS

CONTAINS ALL COMMON SYSTEM SUBROUTINES FOR SYSTEM IO CONTROL. THESE INCLUDE ROUTINES WHICH MOVE BUFFER POINTERS, ETC.

IOCONT

CONTAINS THE SPECIAL PROCESSING FOR THE IO DEVICE UUOS.

SYSCOM

CONTAINS SYSTEM UUO PROCESSING ROUTINES.

SYSCSS

CONTAINS COMMON SYSTEM SUBROUTINES.

IOINI

CONTAINS A LIST OF SPECIAL SETUP COMMANDS REQUIRED FOR IO DEVICES INITIALIZATION.

SYSINI

CONTAINS BASIC SYSTEM INITIALIZATION ROUTINES. THESE RESET JOBS, CALL ROUTINES WHICH HAVE DATA ASSOCIATED WITH THEM TO BE RESET, INCLUDING CLOCK DATA, CORE DATA, TELETYPE ASSIGNMENTS, JOB ASSIGNMENTS, ETC.

CREATION AND DEBUGGING

THESE PROGRAMS ARE USED TO CREATE OTHER SYSTEMS WHICH RUN IN EXECUTIVE MODE. THEY ALSO ALLOW THE DEBUGGING PROGRAM DDT TO BE LOADED WITH A NEW SYSTEM, AND ITS SYMBOL TABLE PRESERVED.

SYSMAKER

IS A PROGRAM IN THE SYSTEM WHICH IS RUN ANY TIME A PROGRAM HAS BEEN CREATED IN USER MODE AND IS TO BE RUN IN EXECUTIVE MODE IN THE POSITIONS OCCUPIED BY THE FORMER SYSTEM. WHEN IT IS CALLED, SYMBOLS WHICH RESIDE IN THE USER AREA PROGRAM ARE FIRST MOVED TO THE UPPER AREA OF MEMORY, AND THE PROGRAM IS MOVED DOWN OVER THE OLD SYSTEM, INCLUDING SYSMAKER. IN THIS FASHION A NEW SYSTEM CAN BE LOADED AS A NORMAL TIME SHARED JOB, AND THEN BROUGHT DOWN INTO THE USER EXECUTIVE AREA.

SYSDDT

IS A DDT WHICH MAY BE USED TO DEBUG THE SYSTEM. IT ASSUMES THAT A SYMBOL TABLE WILL BE IN THE HIGHEST SYSTEM MEMORY LOCATIONS AND IO IS DONE DIRECTLY BY DDT.

APPENDIX OF MONITOR CORE ALLOCATION

EXECUTIVE COMPONENTS

EXECUTIVE COMPONENTS

------	140
SYSINI	42
SYSCON	231
SYSCSS	25
IOCONT	476
IOCSS	337
RUNCSS	244
ERRCON	405
CLOCK	146
CLKCSS	102
IOINIT	1573
COMINI	135
COMCON	200
COMCSS	161
SAVGET	165
CORE	112
SYSMAK	36

IO CONTROL MODULES

MTPSER	546
LPT	162
PTR	156
SCN	757
PTP	256
DTSER	1251
CDR	261

DDT (MONITOR DEBUGGING)

DDT	3126

TOTAL	15227

TABLE OF PHYSICAL DEVICE CHARACTERISTICS

MNEMONIC	NAME	OPTION NUMBER	APPROX. SERVICE ROUTINE SIZE (OCTAL)
TTY, TTY0, TTY1,...TTYN	CONSOLE TELEPRINTER AND/OR HALF/FULL DUPLEX TELEPRINTERS CONNECTED TO 630 DATA COMMUNICATIONS SYSTEM	626, 630 HALF/ FULL DUPLEX 33/35 TELEPRINTERS	600+(31) X NO. OF ACTIVE TELETYPES
PTR	PAPER TAPE READER	760	220
PTP	PAPER TAPE PUNCH	761	300
LPT	LINE PRINTER	646 A,B,C	200
CDR	CARD READER	461 (200 OR 800 CPM READER)	300
DTA,DTA1, ...DTA8	DECTAPE	551. 1 TO 4-555. SHARES 1-136 WITH 516	1300+(200+2)X ACTIVE DECTAPES
MTA,MTA1, ...MTA8.	MAGNETIC TAPE	516 WITH 520, 521, OR 522 FOR 1 TO 8 TYPE 50,570 OR 729VI TRANSPORTS SHARES 7-136 WITH 551.	700+20 X ACTIVE MAGTAPES
DIS	DISPLAY	346	400

TABLE OF DEVICE PROGRAM CHARACTERISTICS

MNEMONIC	BUFFER SIZE (OCTAL)	COMMANDS*	DATA MODES	DATA CONTROL MODES***	STATUS BITS**	ENTER, LOOKUP COMMANDS
TTY,TTY0 TTY1, ...TTYN	17	INPUT, OUTPUT	AL(INPUT) A=AL(OUTPUT)			
PTR	41	INPUT	A,AL,B,I(8 BITS)		IODTER	
PTP	41	OUTPUT	A=AL,B,I(8 BITS)			
LPT	32	OUTPUT	A=AL			
CDR	34	INPUT	A=AL,B,I(12 BITS)		IODTER	
DTA. DTA1, ...DTA8	200	INPUT, OUTPUT, USETO, USETI, UGETF	A=AL=B,D.	IOROEL	IOBKTL	X
MTA MTA1, ...MTA8.	200	INPUT, OUTPUT, MTAPE, GETN	A=AL=B,D	IODENS, IOPAR, IONRCK	IOBKTL, IOBOT, IOTEND.	X
DIS		OUTPUT (SPECIAL DUMP)	D.			

* IN ADDITION TO : ASSIGN, INIT, LOOKUP, ENTER, INBUF, OUTBUF, STATUS, STATZ, STATO, CLOSE, RELEAS, DEASSIGN

** IN ADDITION TO: IOIMPM, IODEND, IOACT, IODERR, MODE BITS 26-35.

***IN ADDITION TO : IOWC, IOCON.

SYSTEMS LIBRARY FILE CONTENTS 3 MARCH 1965

PROGRAM	MNEMONIC	MODE	SIZE(OCTAL)
DIRECTORY			
SYSTEM DIRECTORY			200
EXECUTIVE			
MULTI-PROGRAMMING SYSTEMS		DECDUMP	12000
TRANSLATORS			
MACRO ASSEMBLER	MACRO	DUMP	22000
FORTRAN II (MIN.)	FOR2S	DUMP	22000
FORTRAN II (MAX.)	FOR2L	DUMP	46000
FORTRAN II (MAX.)FOLA	FORFOL	DUMP	56000
FORTRAN ON LINE ASSEMBLER	FOLA	DUMP	10000
OTHER CUSPS			
PERIPHERAL INTERCHANGE	PIP1	DUMP	2000
PERIPHERAL INTERCHANGE	PIP2	DUMP	4000
EDITOR	EDIT	DUMP	2000
LOADER	LOAD	DUMP	2200
RELOCATABLE LINKING LOADER FILES			
DDT FOR USER	USRDDT	RELBIN	5000
FORTRAN OP SYSTEM	FORSE	RELBIN	4000

LOADING THE SYSTEM

1. TURN ON COMPUTER (POWER SWITCH ON CONSOLE)

2. TURN ON ALL IO DEVICES INCLUDING SCANNER.

3. LOAD DECDUMP INTO TOP OF MEMORY
   A. MOUNT A DECTAPE WITH DECDMP AND DECDUMP LOADER ON UNIT #1 (SEE DECTAPE LOADERS DESCRIPTION)
   B. SET AD4RESS SWITCHES TO Ø
   C. PUSH STOP SWITCH DOWN AND RETURN
   D. LIFT START SWITCH UP
   E. DECDUMP WILL RESPOND WITH CR-LF AFTER LOADING.

4. LOAD TIME SHARING MONITOR FROM DECTAPE.
   A. TYPE NL WHERE N IS THE NUMBER OF THE TAPE DRIVE ON WHICH THE MONITOR HAS BEEN MOUNTED.
   B. DECDUMP WILL RESPOND WITH CR-LF WHEN LOADING HAS COMPLETED.
   C. TYPE G TO BEGIN EXECUTION.
   D. AFTER A SECOND OR TWO THE MONITOR WILL TURN ON ALL OF THE PI CHANNELS AND WILL BE READY TO ACCEPT TYPE IN. WHENEVER THE MONITOR HAS NOTHING TO DO, IT WILL RUN THE NULL JOB. THE NULL JOB COUNTS IN ACØ AND RUNS IN AC1. THEREFORE, IT CAN BE DETECTED BY A 1 IN THE PROGRAM COUNTER LIGHTS. THE COUNT IN ACØ BEGINS OVER EACH TIME THE NULL JOB BEGINS TO RUN SO THAT SETTING THE ADDRESS SWITCHES TO Ø WILL GIVE A ROUGH IDEA OF THE SYSTEM IDLE TIME.

5. THE OPERATOR INITIALIZES A JOB AND KEEPS IT WHILE THE SYSTEM IS RUNNING. HE ALSO NAMES HIS CONSOLE OPER SO THAT USER'S MAY COMMUNICATE WITH HIM. TO DO THIS HE TYPES THE FOLLOWING COMMANDS:
   PJOB (WHICH WILL PRINT HIS JOB NUMBER WHICH MUST BE 1)
   ASSIGN TTY:OPER (WHICH WILL GIVE THE OPERATOR'S TELETYPE TTY THE NAME OPER. HE SHOULD THEN ASSIGN TO HIMSELF ANY IO DEVICES WHICH SHOULDN'T BE USED BY USERS BY TYPING '
   ASSIGN DEV
   WHERE DEV IS THE PHYSICAL DEVICE NAME (DTA3 FOR EXAMPLE).

A USER MAY COMMUNICATE WITH THE OPERATOR BY TELETYPE BY TYPING THE FOLLOWING:

```
ATTACH 1
CORE 1
GET DTAØ:PIP
START
OPER←TTY:
```

THE MESSAGE TERMINATED BY THE ASCII END OF MESSAGE CHARACTER <CONTROL>Z.
NOTE: IT IS A GOOD IDEA TO BEGIN AND END WITH CARRIAGE RETURN.

```
<CONTROL>C
CORE Ø
ATTACH N
```

MONITOR ERROR HALTS

THE MONITOR CONTAINS NO ERROR HALTS EXCEPT FOR "IMPOSSIBLE" ERROR CONDITIONS. IF A SO CALLED "IMPOSSIBLE" ERROR HALT OCCURS, WRITE DOWN BOTH THE CONTENTS OF THE PROGRAM COUNTER AND THE MEMORY ADDRESS. THEN TRY PRESSING CONTINUE. IF THE MACHINE HALTS AT THE SAME PLACE IMMEDIATELY, THERE IS PROBABLY A HARDWARE MALFUNCTION. THE OPERATOR SHOULD EITHER CALL IN THE MAINTENANCE MAN OR ATTEMPT THE ERROR RECOVERY PROCEDURE LISTED BELOW.

ERROR RECOVERY PROCEDURE (IN ORDER OF INCREASING SEVERITY)

1. PRESS STOP SWITCH AND HOLD DOWN.
   PRESS CONTINUE SWITCH (TO MAKE SURE IN EXEC MODE)(DO NOT PRESS IO RESET)
   RELEASE STOP SWITCH
   SET ADDRESS SWITCHES TO 144
   PRESS START
   THIS WILL BEGIN BY RUNNING THE NULL JOB.
   THIS WILL AFFECT AT MOST ONE JOB WHICH THE USER WILL HAVE TO RESTART.

2. PRESS STOP SWITCH
   PRESS IO RESET
   SET ADDRESS SWITCHES TO 140
   PRESS START
   THIS WILL INITIALIZE ALL IO DEVICES INCLUDING THE TELETYPES. ALL CONSOLES WILL BE IN COMMAND MODE AND EACH USER MUST ATTACH HIS CONSOLE BACK TO HIS JOB. ALL JOBS MUST BE STARTED OVER.

3. PRESS STOP SWITCH
   PRESS IO RESET
   SET ADDRESS SWITCHES TO 143
   PRESS START
   THIS WILL REINITIALIZE THE MONITOR TO THE STATE IT WAS IN JUST AFTER BEING LOADED. ALL JOBS MUST BE RELOADED.

4. PRESS STOP SWITCH
   RELOAD DECDUMP
   RELOAD MONITOR FROM DECTAPE.

OCCASIONALLY A SYSTEM WILL GET INTO A LOOP. THIS CAN BE DETECTED BY SEEING THAT THE COMPUTER IS NEVER LEAVING EXEC MODE AND IS NOT RUNNING THE NULL JOB (PC=1). USERS WILL REPORT THAT SOMETHING APPEARS TO BE WRONG SO THAT THE OPERATOR DOES NOT NEED TO WATCH FOR THIS CONDITION. WHEN SUCH A LOOP OCCURS, NOTE WHETHER AN INTERRUPT IS IN PROGRESS. IF NOT, PRESS STOP AND LOOK AGAIN. PRESS CONTINUE AND STOP UNTIL NO INTERRUPTS ARE IN PROGRESS. THEN WRITE DOWN THE CONTENTS OF THE PROGRAM COUNTER AND MEMORY ADDRESS. THEN FOLLOW ERROR RECOVERY PROCEDURE.

```
,OPDEF TAPE FOR EXEC SYSTEM IO
,DECEMBER 19,1964


        OPDEF ENTER [77B8]
        OPDEF LOOKUP [76B8]
        OPDEF USETO [75B8]
        OPDEF USETI [74B8]
        OPDEF UGETF [73B8]
        OPDEF MTAPE [72B8]
        OPDEF RELEAS [71B8]
        OPDEF CLOSE [70B8]
        OPDEF OUTPUT [67B8]
        OPDEF INPUT [66B8]
        OPDEF OUTBUF [65B8]
        OPDEF INBUF [64B8]
        OPDEF STATZ [63B8]
        OPDEF STATUS [62B8]
        OPDEF STATO [61B8]
        OPDEF INIT [41B8]
        OPDEF CALL [40B8]
```

```
TITLE  JOBDAT - JOB DATA AREA ASSIGNMENTS
SUBTTL  MARCH 12, 1965

          DEFINE BLK (A,B)
          <JOBPRO=B
          A=Ø
          INTERNAL A
          DEFINE BLK (C,D)
          <C=JOBPRO
          JOBPRO=JOBPRO+D
          INTERNAL C>>

          BLK JOBAC,2Ø          ;SYSTEM USE AC STORAGE
          BLK JOBDAC,2Ø         ;USER'S AC STORAGE WHILE JOB IS INACTIVE
          BLK JOBUUO,2          ;UUO SAVE AND TRAP LOCATIONS
          BLK JOBPC,1           ;BITS Ø-5=APR FLAGS, C(RH)=PC
          BLK JOBPDP,1          ;PUSH DOWN POINTER FOR SYSTEM USE
          BLK JOBREL,Ø          ;C(LH)=UNUSED, C(RH)=RELOCATION ADDRESS
          BLK JOBUXT,1          ;EXIT FROM UUO AT BREAK
          BLK JOBLEV,1          ;UNUSED
          BLK JOBSAV,1          ;TEMPROARY STORAGE FOR UOO HANDLER
          BLK JOBPOV,1          ;PUSH DOWN OVERFLOW TRAP
          BLK JOBTEM,2          ;TEMPORARY SYSTEM STORAGE
          BLK JOBINF,1          ;BITS Ø-11=JOB NUMBER
          BLK JOBJDA,2Ø         :JOB IO DEVICE CHANNEL ASSIGNMENTS
          BLK JOBDDT,1          ;ADDRESS OF USER'S DDT
          BLK JOBCDP,1          ;POINTER TO COMMAND STRHNG
          BLK JOBTRP,2Ø         ;TRAP LOCATIONS
          BLK JOBSYM,1          ;SYMBOL TABLE POINTER
          BLK JOBUSY,1          ;POINTER TO UNDEFINED SYMBOL TABLE
          BLK JOBSA,1           ;C(RH)=STARTING ADDRESS
                                ;C(LH)=HIGHEST LOCATION USED
          BLK JOBFF,1           ;FIRST FREE LOCATION
          JOBPDL=JOBPRO-1       ;FIRST LOC. -1 OF PD LIST
          MJOBPD=JOBPDL-137     ;- LENGTH OF PD LIST

          INTERNAL JOBPDL,JOBPRO,MJOBPD
END
```

```
          XLIST


DEFINE    XP(A,B)              ;SYSTEM PARAMETER
          <A=B
          INTERNAL A
>
,ACCUMULATORS
          XP IOS,0;            IO STATUS WORD
          XP TAC,1;            TEMPORARY
          XP TAC1,2;                    TEMPORARY
          XP PDP,3;            C(LH)=NUMBER OF LOCATIONS LEFT IN PD LIST
,                              C(RH)=PUSHDOWN POINTER
          XP SUB,10            RETURN (PC) FOR ONE LEVEL SUBROUTINES
          XP ITEM,4;           BYTE OR WORD POINTER,  ITEM COUNT
          XP DAT,5;            DATA OR TEMPORARY
          XP DEVDAT,6;         C(LH)=UNUSED,
,                              C(RH)=ADDRESS OF DEVICE DATA BLOCK
          XP JBUF,DAT;         ADDRESS OF JOB BUFFER AREA=JBFADR
          XP PROG,7;                    C(LH)=UNUSED
,                              C(RH)=ADDRESS OF USER'S PROGRAM AREA
          XP JDAT,11;          C(RH)=ADDRESS OF JOB DATA
,ONLY 0 TO 10 NORMALLY REQUIRED
,FOR INTERRUPT SERVICE
          XP BUFPNT,12;        C(JBFADR 18-35)
          XP BUFWRD,13;        BIT 0=IOUSE
,                              BITS 1-17=BUFFER SIZE
,                              BITS 18-35=NEXT BUFFER ADDRESS
          XP UUO,15;           LAST UUO PROCESSED
          XP JB,14; JOB NUMBER
          XP AC1,16;           UNUSED
          XP AC2,17;           UNUSED
```

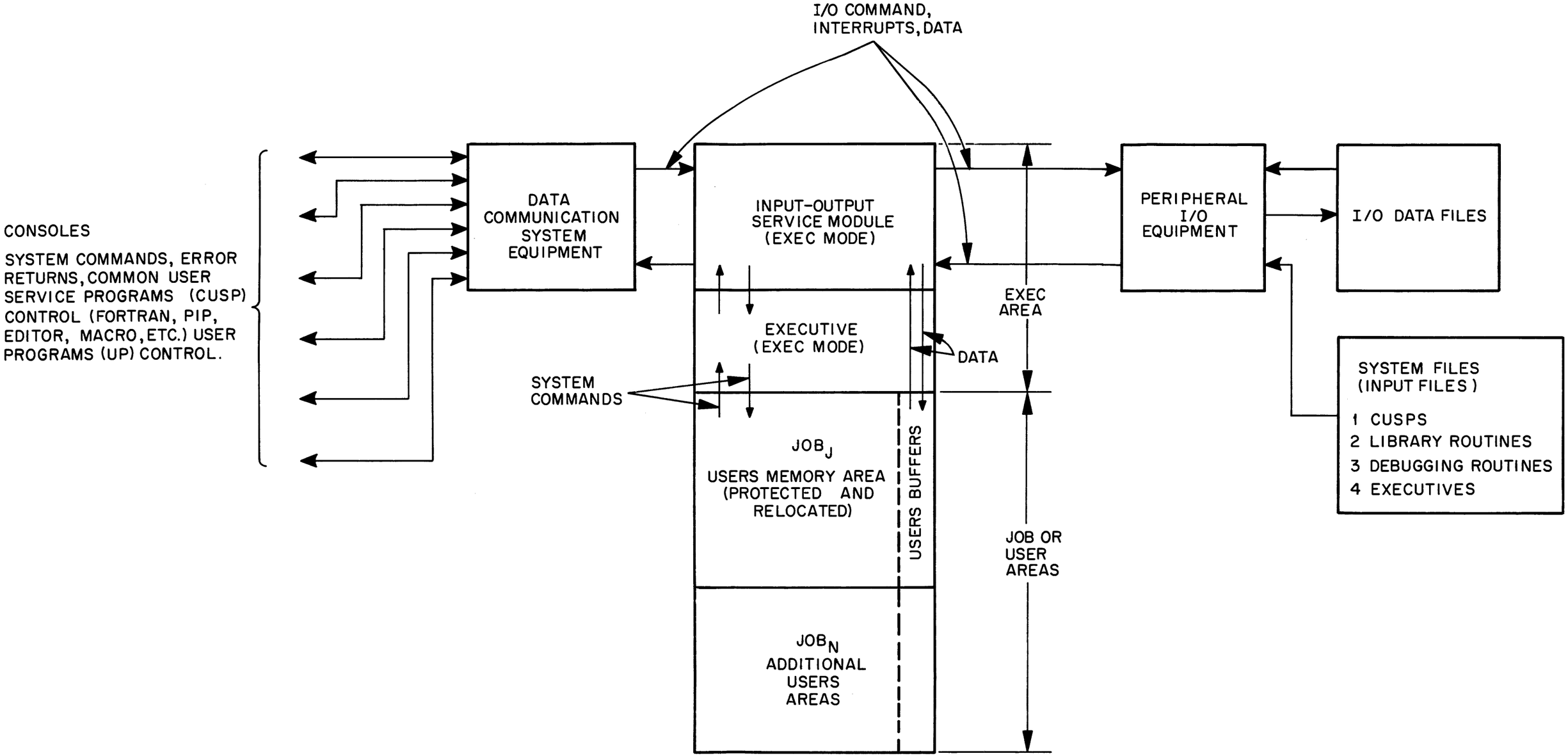
```
, DEVICE DATA BLOCK NAMES
          XP DEVNAM,0;          NAME IN SIXBIT ASCII
,                               C(LH)=DEVICE MNEMONIC
,                               C(RH)=DEVICE NUMBER, LEFT JUSTIFIED
          XP DEVCHR,1;          CHARACTERISTIC
,                               BITS 0-8=JOB NUMBER
,                                    ZERO VALUE IMPLIES NOT ASSIGNED
,                               BITS 9-11=PI PRIORITY CHANNEL FOR DEVICE
,                               BIT 12=UNUSED
,                               BIT 13=IORET. 0 IF DEVICE HAS A SHORT
,                                   DISPATCH TABLE.  1 IF LONG
,                               BITS 14-17=JOB DEVICE CHANNEL NUMBER
,                               BITS 18-23=DEVICE NUMBER, BINARY
,                               BITS 24-35=BUFFER SIZE
          XP DEVIOS,2;          STATUS WORD.  SEE BELOW
          XP DEVSER,3;          C(LH)=NEXT DEVICE DATA BLOCK
,                               C(RH)=DEVICE SERVICE DISPATCH TABLE
,    DEVICE SERVICE DISPATCH TABLE ASSIGNMENTS
          XP DRL,0;                  XXXDSP   :  RELEASE
          XP DCL,1;                  XXXDSP +1:  CLOSE
          XP DOU,2;                  XXDSP +2:  OUTPUT
          XP DIN,3;                  XXXDSP +3:  INPUT;SHORT DISPATCH TABLE
          XP DEN,4;                  XXXDSP +4:  ENTER
          XP DLK,5;                  XXXDSP +5:  LOOKUP
          XP DDO,6;                  XXXDSP +6:  DMPO
          XP DDI,7;                  XXXDSP +7:  DMPI
          XP DSO,10;                 XXXDSP+10:  SETO
          XP DSI,11;                 XXXDSP+11:  SETI
          XP DGF,12;                 XXXDSP+12:  GETF;LONG DISPATCH TABLE
          XP DEVMOD,4           ;BITS 6-11=LEFT HALF OF IMAGE MODE BYTE POINTER
                                ;BIT 35-J=1 IF MODE J IS LEGAL FOR THIS DEVICE
                                ;BIT 18 DEVICE ASSIGNED BY CONSOLE COMMAND
                                ;BIT 19 DEVICE ASSIGNED BY PROGRAM (INIT)
          XP DEVLOG,5;          LOGICAL NAME FOR JOB DEVICE
          XP DEVBUF,6;          C(LH)=OUTPUT BUFFER AREA DDRESS
,                               C(RH)=INPUT BUFFER AREA ADDRESS (JBFADR)
          XP DEVIAD,7;          C(LH)=PROG IN INDEX FIELD
          XP DEVADR,DEVIAD
,                               C(RH)=CURRENT INPUT BUFFER ADDRESS
          XP DEVOAD,10;         C(LH)=PROG IN INDEX FIELD
          XP DEVPTR,DEVOAD
,                               C(RH)=CURRENT OUTPUT BUFFER ADDRESS
          XP DEVCTR,11;
,FOR DEVICE CHARACTERISTIC WORD
          XP IORET,20;          FOR RETRIEVABLE DEVICES.  SHORT DISPATCH
,                                   TABLE=0, LONG=1.
,FOR DEVICE MODE WORD
          XP ASSCON,400000      ;ASSIGNED BY CONSOLE COMMAND ASSIGN
          XP ASSPRG,200000      ;ASSIGNED BY PROGRAM (INIT)
          XP ATTBIT,400000;     LH   TTY ATTACHED TO DEVICE DATA BLOCK
```

```
, I/Ø STATUS WORD ASSIGNMENTS
,DATA MODES:  BITS 32-35
          XP A,Ø;              ASCII
          XP AL,1;             ASCII LINE
          XP AS,2;             ASCII SEQUENCED
          XP ASL,3;            ASCII SEQUENCED LINE
          XP I,1Ø;             IMAGE
          XP B,14;             BINARY
          XP AM,5;             ALTERNATE MODE BREAK (TTY)
          XP AC,6;             CONTROL CHARACTER BREAK (TTY)
          XP DR,16  ;DUMP IN READ CONTROL MODE
          XP D,17;             DUMP
, STATUS BITS
,RIGHT HALF (USER)
          XP IOWC,2Ø;          DON'T COMPUTE WORD COUNT
          XP IOCON,4Ø;         CONTINUOUS (CONT=Ø)
          XP IORDEL,1ØØ;       READ AND DELETE
          XP IONRCK,1ØØ;       READ WITH NO. REREAD CHECK
          XP IODEND,2ØØØØ;     DATA END ENCOUNTERED
          XP IODERR,2ØØØØØ;    DEVICE ERROR
          XP IODTER,1ØØØØØ;    DATA ERROR
          XP IOIMPM,4ØØØØØ;    IMPROPER MODE REQUESTED
          XP IOBKTL,4ØØØØ;     BLOCK TOO LARGE
          XP IOACT,1ØØØØ;      DEVICE ACTIVE
, LEFT HALF (SYSTEM)
          XP IOW,1;            I/O WAIT
          XP IOBEG,2;          VIRGIN DEVICE
          XP IODISC,4ØØØØØ;    DISCONNECT REQUEST
          XP IOFST,4;          NEXT ITEM WILL BE THE FIRST ITEM OF A BUFFER
          XP IOSTRT,1Ø;        IO READY TO START
          XP IO,2Ø;            OUT=1, IN=Ø
          XP IOEND,4Ø;         SERVICE ROUTINE HAS TRANSMITTED LAST DATA


;LEFT HALF USRJDA (JOB DEVICE ASSIGNMENTS) UUO'S FOR THIS CHANNEL SINCE LAST INI
          XP INITB,4ØØØØØ;     INIT
          XP IBUFB,2ØØØØØ;     INIT WITH INPUT BUFFER SPECIFIED
          XP OBUFB,1ØØØØØ;     INIT WITH OUTPUT BUFFER SPECIFIED
          XP LOOKB,4ØØØØ;      LOOKUP
          XP ENTRB,2ØØØØ;      ENTER
          XP INPB,1ØØØØ;       INPUT
          XP OUTPB,4ØØØ;       OUTPUT
          XP CLOSB,2ØØØ;       CLOSE
                                        ;RELEASE CLEARS THEM ALL
```

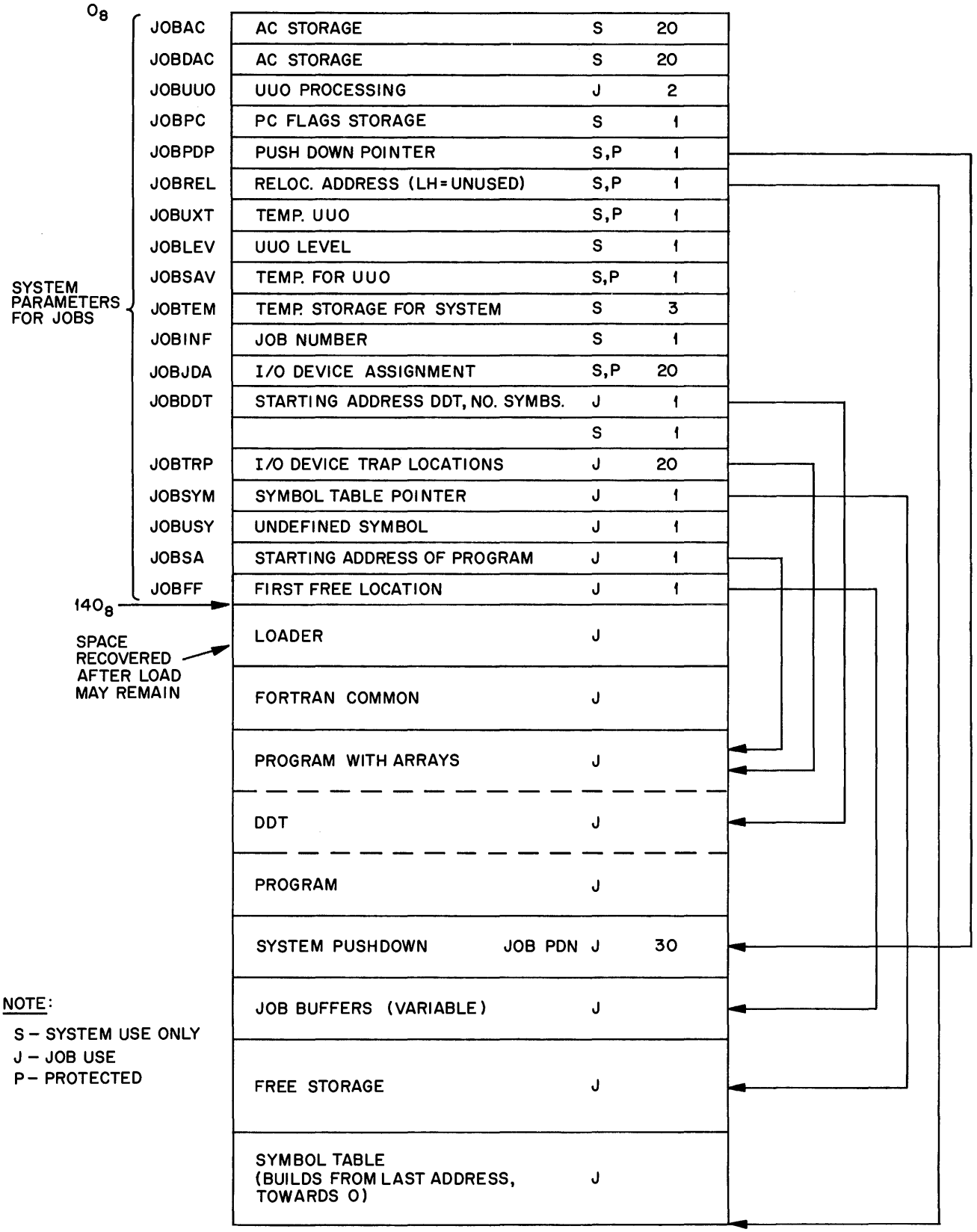
```
, JOB BUFFER AREA HEADER
          XP JBFADR,0;          BIT 0=1 IF THIS BUFFER RING HAS NEVER BEEN
,                                   REFERENCED FROM THE USER'S PROGRAM BY
,                                   AN INPUT OR OUTPUT COMMAND.
,                               BITS 1-17=UNUSED
,                               BITS 18-35=CURRENT BUFFER ADDRESS
          XP JBFPTR,1;          BYTE POINTER TO NEXT BYTE -1
          XP JBFCTR,2;          POSITIVE ITEM COUNT
, JOB BUFFER HEADER             BIT 0=IOUSE
          XP IOUSE,400000;          1 IF BUFFER IS FULL (OR BEING EMPTIED)
,                                   0 IF BUFFER IS EMPTY (OR BEING FILLED)
,                               BITS 1-17=BUFFER SIZE
,                               BITS 18-35=NEXT BUFFER ADDRESS
```

```
,SYSTEM PARAMETERS
          XP USRMOD,10000;     USER MODE BIT IN ARP FLAGS, PC WORD
          XP IODONE,400000;
,JOB STATUS WORD (JBTSTS)
          XP IOWS,400000;      IO WAIT SATISFIED
          XP RUN,200000;       READY TO RUN OR IS RUNNING
          XP JIOW,100000;      JOB IN IO-WAIT STATE
          XP DTW,40000;        DECTAPE CONTROL WAIT
          XP DCW,20000;        DATA CONTROL WAIT
          XP MTW,10000;        MAG TAPE CONTROL WAIT
          XP JNA,4000;         JOB NUMBER ASSIGNED
          XP JERR,2000;        JOB ERROR(ILLEGAL MEM, ETC.)

          XP RUNABLE,RUN+JNA; STATUS BIT PATTERN FOR A JOB TO STAY RUNABLE
```

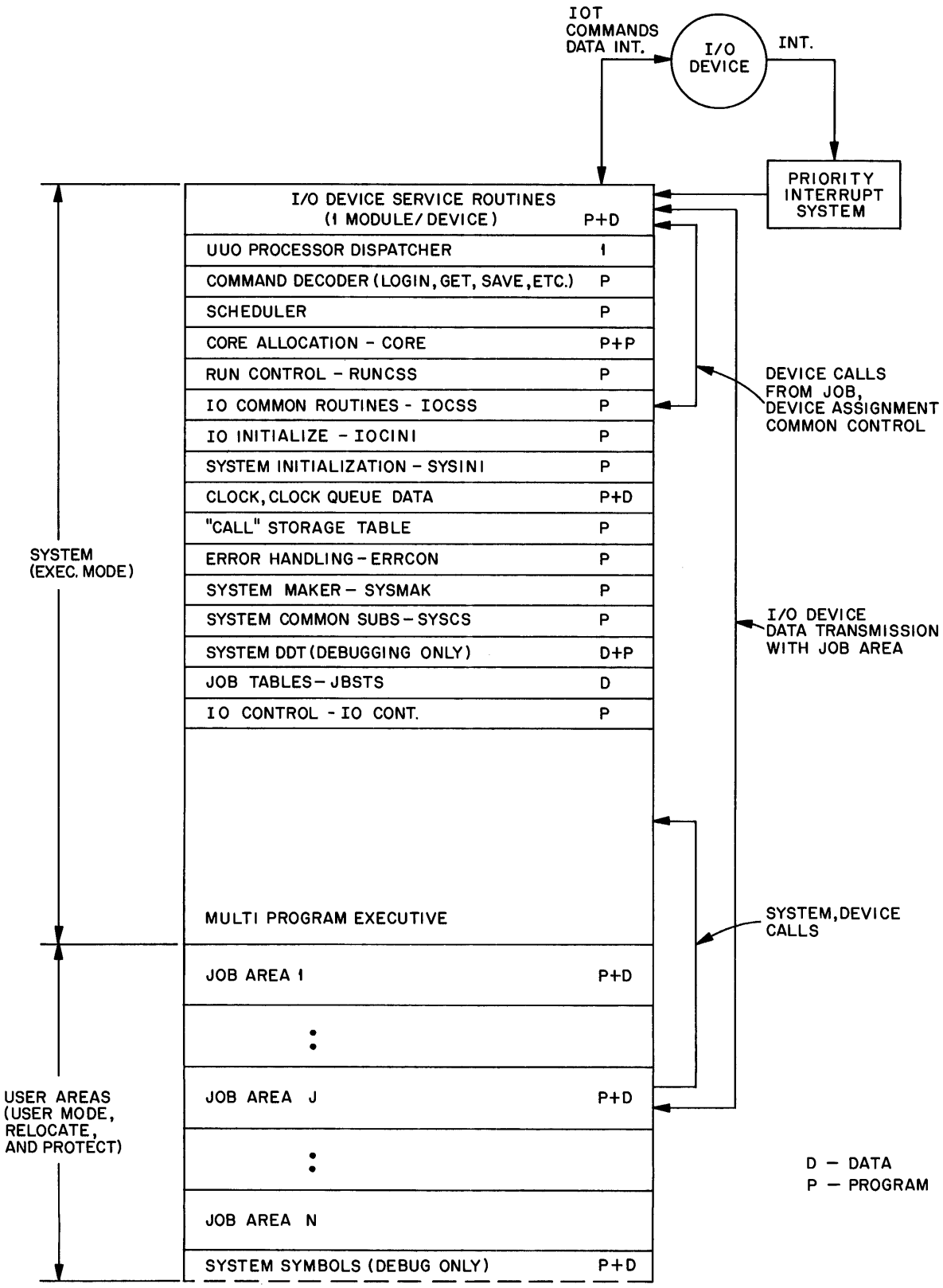
```
, SYSTEM MACROS
DEFINE     ADRCHK (A)
<          EXTERNAL ADRCK
DEFINE     ADRCHK (B)
            <PUSHJ PDP,ADRCK
            HRRZ TAC,B>
            PUSHJ PDP,ADRCK
            HRRZ TAC,A>

           LIST
```



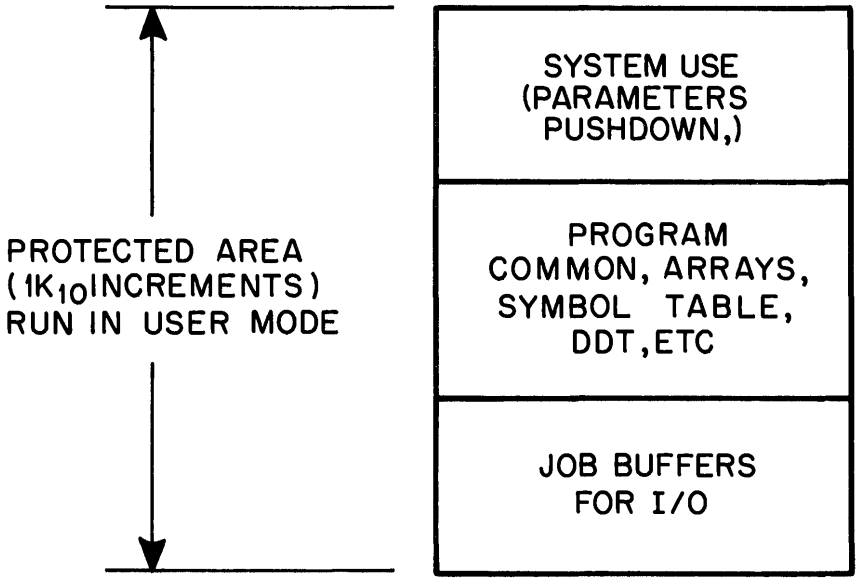
MULTIPROGRAMMING SYSTEM DIAGRAM



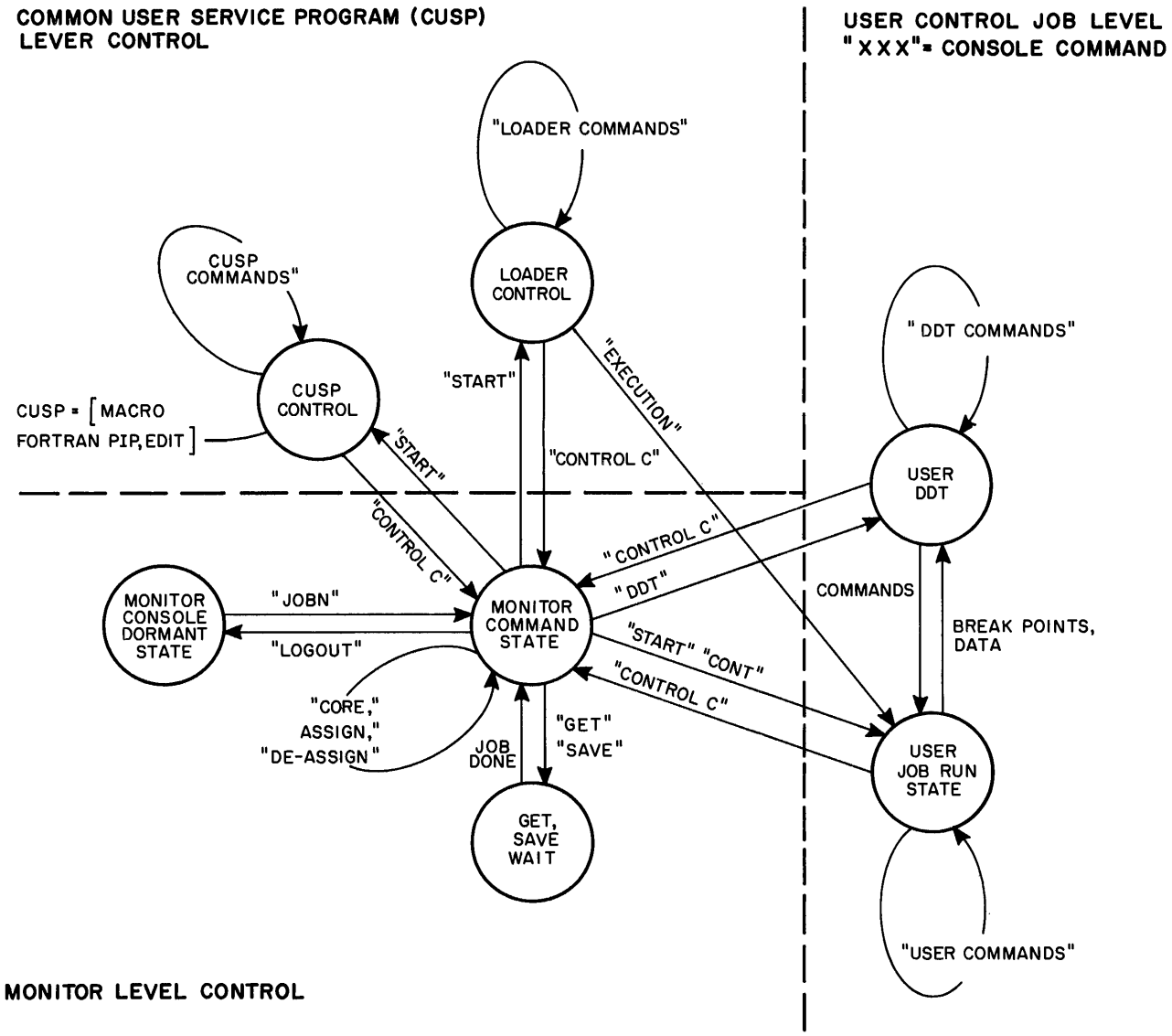
JOB AREA STORAGE



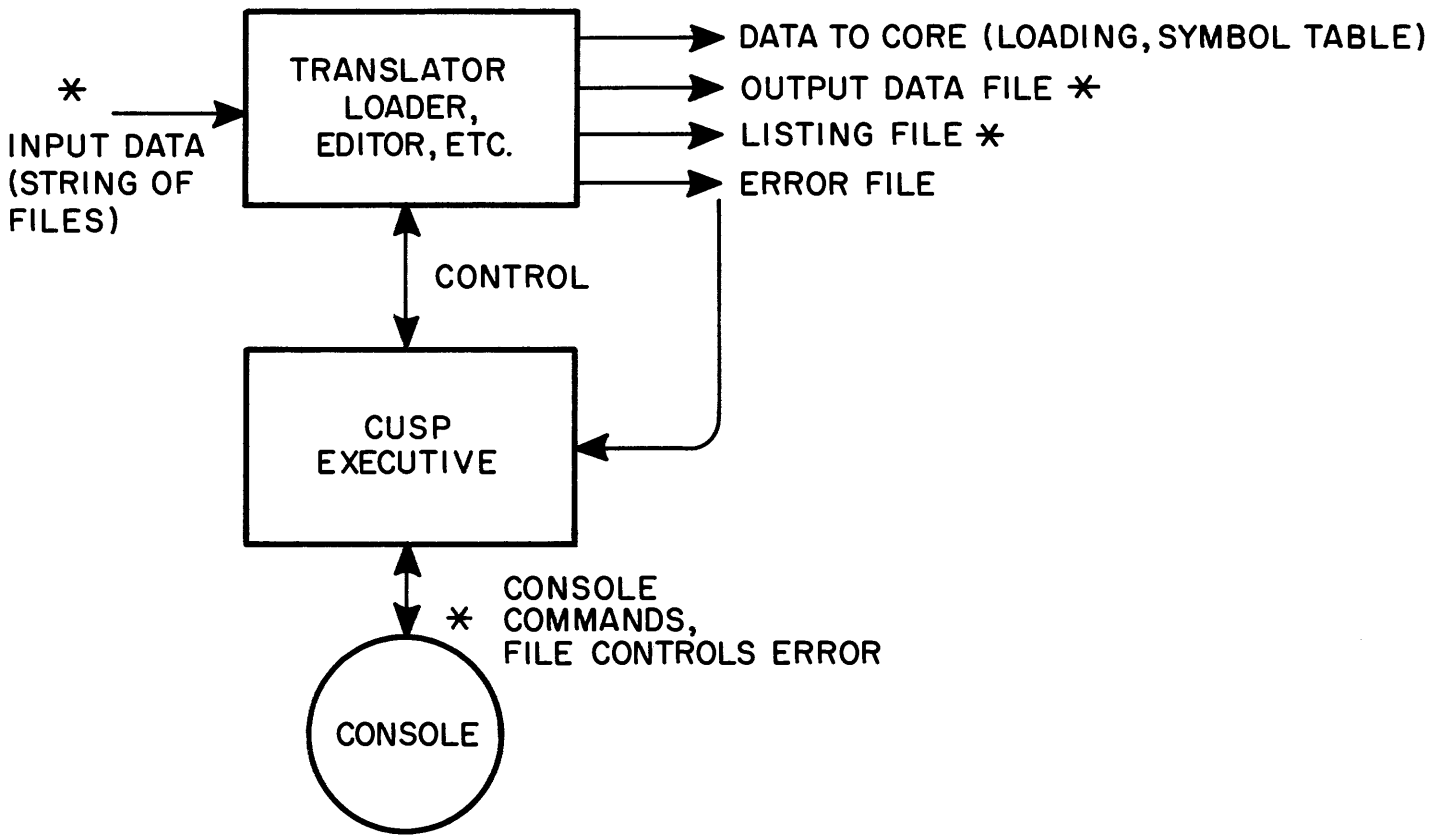
MULTIPROGRAMMING SYSTEM STORAGE



JOB AREA STORAGE (SIMPLIFIED)

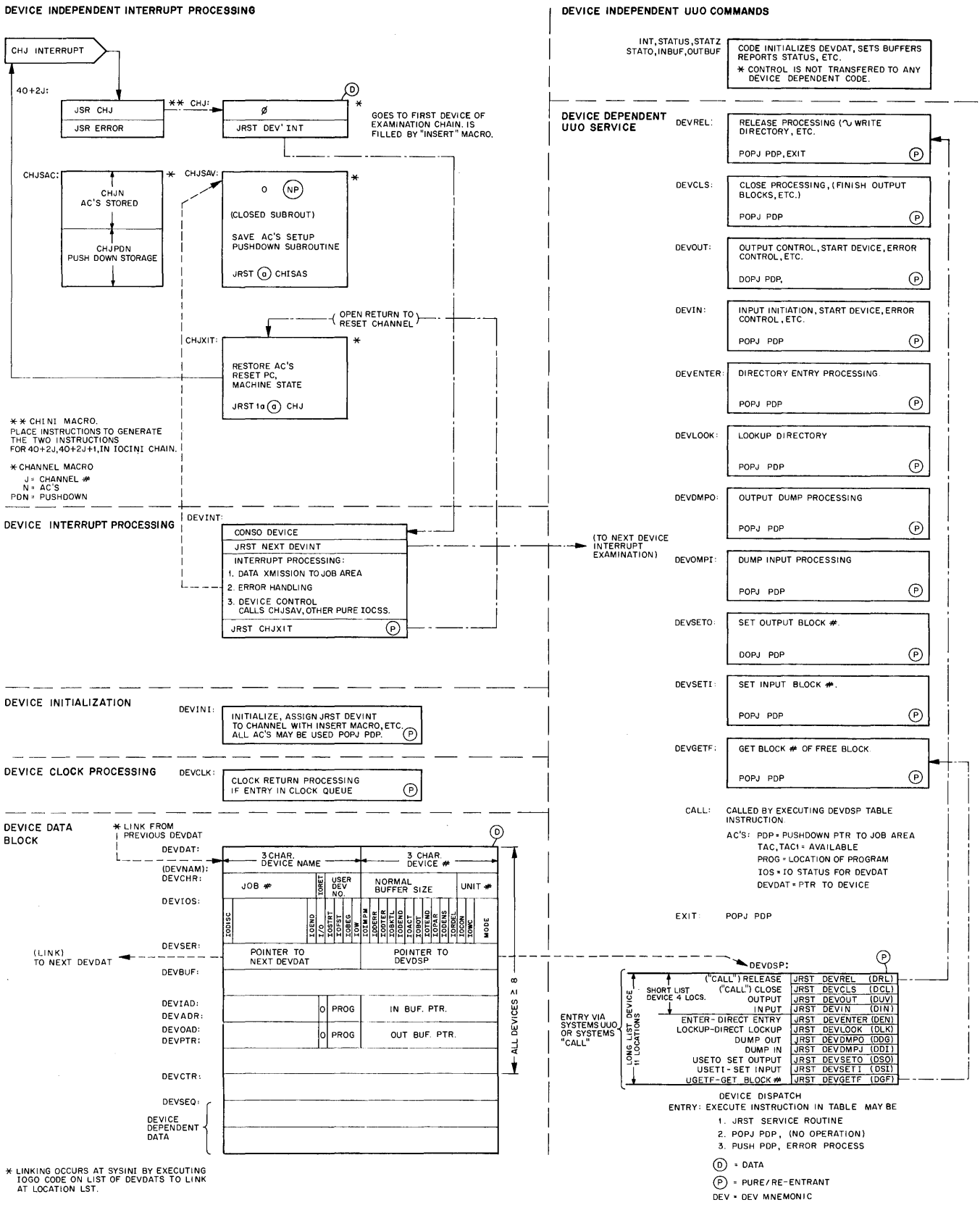


CONSOLE STATE DIAGRAM

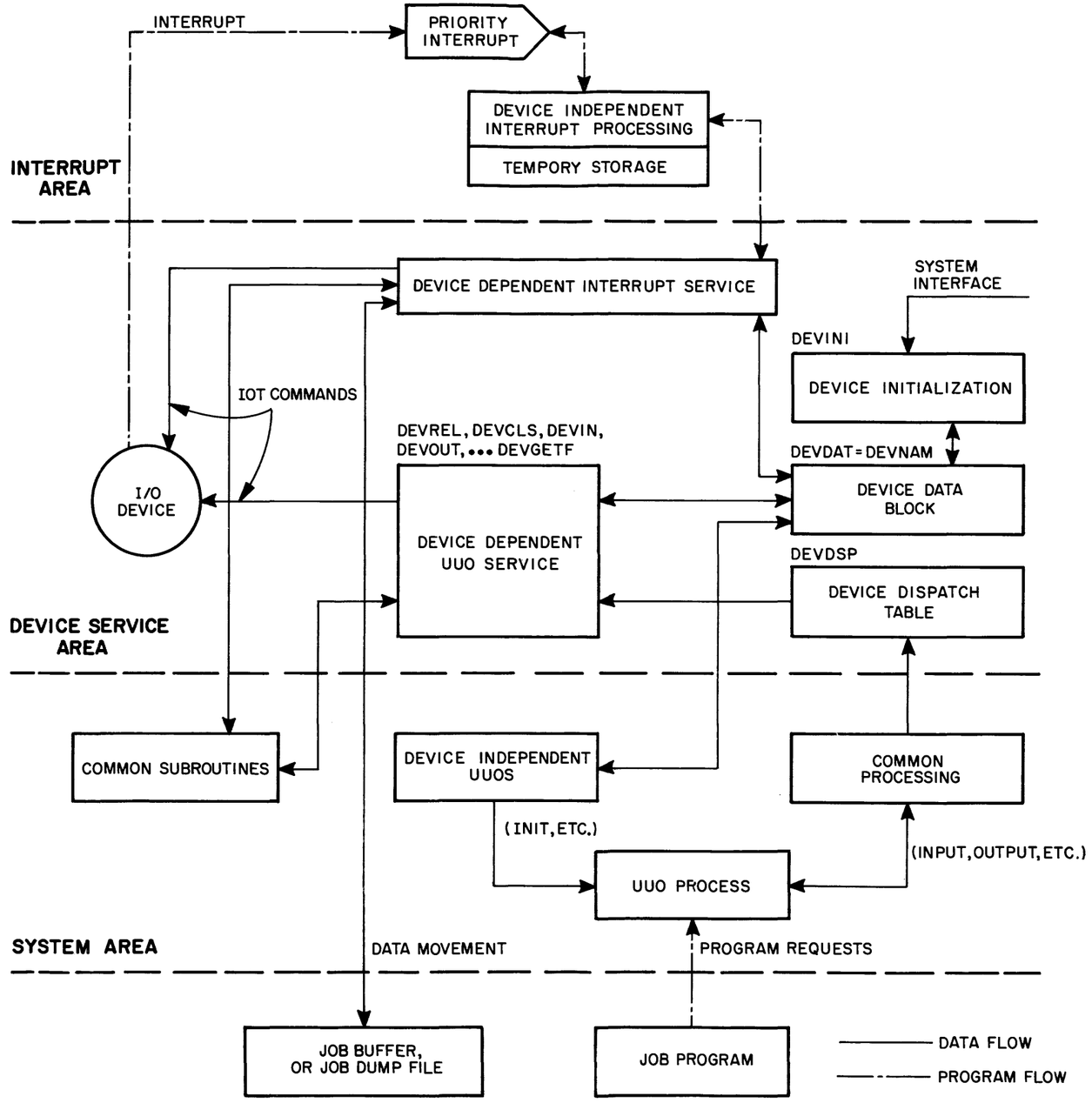


* I/O DEVICE CHANNELS

GENERAL STRUCTURE OF COMMON USER SERVICE PROGRAM - CUSP



BLOCK DIAGRAM OF I/O SERVICE ROUTINES



I/O SERVICE ROUTINE BLOCK DIAGRAM SIMPLIFIELD

SUMMARY OF COMMON USER SERVICE PROGRAMS

PROGRAM	SIZE	NUMBER OF DEST. FILES	SWITCHES	COMMANDS	ERROR MESSAGES
EDITOR	1K	EDITOR Operates Upon One File (Sequenced.)	There are No Switches In The EDITOR. Special Delimitors: CR(CARRIAGE RETURN)=Ends a Command (POINT)=Has the Value of the Last line Number Typed. ALT MODE=Special Delineator For S; When Used Alone Advances and Prints the Line.	Sx Control A (CR)=Select Unit X and Clear the Directory Sx, name Control A ALT MODE=Same as Above But Creates File "NAME" Sx, Name (CR)=Open File "NAME" On Unit X Sx, Name ALT MODE=Create a New File Called "NAME" on Unit X E(CR)=CLOSE THE CURRENT FILE In1(CR)=Insert at Line n1 In1,I2(CR)=As Above, But Set Indexing Increment to I2 Dn1(CR)=Delete Line n1 Dn1,n2(CR)=Dlete Line n1 thru n2 Pn1(CR)=Print Line n1 Pn1,n2(CR)=Print Lines n1 thru n2 Rx(CR)=Resequence The Current File Rx,y(CR)=Resequence, Stepping By y ALT MODE=Return From Text Mode To Command Mode, or If In Command Mode, Prints the Next Line	"*ILC*" ILLEGAL COMMAND "*NLN*" AP or D REFERENCED A NONEXISTENT LINE "*ILS*" The Index Step(In Insert) Has Caused a Line Already in The File To Be Skipped. "*ILR*" The Index Step Has Caused An Already Existing Line to be Referenced. "*UNA*" DecTape Unit is not Available. "*DCE*" DecTape Directory Full "*NFD*" No File Has Been Selected "*FAU*" File Name Already In Use "*NCF*" Not a Current File
PIP1	1K	1	A=ASCII Line N=Delete Sequence Numbers S=Sequence W=Rewind Mag. Tape Z=Create Fresh Directory B=Binary	General Form: Destination←Source 1, Source 2,.... DTA1:Big Fil←PTR:,, (Copy 3 Paper Tape Files onto DecTape) DTA1:Big Fil/S←PTR:,, (Same as Above But Add Sequence No.) LPT:←DTA1:Big Fil (List The File) MTA1/W←(Rewind Mag. Tape) DTA2:/Z←(Create a Fresh Directory)	"DEVICE name NOT AVAILABLE" "DIRECTORY FULL" "NO FILE NAMED file name" "OUTPUT DEVICE name DEVICE ERROR" "OUTPUT DEVICE name DATA ERROR" "OUTPUT DEVICE name IMPROPER MODE" "OUTPUT DEVICE name BLOCK TOO LARGE" "DECTAPE INPUT REQUIRED" "USE 2K PIP" "INPUT FILE file name DEVICE ERROR" ETC.
PIP2	2K(MINIMUM) (Will use All Available Storage)	1	A=ASCII Line B=Binary C=Card Input Surpress Trailing Spaces Imbedded Multiple Spaces To Tabs D=Delete A File L=List Directory Of Source N=Delete Sequence Numbers R=Rename Source File With Name of Dest. S=Add New Sequence Numbers W=Rewind Mag. Tape X=Copy and Clean Up DecTape Z=Create A Fresh Directory T=Same As C, But Without Tabs	General Form: Same as PIP1,but in Addition: DTA2:/X←DTA1: (Copy and Clean Up) DTA2:After/R←Before (Rename A File) DTA2:/D←File 1, File 2 (Delete Two Files) LPT:/L←DTA2: (List The Directory)	Same As PIP1

SUMMARY OF COMMON USER SERVICE PROGRAMS

PROGRAM	SIZE	NUMBER OF DEST. FILES	SWITCHES	COMMANDS	ERROR MESSAGES
FORTRAN	11K or 23K	2	C=Card Image Input (Assumed) T=Teletype or Paper Tape Input R=Read in Rules (Paper Tape Image Assumed)	(1 Pass Compiler) Binary, Listing←Source Ex: DTA3:BINFIL,LPT:←DTA2:FORTIN ←PTR:(T) (No Output,Errors Only) (2 Pass Compiler) Fola Input,Listing←Source Ex: DTA2:FOLAIN,DTA1: LIST←DTA5:FORTIN (Card Image) DTA3:FOLAIN←PTR: (No Listing, Paper Tape Input) DTA3:FOLAIN←DTA1:PIMAGE(T)(Teletype Image)	"FILE name NOT IN DIRECTORY" "DEVICE name DIRECTORY FULL" "DEVICE name UNAVAILABLE"
FOLA	3K	2	NONE	General Form: Binary,Listing←Source 1, Source 2,.... PTP:, LPT:←PTR: DTA2: BINARY, DTA3: LIST←DTA1: FOLAIN ←DTA3: FOLAIN (No Output, Errors Only) PTP:←PTR: (Binary Onto Paper Tape, No Listing) ,LPT:←PTR: (No Binary, Listing on Printer)	"COMMAND ERROR" "DEVICE name NOT AVAILABLE" "CANNOT FIND FILE name" "CANNOT ENTER FILE name"
MACRO	9K	2	S=Surpress Listing L=Reinstate Listing E=Expand Macros on Listing X=Turn Off Expansion F=Fix Symbol Table	General Form: Binary, Listing←Source 1, Source 2,.... PTP:,LPT:←PTR:,,,(4 Input Files) ←DTA3: Macin (No Output,Errors on User TTY) DTA1: Bin Out←DTA2: Macin (No Listing Errors on TTY) ,LPT:←MTA1: Assemb (No Binary) DTA2: Bin Out, LPT:←(S)DTA6:MAC1,(L)PTR:, (No Listing of File MAC1, Two Files from PTR)	Same Messages As FOLA (See Above)
LOADER	1K	1	S=Load Local Symbols W=Do Not Load Local Symbols L=Enter Library Search Mode N=Leave Library Search Mode A=Ignore All Subsequent Starting Addresses M=Print Storage Map R=Restart Loading I=Set Loader To Current State ALT MOD KEY=Begin Execution '(SINGLE QUOTE)=Enter Library Search for Previous File	General Form: Memory Map←(Switches) Source 1,.... LPT:←DTA3:BIN1(S), PTR:,DTA4:USRDDT ←(S)PTR: (Map On User TTY)	"ILLEGAL FORMAT" "DATA ERROR" "CORE EXCEEDED" "ILLEGAL COMMON" "COMMAND ERROR"

INSTRUCTION MANUAL

TYPE 164
MAGNETIC CORE
MEMORY SYSTEM

PDP-6

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

DEC-06-H1BA-D

TYPE 164 MAGNETIC CORE MEMORY SYSTEM

INSTRUCTION MANUAL

December 1967

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

CONTENTS

CHAPTER 1
INTRODUCTION AND DESCRIPTION

CHAPTER 2
INSTALLATION AND INDICATORS

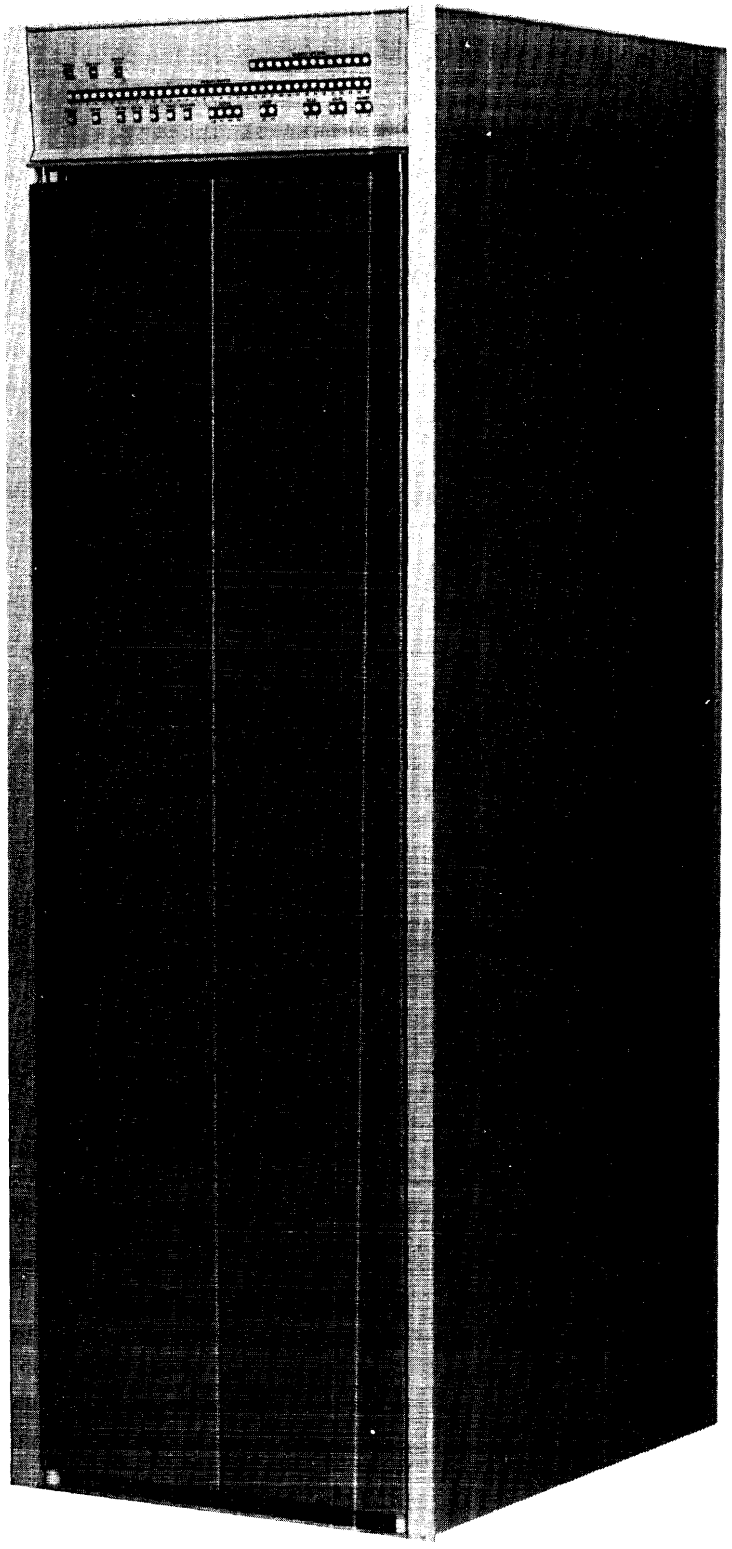
CHAPTER 3
SYSTEM DESCRIPTION

ILLUSTRATIONS

ILLUSTRATIONS (cont)

TABLES

TYPE 164 MAGNETIC CORE MEMORY SYSTEM



Type 164 Magnetic Core Memory System

CHAPTER 1
INTRODUCTION AND DESCRIPTION

Type 164 Magnetic Core Memory System (Frontispiece), manufactured by Digital Equipment Corporation, stores standard PDP-6* 36-bit words, and can accommodate a parity bit if required. This memory performs with either a single- or multiple-processor configuration. In addition, a system configuration can incorporate several of these memories. A single memory can accommodate up to four processors and includes priority logic to control processor access. All memories for use with a single processor are connected to the processor-memory bus, and each memory can be connected through to as many as four processor-memory buses. The standard Type 164 Memory stores 16,384 words.

1.1 PURPOSE AND SCOPE

This manual provides initial instruction in the use of the Type 164 Memory System. Information suitable for maintenance of the system sufficient for use by personnel with experience in servicing similar types of equipment also is included.

This document is one of several associated with the PDP-6 computer. For complete and comprehensive coverage of areas not included in this manual, the operator should refer to the list of documents under Paragraph 1.3.

1.2 SYSTEM REQUIREMENTS AND CHARACTERISTICS

Table 1-1 lists the system requirements.

Table 1-1
System Requirements

Access time from processor	Single processor 850 ns ± 50 Multiple processors 950 ns ± 50
Total cycle time of memory	1.65 μs
Input ac voltage	105-125 at 60 c/s 220-250 at 50 c/s
Current	Nominal 11A Surge 16A
Power dissipation	995W
VA	1265

*PDP is a registered trademark of Digital Equipment Corporation

Table 1-1 (cont)
System Requirements

Heat dissipation	3396 Btu/hr
Temperature range (Operating)	70° to 85° F
Humidity range	30% to 80%

1.2.1 Access Time

Total cycle time for the core memory is 1.65 μs, but this time is not a factor in system operating speed unless calls to the same memory are so frequent that processor time is lost in waiting, in which case memories should be interleaved. The total time required for access by the processor is of importance when considering system operating speed. This time is dependent both on the speed of the memory and the time required by the memory subroutine in the processor. In Table 1-2, all times are in nanoseconds. The time in memory is the actual time consumed within the memory for read access and write access (the time from the appearance of the request signal at the memory until acknowledgement or the read restart is generated). (See Chapter 4.) The access time is the total access time for the Arithmetic Processor Type 166 and is the interval between the pulse which requests memory access subroutine and the return pulse which restarts the waiting sequence. The times are given below for a call without relocation or stopping on parity errors, with a bus length of 10 ft. For each additional foot of cable, add 3 ns; when a core address is relocated, an additional 100 ns is required; for parity checking with error stop, add 100 ns to the read-access time for core.

Table 1-2
Access Times

	Time in Memory (ns)	Type 166 (Exec Mode) Access Time (ns)
Core Memory		
Read access	Single processor 500±50 Multiple processors 600±50	Single processor 850±50 Multiple processors 950±50
Write access	Single processor 100±50 Multiple processors 200±50	Single processor 500±50 Multiple processor 550±50

NOTE: Add 150 ns to read and write access time when Type 166 Arithmetic Processor is in the user mode.

For read/write access, the time required for the first memory call is the time given for read. No time is required within the memory for the second call because it is already waiting for data and the write restart, and no response is expected of it. (See Chapter 4.) The time required for the call in the processor is the time required to send the restart and terminate the subroutine.

1.2.2 Physical Description

The Type 164 Memory System is housed in a single standard 19-in. DEC cabinet. Figure 1-1 shows the major equipment layout within the cabinet, and Table 1-3 presents the major physical characteristics. Inside the doors at the rear of the rack is an inner plenum door on which are mounted the required power supplies and power control panels.

Table 1-3
Physical Characteristics

Dimensions	
Width:	22-1/4 in.
Depth:	27-1/16 in.
Height:	69-1/8 in.
Service Clearance	
Front:	8-3/4 in.
Rear:	14-7/8 in.
Weight	
	560 lb

The figures listed in Table 1-3 are for memories that stand alone. Whenever a memory is added to a frame containing either a processor or another memory, the width is decreased by 2 in. Clearance required for the double doors is 9 in., and for the rear plenum doors, 15 in.

Intake fans at the middle protion of the rack cool the top and bottom logic modules by blowing filtered air between them. Additional fans mounted on the plenum door cool ambient temperature of the bay.

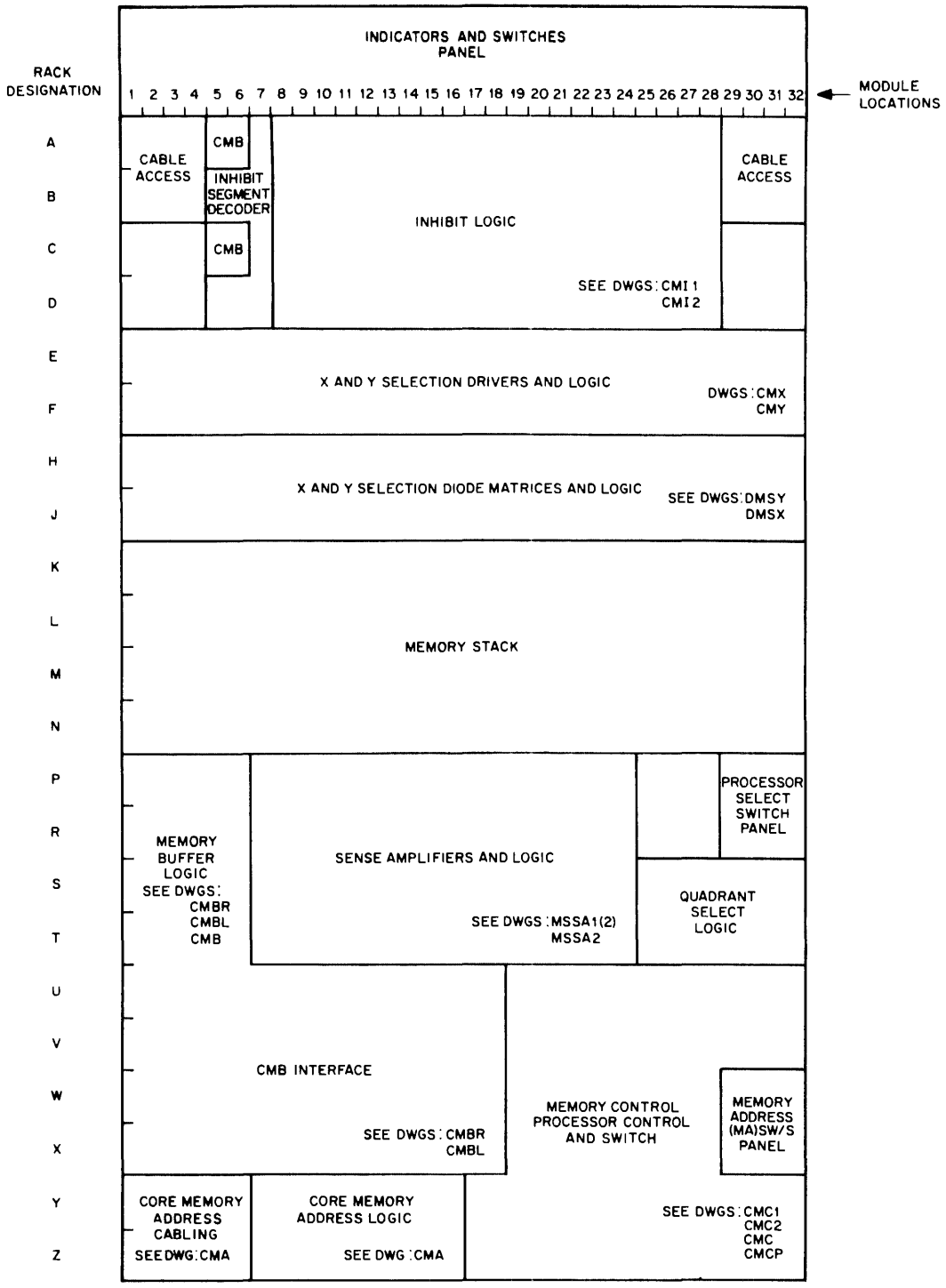


Figure 1-1 Type 164 Memory System, Equipment Layout

1.3 REFERENCE DOCUMENTS

The listing of Table 1-4 itemizes the standard documentation provided for use with the PDP-6 system. Other documents may be furnished if they are applicable to customer requirements.

Table 1-4
Hardware and Software Documentation

Central Processor Type 166 Maintenance Manual, F67	Instruction and servicing information.
PDP-6 Users Handbook, F65	Programmers manual for PDP-6 system.
Module Catalog, C105	Complete data concerning modules and module circuits used in the PDP-6 system.
PDP-6 Program Library	Detailed description, listings and tapes.
PDP-6 Software Documents	Includes software system manuals such as FORTRAN, DDT debugging, etc.

1.4 REFERENCE CONVENTIONS

The Digital Equipment Corporation engineering drawing conventions and instruction manual referencing should be understood at this point. A study of the reference conventions in this paragraph and Chapter 6 will save considerable time and preserve thought continuity while reading the text that follows. Any reference to figure numbers or table numbers indicates that the illustration or table is located in text. For example, Figure 3-1 is located Chapter 3 and is the first illustration in that Chapter.

All engineering drawings have a full drawing number. These drawings are included in Chapter 6. In text, references to engineering drawings are abbreviated. For example:

Full drawing no. D-164-0-SBD

- D — Drawing original size
- 164-0 — Drawing set identifier and no. in set
- SBD — Mnemonic identifier

For this number, the first reference in text is 164-0-SBD; all subsequent references are -SBD.

To locate a specific signal or function, the origin of the signal on a specific drawing is stated in one of two conventions.

1. "The OGN pulse developed at PA (D23, SBD)..., where PA is the module type."
   D23 is the physical location of the module by rack and module slot, and
   SBD is the specific drawing on which the reference can be found.
2. "The CLR pulse is generated at module PA (SBD:C4)..., where PA is module type."
   SBD is the drawing, and
   C4 is the coordinate location on the drawing for the module.

CHAPTER 2
INSTALLATION AND INDICATORS

2.1 SYSTEM CABLING

2.1.1 Power Cabling

The standard power cables supplied with the 164 Memory System are 25 ft long and 11/16 in. in diameter and permanently wired to the unit. These cables have a Hubbel Twist-loc plug number 34-3331, rated at 115V, 30A. Thus, the primary power source must have a 115V, 30A Hubbel Twist-loc flush receptable part number 34-3333 or equivalent.

2.1.2 Bus Cable Requirements

The central processor and memory system are interconnected by eight 9-conductor coaxial cables. Four of these have two DEC 10350 female connectors (two 9-conductor cables per 10350) on one end, and four DEC W028 connectors on the other as shown on Dwg. D-164-0-CCD, sheet 1. The other four cables have DEC 10360 male connectors at one end and four DEC W028 connectors on the other end of each cable, as shown on Dwg. D-164-0-CCD, sheet 2.

The signals transmitted over these cables are listed in Table 2-1.

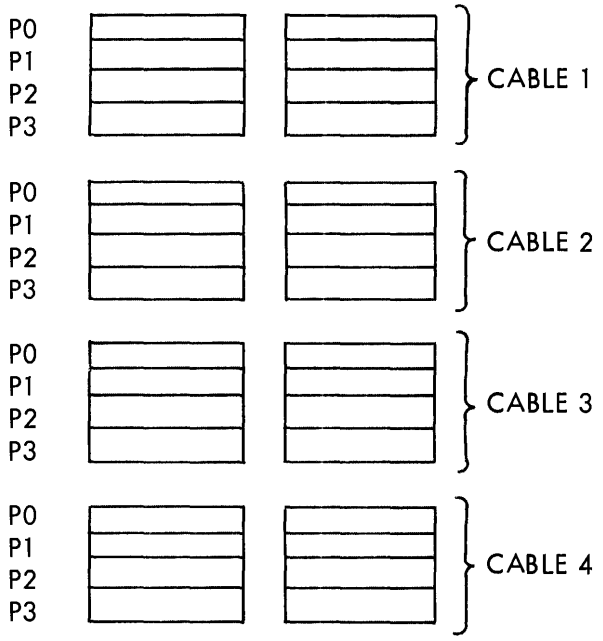
Table 2-1
PDP-6 Memory Bus Interface

PDP-6 PROCESSOR END					164 Memory End	
Pin	DEC 10360 Memory Cable No. 1	DEC 10360 Memory Cable No. 2	DEC 10350 Memory Cable No. 3	DEC 10350 Memory Cable No. 4	DEC * W028 Pin No.	
A	Gnd	Gnd	Gnd	Gnd		
B	ADDR ACK →	MA 22 (1) ◆→	MB 0 (1) →	MB 18 (1) →	D	
C	RD RS →	MA 23 (1) ◆→	MB 1 (1) →	MB 19 (1) →	E	
D	WR RS →	MA 24 (1) ◆→	MB 2 (1) →	MB 20 (1) →	H	
E	PAR (1) →	MA 25 (1) ◆→	MB 3 (1) →	MB 21 (1) →	K	First
F	RQ CYCLE ◆→	MA 26 (1) ◆→	MB 4 (1) →	MB 22 (1) →	M	W028
H	Spare	MA 27 (1) ◆→	MB 5 (1) →	MB 23 (1) →	P	Connector
J	Gnd	Gnd	Gnd	Gnd		
K	MA 18 (1) ◇→	MA 28 (1) ◆→	MB 6 (1) →	MB 24 (1) →	S	
L	MA 18 (0) ◇→	MA 29 (1) ◆→	MB 7 (1) →	MB 25 (1) →	T	
M	MA 19 (1) ◇→	MA 30 (1) ◆→	MB 8 (1) →	MB 26 (1) →	V	

Table 2-1 (cont)
PDP-6 Memory Bus Interface

PDP-6 PROCESSOR END					164 Memory End	
Pin	DEC 10360 Memory Cable No. 1	DEC 10360 Memory Cable No. 2	DEC 10350 Memory Cable No. 3	DEC 10350 Memory Cable No. 4	DEC * W028 Pin No.	
N	MA 19 (0) ◇—	MA 31 (1) ◆—	MB 9 (1) →	MB 27 (1) →	D	
P	MA 20 (1) ◇—	MA 32 (1) ◆—	MB 10 (1) →	MB 28 (1) →	E	
R	MA 20 (0) ◇—	MA 33 (1) ◆—	MB 11 (1) →	MB 29 (1) →	H	
S	Gnd	Gnd	Gnd	Gnd		Second W028 Connector
T	MA 21 (1) ◇—	MA 34 (1) ◆—	MB 12 (1) →	MB 30 (1) →	K	
U	MA 21 (0)	MA 35 (1) ◆—	MB 13 (1) →	MB 31 (1) →	M	
V	MA 22 (1)	MC RD RQ ◆—	MB 14 (1) →	MB 32 (1) →	P	
W	MA 22 (0)	MC WR RQ ◆—	MB 15 (1) →	MB 33 (1) →	S	
X	MA 22-	Spare	MB 16 (1) →	MB 34 (1) →	T	
Y	31=0 ◇—	Spare	MB 17 (1) →	MB 35 (1) →	V	
Z	Gnd	Gnd	Gnd	Gnd		

(SOURCE 2L5) (SOURCE 2L20) (SOURCE 2E25) (SOURCE 2J25)



*NOTE: Each DEC 10350 and DEC 10360 connector and cable is terminated to two DEC W028 connectors. (Refer to BUS CABLE REQUIREMENTS Paragraph 2.1.2.)

2.1.3 Inter-Memory Cabling

Interconnection between Type 164 Memory to another Type 164 Memory is accomplished by eight 9-conductor coaxial cables with DEC W028 connectors on each end.

2.1.4 Margin Check and Remote Power Cable

Each system is supplied with a margin check and remote power turn-on cable which plugs into the power connector bracket between the Type 164 Memory and processors. (See Dwg. IA-B-7405423-0-0).

2.1.5 Termination Requirements

Each 164 Memory System on the end of a bus should be supplied with eight DEC W028 connectors with handles. These should be ordered separately. Each W028 will have nine 100 ohms, 1/4W, 5% or 10% wired-in terminators to ground. (Refer to Dwg. IA-B-5403930-0-0.)

2.1.6 Pressurized Cabinets

A 164 Memory System which is to be connected mechanically to a unit that contains system type modules, must have a side panel installed so that pressurization can be maintained in the unit with system modules. Cable access through the side panel is 8 in. x 4 in. These side panels are not necessary when two Type 164 Memories are connected together. These panels should be ordered separately (Part No. 74-05415 and Dwg. D-MD-7405415-0-0).

2.1.7 Cable Access

If the Type 164 Memory is remotely located, cable access is through the bottom of the cabinet.

If the Type 164 Memory is connected mechanically to another Type 164 unit or non-system type module unit, cable access is through the side or bottom of the cabinet.

If the Type 164 Memory is connected mechanically to a system type module unit, cable access is through the side panel as explained under pressurized cabinets.

2.1.8 Cooling

All modules will have styrofoam strips 1/8 in. x 2-1/4 in. x 16 in. between handles. These must remain in place to insure that modules are not damaged by excessive heat.

2.1.9 Multiprocessor Input Module Requirement

10 each - W102 per additional processor input

1 each - B684 per additional processor input

8 each - W028 (cable terminator) per additional processor input if Type 164 Memory unit is at the end of the bus.

2.2 INDICATOR PANEL AND SWITCHES

The indicator panel (Figure 2-1) contains the POWER, RESTART, and SINGLE STEP switches and register flip-flop indicator lights that illustrate system conditions for memory address and memory buffer plus control conditions.

A processor selection switch and restart switch are mounted on the rack as shown in Figure 2-2. Table 2-2 lists the system switches and functions.

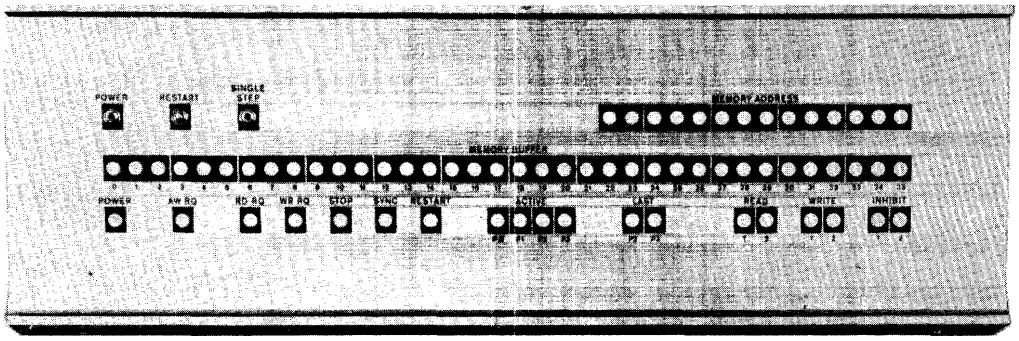


Figure 2-1 The Indicator Panel

Table 2-2
System Controls

Switch	Location	Function
POWER	Indicator Panel	Primary power for memory
RESTART	Indicator Panel	To restart memory cycle manually
SINGLE STEP	Indicator Panel	Allows one-step operation for maintenance under control of RESTART switch
PROCESSOR SELECTION	S6 Front of Cabinet	Manual selection of a single processor or multi-processor operation
RESTART	S5 Front of Cabinet	Manual restart

Table 2-2 (cont)
System Controls

Switch	Location	Function
SEL/DES PO	S23 Front of Cabinet	Manual selection or deselection of individual processors
SEL/DES P1	S24 Front of Cabinet	
SEL/DES P2	S25 Front of Cabinet	
SEL/DES P3	S26 Front of Cabinet	
PO NORM/INTL	S1 Front of Cabinet	Select normal or interleaving memory operation (more than one) (Type 164 Memory)
P1 NORM/INTL	S2 Front of Cabinet	
P2 NORM/INTL	S3 Front of Cabinet	
P3 NORM/INTL	S4 Front of Cabinet	
MA 18 0-1 P0, P1, P2, P3	S7, S8, S9, S10	Select code
MA 19 0-1 P0, P1, P2, P3	S11, S12, S13, S14	
MA 20 0-1 P0, P1, P2, P3	S15, S16, S17, S18	
MA 21 0-1 P0, P1, P2, P3	S19, S20, S21, S22	

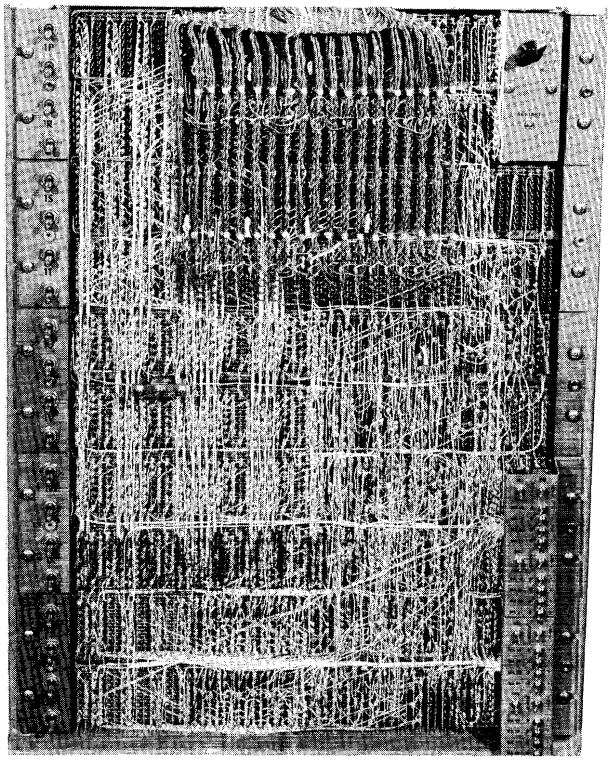


Figure 2-2 System Switches

2.3 OPERATION

A standard panel (Figure 2-1) is installed on all memories, although some of the indicators may not be used. The controls and indicators are primarily for maintenance; most of the lights change too rapidly to be significant when the computer is running normally. On the upper left of the panel are two toggles and a pushbutton. Memory power can come on only when the POWER switch is up, so this switch is usually left on at all times so that the memory comes on when power is applied to the computer system. The POWER light at the lower left turns on immediately when power is applied. If the SINGLE STEP switch is up, the memory stops with the STOP light on at the end of every cycle; it can be restarted by pressing the RESTART button.

Following a stop, the memory appears to the processor to be nonexistent when the next access is requested. Therefore, to use single-step mode at the memory, the operator should latch on the MEMORY STOP key at the processor console so that he may single-step the processor by memory subroutine-calls or turn on the DISABLE MEMORY switch so that the processor stops whenever the memory does not respond to a request for access. If a malfunction causes the memory to hang up with the STOP light off, it can be restarted by pressing RESTART while holding on the stop override button. This button is located behind the double doors on the front of the bay.

2.3.1 Core Memory Indicators

If the AW RQ light on a core memory remains on while the computer is running, the particular memory is not currently in use. One of the LAST lights may also stay on during operation. This light indicates which of processors 2 and 3 was the last to have access. The remaining indicators display useful information only when the memory has stopped (during maintenance procedures) or the computer has stopped. MEMORY ADDRESS indicates the location within the memory to which the last access was made; MEMORY BUFFER displays the word read or written, but all the lights are off following the read portion of a read-write cycle. When a memory cycle starts, the AW RQ light goes off, and one of the four ACTIVE lights goes on, indicating which processor gained access. The RD RQ and WR RQ lights indicate whether there is a read request, a write request, or both. At the end of the read portion of the memory cycle, the SYNC light goes on, and if new data is to be written, the RESTART light goes on when the processor sends the write restart. The READ, WRITE, and INHIBIT lights represent the drive currents applied to the cores, and they can be on at the completion of a cycle only if a malfunction occurs.

If the memory is stopped independently of the processor, the ACTIVE lights will be out, STOP will be on, and the remaining lights will reflect the cycle just performed. On an instruction stop, or the stop following an examine or deposit from the processor console, the AW RQ light will be on, indicating that the memory is free and awaiting a request; ACTIVE will be off; and the other lights will reflect last

cycle. On a memory stop at the processor (i.e., a stop following the completion of a memory subroutine), the lights reflect the type of stop. If the stop follows the read part of a read/write access, the memory is at the middle of its cycle, so AW RQ will be off; one of the ACTIVE lights will be on; and RESTART will not yet be on. For any other type of memory stop, the lights are the same as for an instruction stop.

The remaining controls are located behind the double doors. The operator selects normal or interleave operation by using the toggle switch mounted on a bracket and listed in Table 2-2. In normal operation, the switch is in the NC position and the memory responds to a block of 16K consecutive addresses. To interleave the locations in a pair of memories, are interleaved, the MEMORY ADDRESS lights still indicate the internal location to which access was made. Bit 35 at the memory now corresponds to bit 21 at the processor, and processor bit 35 determines memory module selection. This exchange must be considered when working with the memories from the processor console.

Power supplies are mounted on the rack plenum door. The red light on the power supply is on whenever the ac line is plugged in. Beside the light are the ac circuit berakers.

Switching S1 to LOCAL turns on memory power; with the switch in REMOTE, memory power goes on and off with system power which is controlled from the control panel POWER switch.

CHAPTER 3
SYSTEM DESCRIPTION

All core memories associated with a central processor are connected to the processor memory bus. Also, each memory can be connected to up to four memory buses (four processors). A core memory responds to a request for access over any associated processor memory bus and includes within its logic a priority network that determines which processor shall have access. In addition to standard 36-bit word storage, the core memory also accommodates a parity bit when the Type 169 Parity Option is installed.

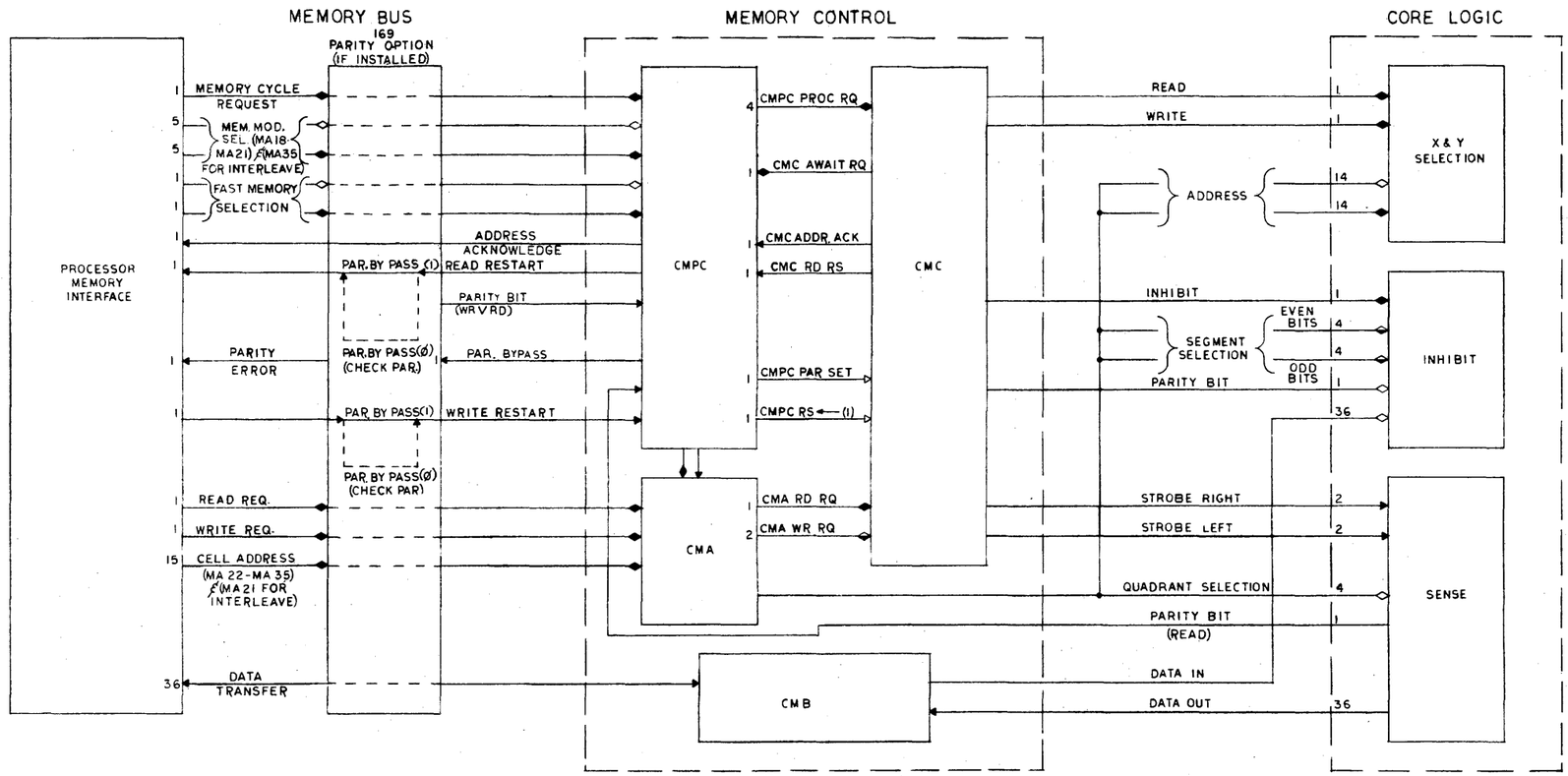
A processor may request three types of access to memory: read, in which the memory subroutine (in the processor) retrieves a word from memory; write, in which the subroutine sends a word to memory to be stored; read/pause/write, in which the processor combines two memory subroutine calls into one memory-access, the first to retrieve a word, the second to store a new one.

A single processor can select any location out of 262,144 using an 18-bit address. A 16K memory module is selected by the 4 MSBs of the address, and the remaining 14 bits select a location within that memory. The address of each memory is prewired within its interface with the memory bus.

In systems with added memory modules, interleaving memory addresses between two modules can be selected to reduce processor waiting time. The two core memories are n and n+1 where n is even. Consecutive addresses supplied by the processor switch access back and forth between modules. Interleaving is accomplished by throwing a switch that makes jumper changes in the module involved. These changes substitute address bit 35 for bit 21 in the selection of memory and substitute 21 for 35 in the address of a location within the selected memory. In the first 16K block, all even addresses address the even locations in the first memory, and all odd addresses address the even locations in the second module. In the second 16K block, all even addresses address odd locations in first module, and all odd addresses address odd locations in the second module. In this way, the processor cycles can overlap memory cycles; that is, after addressing the first module, the processor does not need to wait for completion of the memory cycle in module one and can go on to address the second module.

3.1 DISCUSSION OF SYSTEM BLOCK DIAGRAM

As shown in Figure 3-1, system block diagram, pulses transfer data between the processor(s) and memory over 36 data lines. (See Table 2-2.) With a parity option included, a pulse line carries the parity bit between the memory and the parity logic. The processor provides 14 levels (MA22-35) as the address and 2 levels as read and write request signals. Memory module selection and interleaving are selected by five level times, MA18-21 and MA35, respectively.



NOTE:
DOTTED LINES REPRESENT FUNCTIONAL SIGNALS WITHIN THE 169 PARITY OPTION

Figure 3-1 System Block Diagram

In return to processor requests, the memory control logic provides an ADDRESS ACKNOWLEDGE pulse which initiates processor functions to execute the requested access (read or write).

For read access, the processor sends a READ REQ (request), and the memory control responds by sending an ADDRESS ACKNOWLEDGE pulse. Then data and a READ RESTART trigger completion of the access routine. For write access, the processor sends the WRITE REQUEST SIGNAL. When the memory controls return ADDRESS ACKNOWLEDGE, the processor sends data and a WRITE RESTART pulse to memory control. For read/write access, the processor sends both READ and WRITE requests at the beginning of the first memory subroutine. After acknowledgment, data and READ RESTART trigger completion of the read portion of the access, then the WRITE RESTART is generated at the processor to complete the write portion of the cycle. In the memory, the cycle stops after completing the read part to await the WRITE RESTART and new data (READ/PAUSE/WRITE).

The memory control section sends the acknowledge signal and, simultaneously, loads the address and request signals from the selected bus into the address register (CMA) and read and write request flip-flops. The outputs of CMA are applied to the core logic to select the proper address in the core stack. In read access, the processor waits during the read portion of the memory cycle until the memory strobes the sense-amplifier outputs, sending data into the CMB and over the bus. For read access, data is retained in the CMB.

After returning the READ RESTART and disconnecting itself from the bus (so that the processor can use another memory), the active memory completes its cycle, writing the word in CMB back into the same location. For write access, acknowledgment causes the processor to send the WRITE RESTART and data immediately. Memory control gates the data into CMB and disconnects from the bus. The memory then performs its complete cycle, first reading to clear the location and then writing the CMB information into it. For read/write access, the memory begins by performing the same function as in read access. The processor completes its first memory subroutine upon receiving the data, but the memory does not disconnect; instead it completes only the read portion of its cycle, and then pauses with a gate open to allow data from the bus into CMB. When the processor makes the second call, it sends the data and the write restart. The memory then disconnects and continues with the write portion of its cycle, writing the new data into the cleared location.

The processor selection portion (CMPC) of memory control decodes bits 18 through 21 to determine when this memory is being addressed. For interleave operation, bit 21 is substituted for 35 at CMA and 35 for 21 in CMPC. In addition, the CMPC receives signals from S6, the processor selection switch for manual selection of up to four processors, P0, P1, P2 or P3, and for automatic operation of all four processors through the MULTI position of S6.

To select a single location within the stack, CMA provides a 14-bit address; two 7-bit halves are decoded separately to select individual X- and Y-windings.

The stack is also divided into four quadrants with four sense windings per plane (every X- and Y-winding crosses two quadrants). The most significant bits for both halves of the address (bits 22 and 29) determine which quadrant (Q0, 1, 2, 3) contains the addressed location. Sense amplifiers detect the sense windings from only the selected quadrant. Separate strobe pulses are applied to sense amplifiers for load reasons and due to the characteristics of the stack, the strobe for bits 9 through 17 and 18 through 26 is delayed 20 ns as compared to bits 0 through 8 and 27 through 35. The delaying process is achieved by the addition of two inverters.

WRITE and 14 ADDRESS bits (7 bits for X and 7 for Y) accomplish X- and Y-selection for data input to the core stack. INHIBIT and four lines of SEGMENT SELECTION provide the current for switching the cores of the selected address in accordance with the 36 data inputs.

3.1.1 Central Processor/Memory Interface

Up to four CPs can address a single memory module and/or a single CP can address up to four memory modules. Within a memory module, the CMC logic contains a priority system to control multiprocessor access to the memory. The processor sends bits 18 through 21 and ~ FM as the code for a module, and each module recognizes its own code.

In the processor, the interface that connects with the memory bus includes:

Memory address register (MA)
Memory buffer (MB)
User mode registers (PR and RLR)
Memory indicator register (MI)
Control logic for the memory subroutine.

Flow and control for the memory subroutine for the processor is shown in Dwg. FD-D-167-0-FD contained in the Type 166 Central Processor Maintenance Manual.

3.1.2 Memory Control (Prints CMPC, CMC1, CMC2 (1 of 2), CMA, CMB, CMBR)

The basic memory control scheme is shown in Figure 3-2 and consists of the major logic elements as follows.

Processor control logic (CMPC)
Memory control logic (CMC)
Memory address logic (CMA)
Memory buffer logic (CMB).

The CMPC logic receives from the processor MA bits 18, 19 and 20 which carry the code of the processor requesting access, and bits 21 and 35 for interleaving. The processor priority logic provides a processor request signal (P0RQ, P1RQ, etc.) to set the proper flip-flop in the CMC logic. These flip-flops and/or settings of a processor selection switch initiate address acknowledge and read restart signals to the processor, and control gating of memory address bits to the X- and Y-selection inhibit and sense logic.

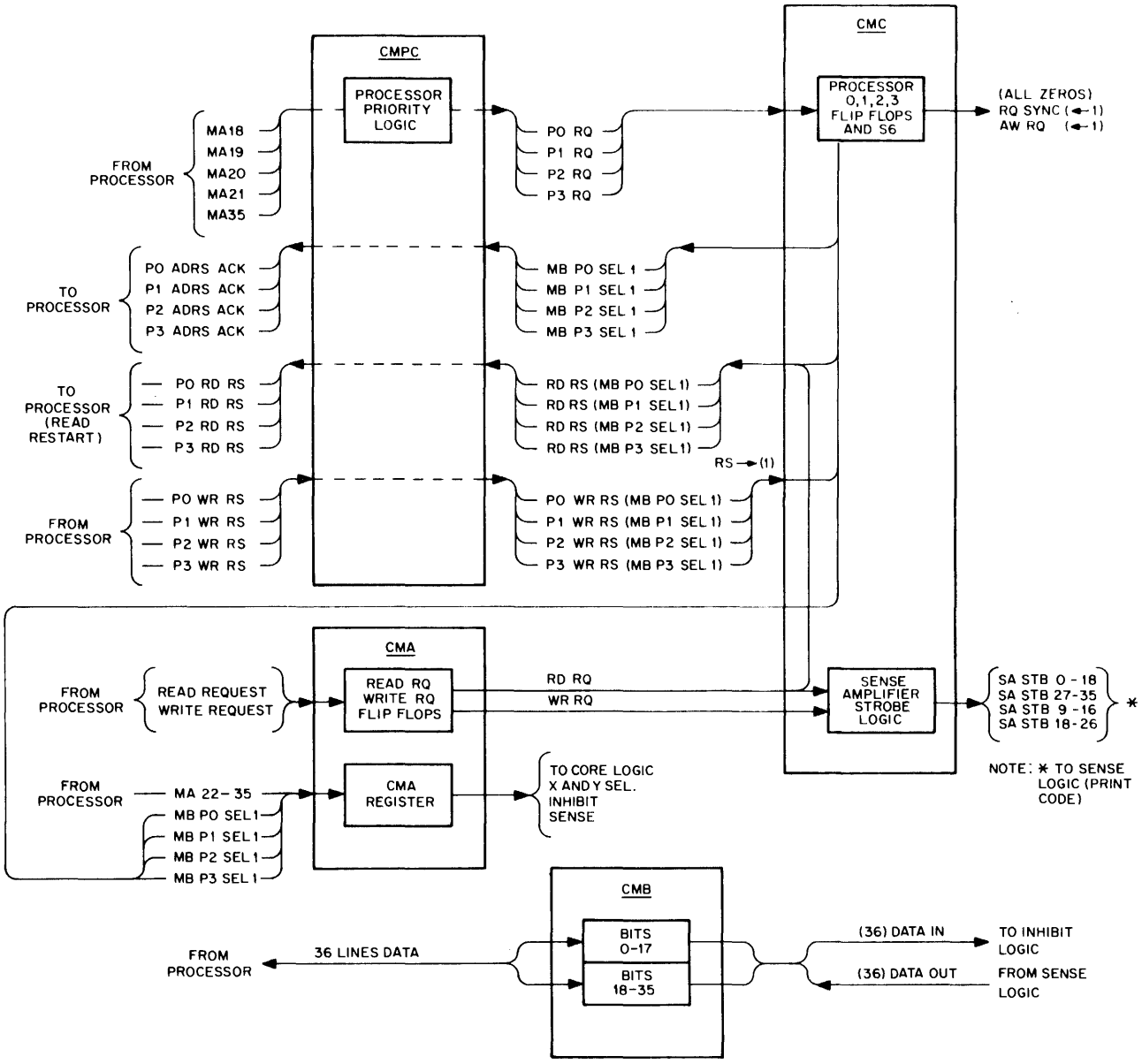


Figure 3-2 Memory Control Block Diagram

The CMA receives read and write request signals to drive the sense amplifier strobe logic in the CMA logic. The outputs of this logic provide strobe signals for all word bits 0 through 35 plus parity for the sense logic.

The CMB logic receives 36 data lines from the processor bus for execution of a write command and 36 bits from the sense logic for execution of a read command as indicated earlier.

In CMB, information is strobed in from or out to only that bus which connects the processor selected by the processor priority logic in CMPC. Bits 22 through 35 are provided for normal mode. Interleaving is under switch control (S10 on CMPC logic) to substitute bit 21 for 35.

3.1.3 Detail Control Logic (Prints CMPC, CMC1, CMC2 (2 sheets), CMA, CMB, CMBR)

3.1.3.1 Data In and Out - The first three logic drawings for memory control show the 14-bit address register on BS-D-164-0-CMA and the 36-bit data buffer on BS-D-164-0-CMBR and CMBL. On all three drawings, input connections from all four possible processor/memory buses are shown. Data is strobed in from or out to only the processor/memory bus that is selected by the memory control logic; that is, either CMC P0, P1, P2 or P3 SEL.

The read and write request flip-flops are shown on CMA location B7, 8. Either or both of these control flip-flops is set at the beginning of the memory cycle by the READ REQUEST or WRITE REQUEST signal from the processor. The flip-flop output enables gates for sensing, clearing CMB, and starting the write portion of read access. The MA PO SEL, etc., signals are enabled by manual selection at S6 on Dwg. CMC2 (Sheet 2), or by the active flip-flop in multiprocessor mode.

CMBR and CMBL show the left and right halves of the core memory buffer CMB. The (CMC) CMB CLR pulse is generated at the beginning of every cycle by T1B. T1B was generated by RQ SYNC flip-flop set state which is controlled by T0 pulse, generated by any processor request for access and CMC SP (single processor mode), or by CMC RQ SYNC (⟶ 1) (for multiprocessor mode) on Dwg. CMC2 (Sheet 1).

CMB CLR is also generated again in the same cycle of a READ/WRITE. This is done 300 ns after T2 along with the gating of READ and WRITE RQ on a (1). This will clear the CMB prior to sending of new data to be written into memory.

There are two logically equivalent gating levels for each processor/memory bus. One gates the output pulses from the core bank sense amplifiers onto the bus; the other gates pulses from the bus into the CMB register. In this structure and in a read access, pulses from the sense amplifiers at the first level of gating are applied to the bus and to the second level gating for regeneration of data. During write access, data from the bus passes through the second level gating into the CMB register . (Drawing CMB shows the buffers through which the CMB outputs are applied to the core logic (to sense and inhibit lines.)

3.1.3.2 Memory Access Control Logic - The sequence of events and timing of memory access is described in Paragraph 3.3 and Figure 3-11. Here, the memory access control logic is described. This logic is shown on Dwg. CMPC, CMC1, and CMC2 (Sheets 1 and 2).

Drawing CMPC shows the control connections between the memory and the processor/memory buses. The left-hand half of each bus connecting module (1W31, 1X31, 1Y31, 1Z31, etc.) determines whether the memory is being addressed by decoding address bits according to switch selected connections. For interleaving, bit 35 can be switch-selected to replace bit 21. These signals are ANDed with two

control inputs: the processor request signal (P0RQ, P1RQ, P2RQ or P3RQ) from the processor shown at the right center of the drawing; and the AWAIT RQ(1) from the await request flip-flop on Dwg. CMC2 (Sheet 1). The processor request signal initiates an access to memory, a CMC AW RQ(1) enables the gate to allow the CMPC PQ RQ, etc., whenever a processor request arrives over the bus.

The four processor request signals are ORed in 1V28 as shown on Dwg. CMC1 (Core Memory Control Timing) to provide T0 and thus trigger the time chain.

The four CMPC processor request signals are also applied to the processor selection logic on CMC2 (Sheet 2). They serve as set inputs to the processor active flip-flops (PQ ACT, etc.) to determine which processor will have access for the current memory cycle. During a cycle, only one active flip-flop is set and it generates a pair of selection levels (CMC MB PO SEL, etc.) that are inputs to the two-level gating structure on CMBR and CMBL, thereby controlling data transfers over the bus. These signals are also enabling inputs to control gating on the right-hand side of CMPC to control transfer of address acknowledge signals at T1 time (Figure 3-11, Timing Chart), read restart signals (CMC RD RS), and write restart signals from the memory and processor to the bus.

On Dwg. CMC2 (Sheet 2), the processor priority logic controls processor access. If P0 requests access (the highest priority), P0 ACT is set and the other three flip-flops are held reset. If no P0 request is present and P1 requests access, P1 ACT becomes set and P2 and P3 are held reset. This order does not hold true for the last two flip-flops; if it did, processor 3 might never gain access. To avoid this, the last processor flip-flop (LAST PROC) is connected to remember which of P2 or P3 had last access. (The gating is enabled by the T2 timing pulse.) Thus, whenever P2 and P3 request access simultaneously, and neither P0 or P1 request access, the current cycle is given to whichever processor has gone without access the longer time. This is accomplished because simultaneous access requests of P2 and P3 complement the flip-flop. The LAST PROC flip-flop outputs are each applied to a gate at the reset side of each flip-flop. The other input to these two gates are the set sides of the flip-flops (i.e., P3ACT (1) is applied to the gate at the reset input to P2 ACT flip-flop and the converse of this). Thus a P3 RQ and P2 LAST (1) resets P2 ACT and allows P3 access. Conversely, a P2 request and P3 LAST (1) allows P2 access.

If only one CMPC processor request signal is present, when the memory becomes free, the corresponding active flip-flop is set and all others are left clear. When there are conflicting requests, the nets below the flip-flops determine priority. Every processor requesting access tries to set its active flip-flop by grounding its unbuffered 1 output. But the highest priority flip-flop that is set grounds the 0 sides of the others. This situation becomes stablized when the first pulse in the timing chain T0 clears the AWAIT RQ flip-flop (1V30, CMC2 (Sheet 1)) negating all processor request signals so that the indeterminate flip-flops become clear. Thus only one flip-flop is set.

The flip-flop that remains 1 generates the selection levels which connect CMBR and L logic to the selected bus, and gates control signals shown at the right in CMPC. Thus the address acknowledge and the read restart are sent back to the selected processor, the write restart from that processor is received, and the address strobe jam pulse, generated by the timing chain, loads CMA from the selected bus.

If a priority option is installed on a bus, the parity bit in or out passes through the right-most circuit on CMPC. In reading, the parity sense amplifier signal accompanies the data out; in writing, the parity bit accompanies the delayed write restart following the data.

3.1.3.3 Control Flip-Flops - Figure 3-3 illustrates basic sequencing of control flip-flops shown on engineering drawings CMC1 and CMC2 (Sheets 1 and 2). The three flip-flops READ, WRITE, and INH control the read, write, and inhibit drivers in the core logic. All these flip-flops are initially cleared by PWR CLR.

The READ flip-flop is triggered by the T1A pulse from the timing chain. The control is arranged to reset READ 450 ns after T1A. INHIBIT is set by T3 and WRITE by T4; then, T5 resets all three and sets AWAIT REQUEST on Dwg. CMC2 (Sheet 1). At this time, the memory becomes again free. READ (1) is on 450 ns, WRITE (1) is on 500 ns, and INHIBIT (1) is on 550 ns. CMC RQ SYNC flip-flop and CMC CYCLE DONE flip-flop on Dwg. CMC1 are used as synchronizing flip-flops. The RQ SYNC flip-flop monitors the priority network to see whether a processor request has been generated. This is done in two modes of operation. The first mode being SINGLE PROCESSOR MODE on which T0 which was generated by a processor request will set the RQ SYNC flip-flop. Second being MULTIPROCESSOR MODE in which T0 again is used, but is delayed by 100 ns (the time allowed for the priority network to deselect and settle down) and regenerated as RQ SYNC (⟶1) pulse which set RQ SYNC flip-flop.

The CYCLE DONE flip-flop monitors the timing control of the memory at the time the timing cycle has been completed. CMC T6 will set the CYCLE DONE flip-flop to a one provided that the CMC STOP flip-flop has not been set (this STOP FF is set by a switch or T2 on the indicator panel for single cycle operation) at the completion of a memory cycle.

When both RQ SYNC and CYCLE DONE flip-flop are set, the timing chain will be initiated starting with T1A, T1B, and MA JAM. These flip-flops are both cleared at T2 time and are also cleared by power clear pulse.

On Dwg. CMC2 (Sheet 1), the AWAIT RQ flip-flop is set at T5 or AW RQ (⟶1) from 1X26H. AW RQ (⟶1) pulse is an automatic setting of the AW RQ flip-flop in the event of a lower priority being first selected then deselected by noise on a higher priority bus line in which case all the active flip-flops would be left in the zero state, and the AW RQ FF being clear by T0 developed by the lower priority being selected first. This case is rare, but gives the priority network a second chance to

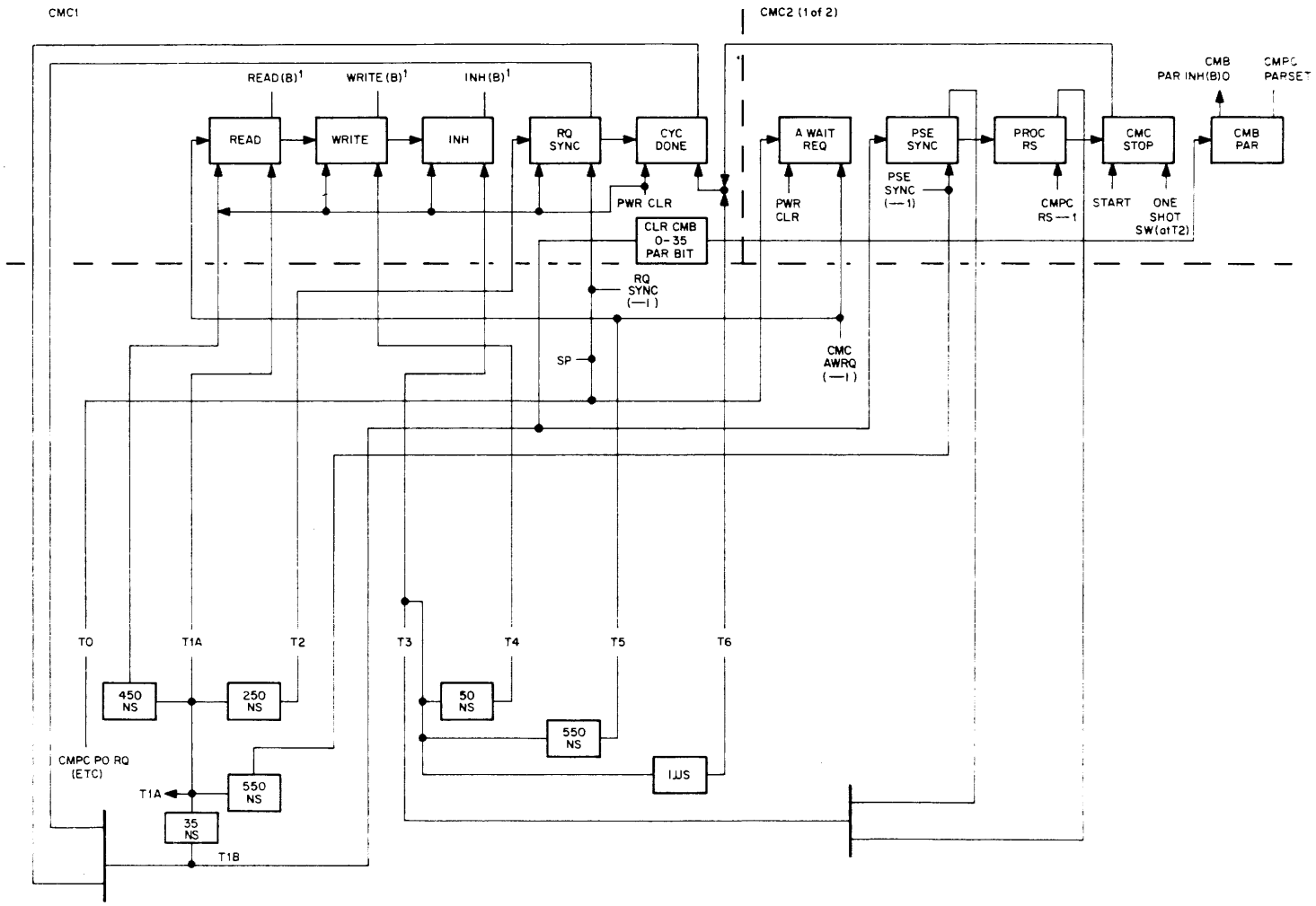


Figure 3-3 Control Flip-Flops

see what processor desires access. AWAIT RQ (1) indicates that the memory is free. The PSE SYNC flip-flop is set by the control signal PSE SYNC (⟶1), and functions to synchronize the write part of the cycle. PSE SYNC flip-flop is a synchronizing flip-flop which monitors the completion of the read protion of a cycle in either READ ACCESS, WRITE ACCESS or READ/WRITE. It is generated 100 ns after clearing the READ flip-flop. In the READ ACCESS, PSE SYNC is ANDed with WR RQ (1) to generate T3 which starts the write portion of the cycle. In the WRITE ACCESS, and READ/WRITE MODE PSE SYNC is ANDed with PROC RS to generate T3, which starts again the write portion of the cycle. The PROC RS flip-flop, through the RS ⟶1 signal from CMPC, receives the restart signal from the processor. The STOP flip-flop is controlled by a single step switch which allows T2 to set it.

3.1.3.4 Single Step Operation - If the CMC STOP flip-flop on Dwg. CMC2 (Sheet 1) is a ONE by T2 in a single operation selected by the SINGLE STEP switch, the timing chain stops at T6. To restart, the operator must push the RESTART button shown on CMC1 to generate the CMC RESTART pulse. This pulse provides CMC START and thus CMC STATE CLR. CMC START provides T6 to set up the CYCLE DONE flip-flop for a new memory access, and T5 to set AW RQ flip-flop.

3.1.4 Core Logic

The core logic consists of X- and Y-selection, inhibit logic, and sense logic. The core stack consists of 37 planes for 36 data bits and one parity. A simplified core stack arrangement is shown in Figure 3-4.

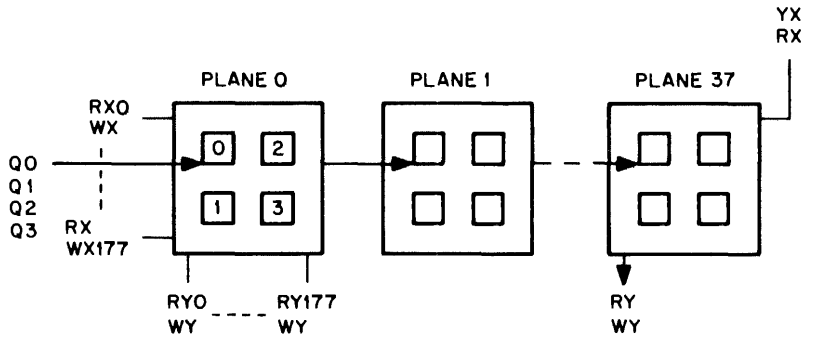


Figure 3-4 Simplified Core Stack Arrangement

Through all planes, the memory is divided into four quadrants for sensing and four segments for inhibiting. Figure 3-5 shows the addressing bit assignments for the memory.

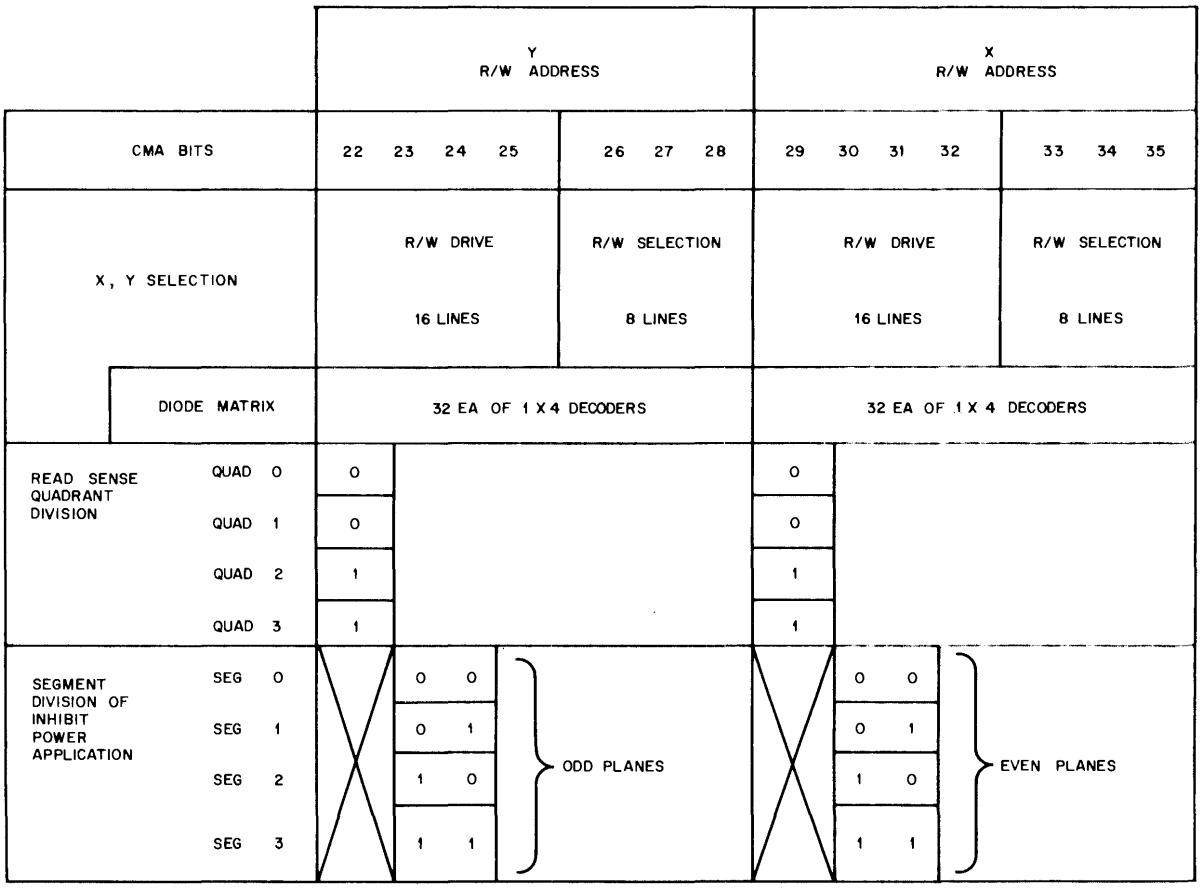


Figure 3-5 Core Memory Type 164 Addressing Bit Assignments

In a read cycle, X- and Y-selection is controlled by the 14 CMA bits shown and sense quadrant selection by the 22 and 29 bits, inhibit segment selection by 23 and 24 for odd bits and 30 and 31 for even bits. By this arrangement, any address can generate a proper sense quadrant selection signal and inhibit segment selection level. In this way, when an address is called only sufficient drive current is required to drive the selected sense quadrant and inhibit segment.

3.1.4.1 X- and Y-Selection Logic - The details of the logic are contained on drawings CMX, CMY, DMSX, and DSMY. The schemes for selection are the same for both X and Y; therefore, only the X selection is described and shown in the simplified diagram Figure 3-6.

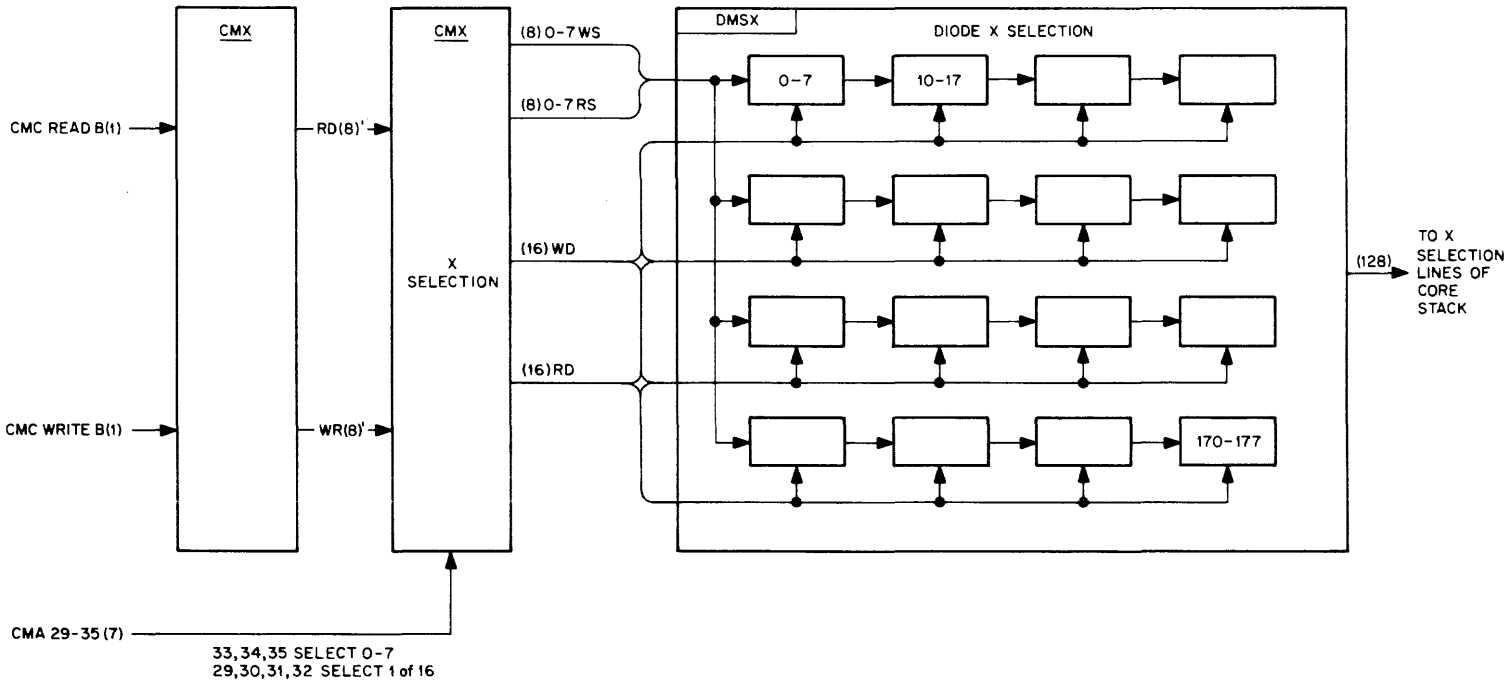


Figure 3-6 X-Selection Logic

The READ and WRITE flip-flops of CMC provide the enabling signals to the X selection logic which receives address bits 29 through 35. The outputs are 1 of 8 write or read lines that provide a single line into one of the 16 octal group circuits of the diode matrix, and 1 of 16 lines selects the proper octal group. The Y selection is accomplished in the same way. Bits 33, 34, and 35 select the 0 to 7 count within each octal group; bits 29, 30, 31 and 32 select 1 of 16 separate octal groups. The output of the diode matrix is 1 of 128 X-selection lines to the core stack.

3.1.4.2 Inhibit Logic - Inhibit logic details are contained on drawings CMI-1 and CMI-2. The simplified scheme is shown in Figure 3-7.

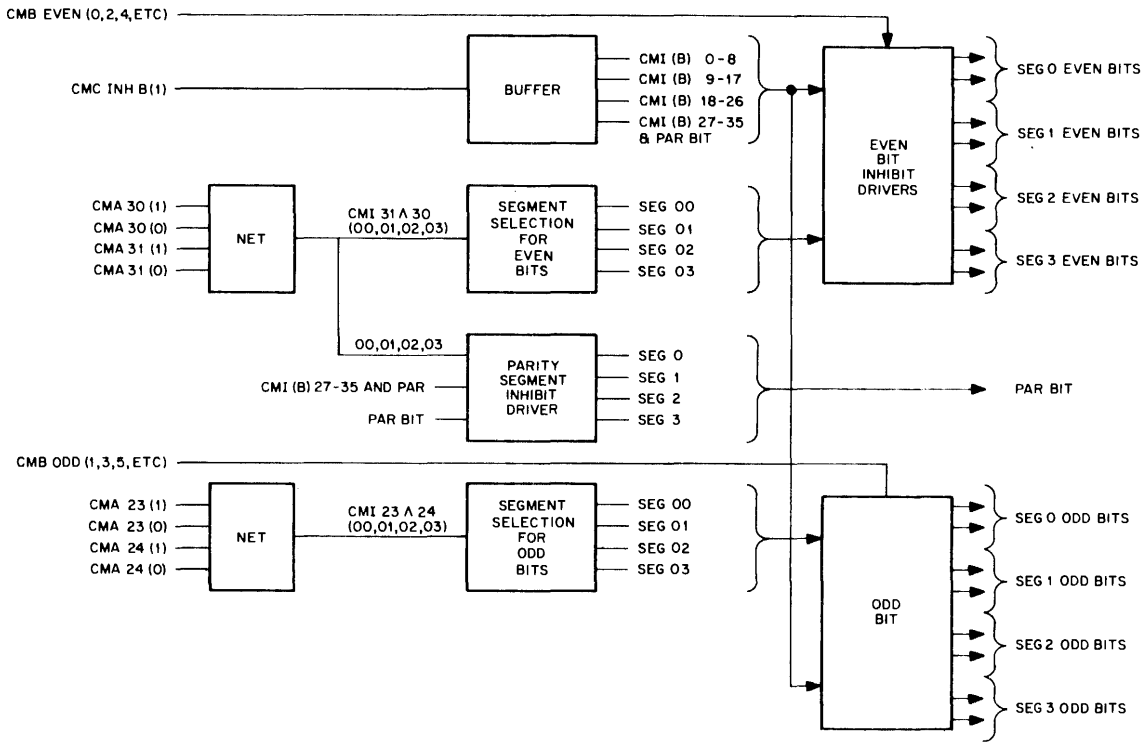


Figure 3-7 Inhibit Logic

The nets determine which segment contains the addressed location by decoding address bits 23 and 24 for odd bits and 30 and 31 for even bits. The inhibit driver for a plane provides an output pair for each of four segments. Inhibit current flows only for the duration of the inhibit signal from memory control, and only through drivers corresponding to CMB 0s at the CMB bit inputs. Thus, during a write cycle, 1s are written into all cores of the addressed location except into those planes which receive inhibit current.

Figure 3-8 shows the winding configuration for four segment inhibit winding for a 16K memory Type 164 System.

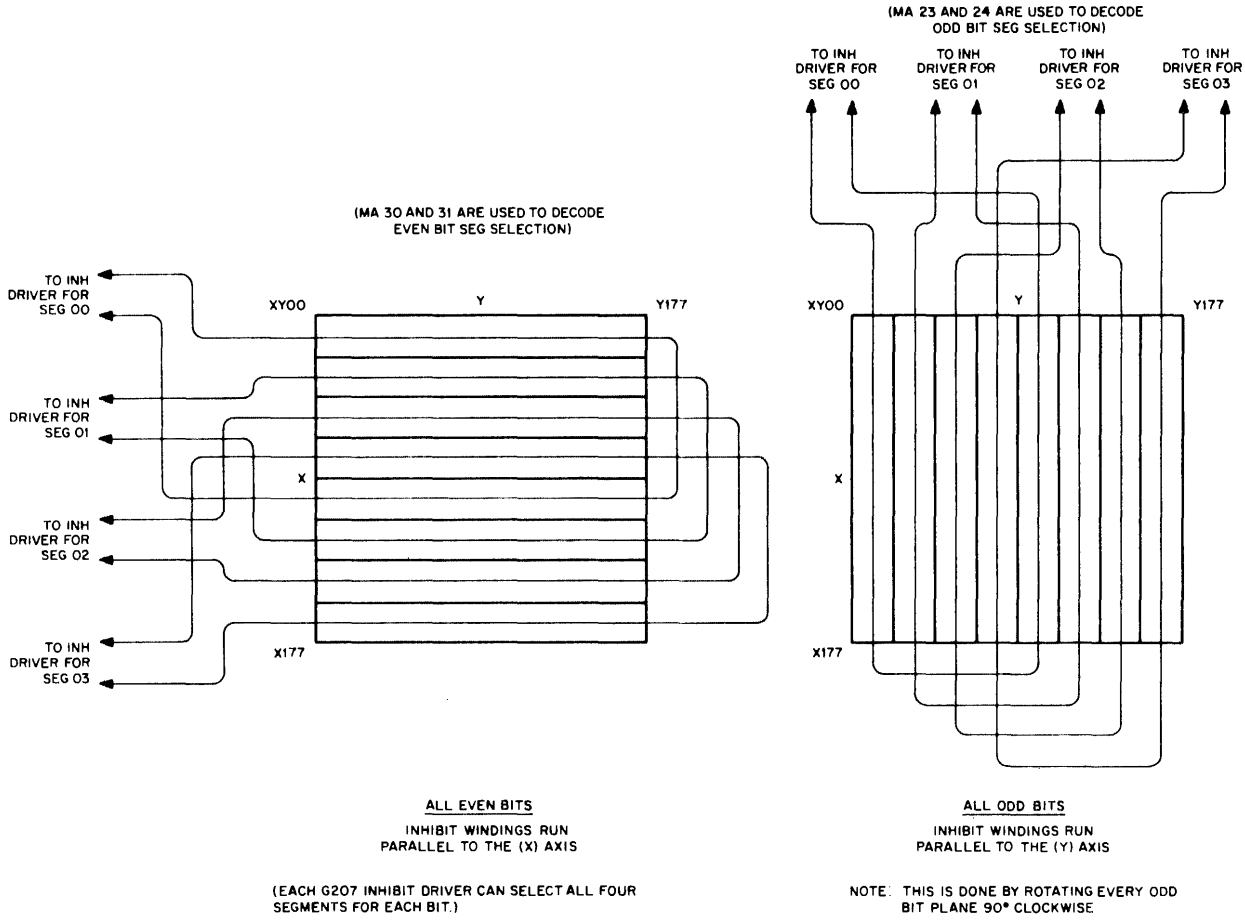


Figure 3-8 A Segment Inhibit Winding for 16K - 164 System

3.1.4.3 Sense Logic - Details of the sense logic are contained on MSSA1, (Sheets 1 and 2) and MSSA2. Figure 3-9 is a simplified diagram of the sense logic.

For read-sense quadrant selection, bits 22 and 29 are decoded. These are the MSB's of the X- and Y-addresses. The decoding circuits provide four signals for each quadrant to drive the sense amplifiers. Strobe inputs provide timing control. As in the inhibit scheme, the sense logic provides one pair of sense lines for each quadrant.

Figure 3-10 shows the winding configuration for four-quadrant sense winding for a 16K memory Type 164 System.

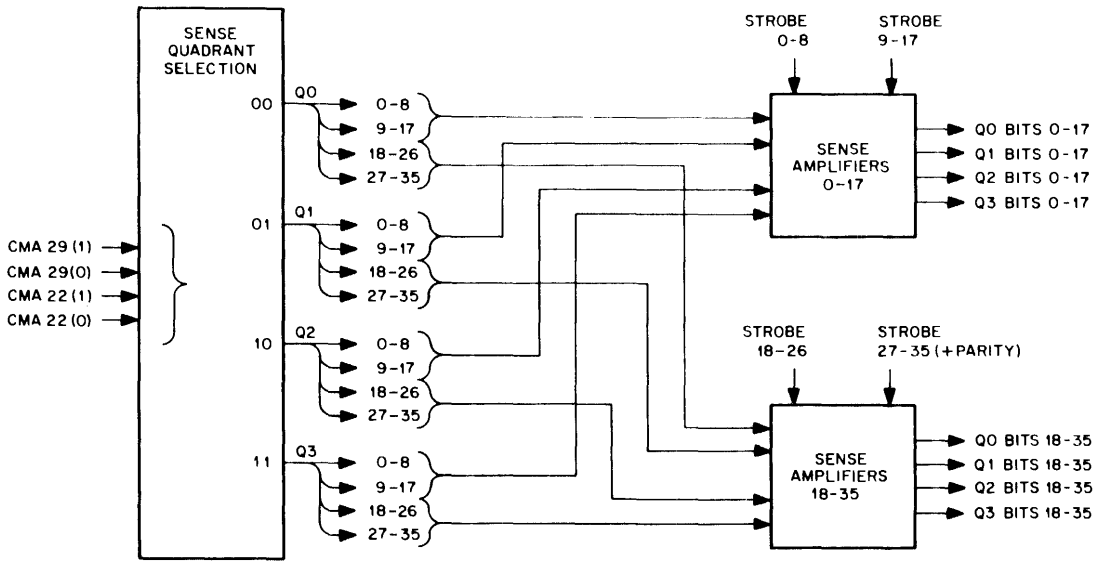


Figure 3-9 Sense Logic

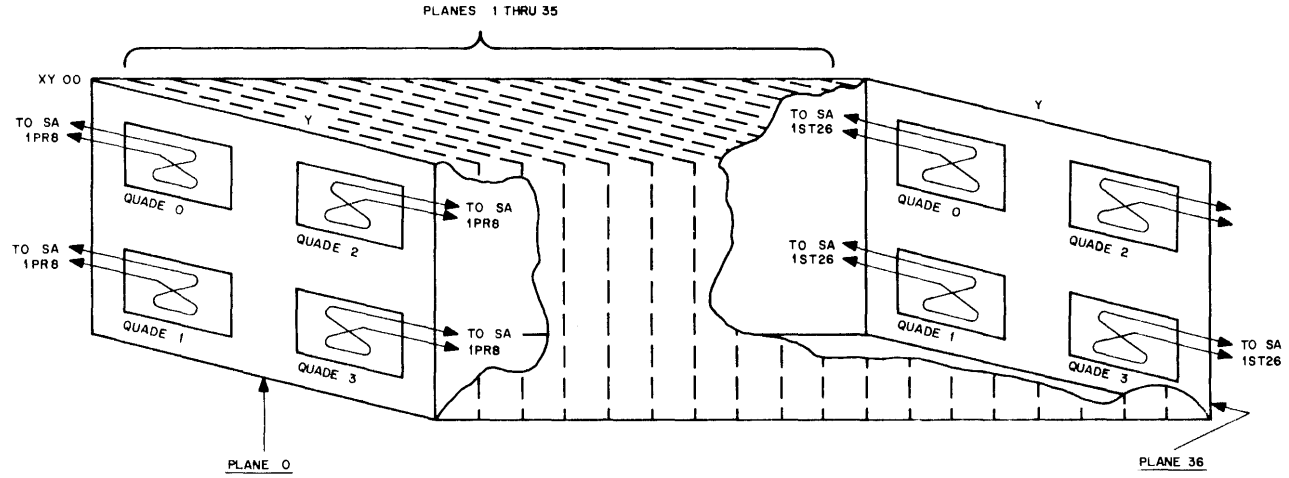


Figure 3-10 Four Quadrant Sense Windings for 16K - 164 System

3.2 TIMING

The memory system timing chain and associated logic are shown on Dwg. CMC1. Figure 3-11 briefly summarizes generation of the timing chain pulses.

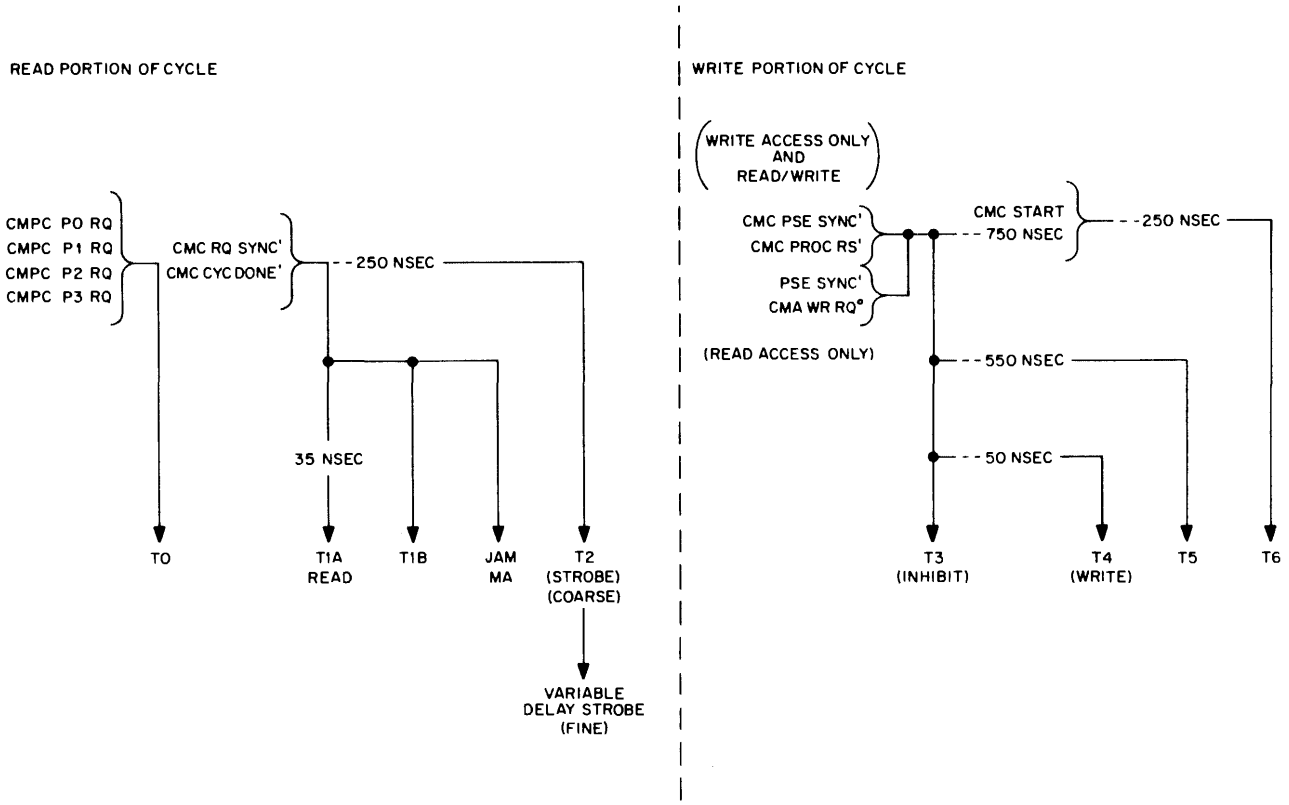


Figure 3-11 Timing Chart

Any CMPC processor request signal triggers the chain by generating T0. The T0 pulse sets the RQ SYNC flip-flop which generates T1A to provide JAM CMA 22 through 29 and JAM CMA 30 through 35 to set the address into CMA. This same flow generates CMB CLR 0 through 35 and the PAR BIT signal to clear CMB at the start of the cycle.

At this time, the processor priority logic on Dwg. CMC2 (Sheet 2) selects the processor that has priority for access.

If access is for write, the processor upon receiving ADDRESS ACKNOWLEDGE immediately loads CMB and returns the WRITE RESTART signal to CMPC. The processor generates a restart strobe RS < (1) which sets the PROC RS flip-flop on Dwg. CMC2 (Sheet 1). Because the memory at this point has the information for writing during the second part of the cycle, the PROC RS flip-flop transition triggers the STATE CLR signal on CMC1 to clear the active flip-flops on CMC2 (Sheet 2). PROC RS flip-flop is also ANDed with PSE SYNC flip-flop which initiates T3, the write portion. In this way, the memory is effectively disconnected from the processor and can then finish the write process while the processor is free to address another memory bank.

If access is for read, T1A occurs 35 ns after T1B and turns on the read signal to the core logic by setting the READ flip-flop on CMC1. Signal T1B on drawing CMPC enables the logic to send the ADDRESS ACKNOWLEDGE pulse back to the processor that has access. Then, 250 ns after T1A (READ), T2 (STROBE) occurs and triggers strobes for the sense amplifiers at the proper time during the read interval.

The logic at the right of CMC1 shows T2 connected to a variable delay to generate RD RS, and the strobe signals SA STB 0-8, 9-17, 18-26, and 27-35. While information is strobed from the sense amplifiers into CMB and out over the bus, the RD RS signal goes to CMPC for return to the processor.

In restoring the information the CMC WR RQ flip-flop is ANDed with PSE SYNC flip-flop to initiate T3 the write portion. T3 also is gated with WR RQ on a 0 to develop a state clear the active flip-flop, and thus disconnect the bus from memory to processor. No WR RS signal is initiated during a read access only. Again the memory will go on and write leaving the processor free to address another memory bank.

The READ/WRITE is much the same as a read access except for the following: The condition of CMA RD RQ (1) and CMA WR RQ (1) is true. This enables a second CMB CLR pulse to be generated 300 ns after T2 so that the CMB register will be cleared when it is ready to accept new information for the write portion of a READ/WRITE CYCLE. A WR RS pulse is sent back to the memory to set the PROC RS flip-flop which is gated with PSE SYNC flip-flop to generated T3 which is the start of the write portion. The PROC RS flip-flop also generates a state CLR pulse to clear the active flip-flop and again disconnect the bus and leave the processor free to access another memory bank.

In the second part of the cycle, the word CMB, whether supplied by the processor or read from memory during the first part, is written in the addressed location. T3 starts the inhibit level by setting the INH flip-flop on CMC1. Then, 30 ns later, T4 sets the WRITE flip-flop to provide the WRITE(B) 1 level. At 500 ns after T4, T5 is generated to terminate the write cycle by resetting the WRITE and INH flip-flops, and setting the AWAIT RQ flip-flop on CMC2 (Sheet 1) to indicate that the memory is again available for a processor request. T6 or CMC STOP B (0) provides the CMC CYC DONE (1) to set the CYC DONE flip-flop.

3.3 POWER SUPPLIES AND CONTROLS

The remaining logic to be discussed is associated with memory power turn on and power failure. Connections to the power control relay Type 836 are shown on Dwg. 164-0-CMC2 (Sheet 1). The PSOK signal generated from the 739D power supply, and W505 (low voltage detector) which indicates when power levels are satisfactory for operation, are shown on Dwg. 164-0-CMC1.

The PSOK logic level is an input to the 200 ms leakout delay (IUV32). The 0 output of this delay supplies the input to pin V of the start delay (IUV31). This type of delay is triggered by a transition to ground at pin V (delay period does not begin until the pulse at V disappears). Thus when the power comes on, both delays come on in the 1 state.

At turn-on time, power voltage changes are slow so that the 1 output of the upper delay that triggers the power start through PA at IU30-U is slow. To prevent premature processor access, the 1 output of the lower delay held asserted by PSOK, holds CMC AW RQ in the 0 state and thus prevents any processor from gaining access.

When the start delay (1UV31) times out (100 ms), its 0 output supplies a power clear level that holds off the core-control flip-flops.

When power becomes satisfactory (PSOK), the period of the lockout delay (200 ms) begins. At its completion, it drops the request lockout ($\not 0$ AWAIT RQ), and its 0 output sets and holds the start delay (100 μs). The transition in this generates the START signal and enables T5, which sets the AWAIT RQ flip-flop to indicate that the memory is available to the bus, and T6 which sets CYC DONE flip-flop (1).

When the memory is operating, the lockout delay (IUV32) is off and its 0 output holds the start delay on to insure continuous operation. If power levels become unsatisfactory, PSOK is negated, triggering the lockout delay and allowing the start delay to begin. During this period, the current memory cycle is completed. After this delay times out, the 0 output power-clear level inhibits further operation.

Each memory has a pair of power controls, Types 836 and 834, shown on Dwg. CMC2 (Sheet 1). The Type 836 Power Control contains a diode gate that receives the -15V turnon signals supplied from the power terminal. The output of the gate goes through the power switch on the control panel and back to the 836 to energize a relay that turns on the Type 834 Power Control. The Type 834 Power Control receives the local ac line voltage and provides this to the system power supplies.

Power supply Type 728 provides +10 and -15 Vdc and Type 778 Power Supply also provides -15 Vdc for the memory logic and the core memory control. Power Supplies 739 provide voltage for circuits associated with the driving of the core bank.

CHAPTER 4
SPECIAL MEMORY MODULES

This chapter explains the seven special memory modules used in the Type 164 Magnetic Core Memory System. These modules are analogue in nature and are not explained in the standard logic handbook. To aid the maintenance personnel with troubleshooting, a brief functional description of each module that indicates the input and output signals is provided. A block diagram of each module is also provided.

4.1 G005 SENSE AMPLIFIER

This double-height module detects the ONE and ZERO outputs from coincident-current core memory systems. Each module has the capability of sensing one of four fields. The field being sensed is selected by holding the appropriate preamplifier field select gate at the -3 Vdc level and the other three field select gates are deselected with a -6 Vdc level. Refer to Figure 4-1 for the following functional description.

To obtain optimum operation, each preamplifier has a balance potentiometer that is used to adjust the preamplifiers output within ±200 mV of balance. The outputs of the four preamplifiers are connected in parallel to a common amplifier. From the common amplifier, the output signal is applied to a rectifying slicer and the resultant signal from the slicer is used to enable a DCD gate. When the selected core output is a ONE, the DCD gate is enabled and the pulse amplifier is gated by the 40 ns strobe pulse. The pulse amplifier generates a 100 ns output pulse.

A G008 Master Slice Control determines the operating current of the preamplifier and the common amplifier and also the slice level for the rectifying slicer.

Inputs

Preamplifier:	Accepts a signal up to a 50 mV from the sense lines.
Strobe:	A 40 ns negative pulse that draws 1 mA at -3 Vdc and negligible current at ground.
Field Select:	SELECT: A -3V level that draws approximately 5 mA due to the transient effect of the memory stack capacitance. DESELECT: A -6V level that draws approximately 10 mA.
First Stage (Clamp Level):	Operates at 3.8 Vdc with respect to +15 Vdc and draws 0.7 mA.
Second Stage (Clamp Level):	Operates at 8.0 Vdc with respect to +15 Vdc and draws 0.3 mA.
Slice Level:	Operates at a dc level between 5.5V to 5.9V with respect to +10 Vdc and draws 0.4 mA.

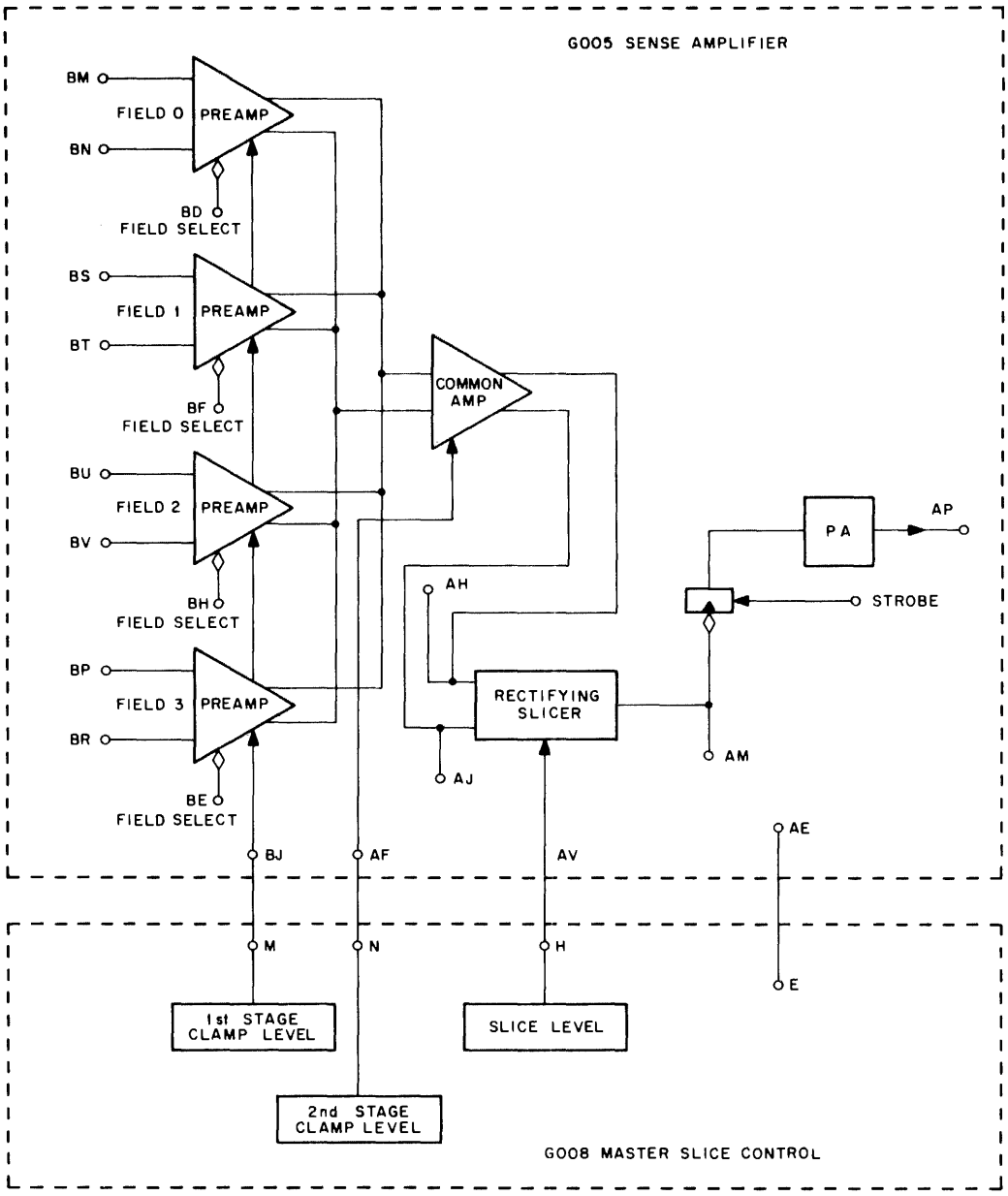


Figure 4-1 G005 Sense Amplifier

Output

Amplifier:	Overall amplifier gain between 65 and 85, measured single - ended at the outputs AJ or AH.
Rectifying Slicer:	Output capable of driving 1.0 mA at ground level and 8.5 mA at -3 Vdc.
Pulse Amplifier:	A 100 ns 3V negative pulse capable of driving 10, 2-mA diode gates. A negative pulse out is a ONE.

4.2 G008 MASTER SLICE CONTROL

This module supplies the first stage clamp level, the second stage clamp level, and the slice level required by the G005 Sense Amplifier. Refer to Figure 4-2 for the module block diagram. Each circuit consists of a zener diode network with silicone diodes for temperature compensation. Emitter follower stages provide current driving capability and low-impedance outputs.

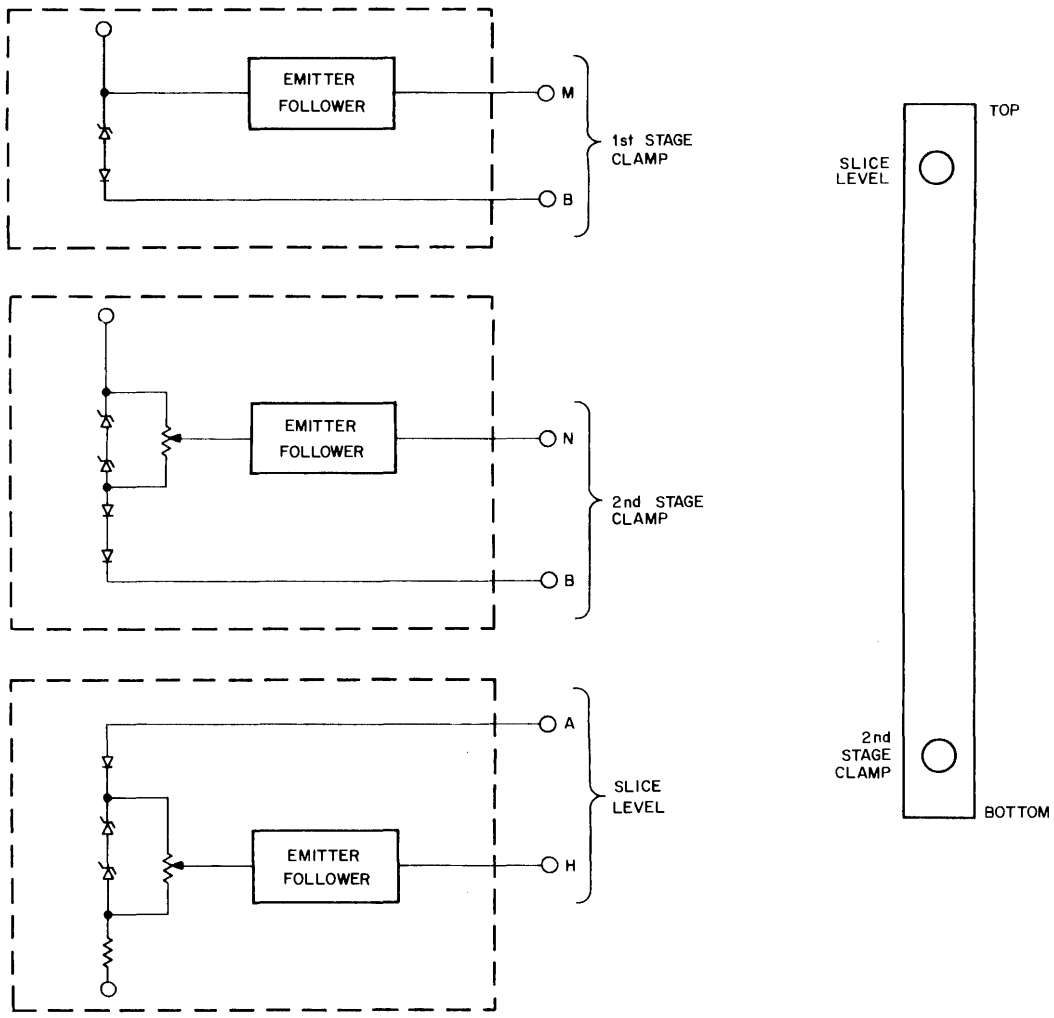


Figure 4-2 Master Slice Control

Output

First Stage (Clamp Level):	3.9 Vdc with respect to -15 Vdc
Second Stage (Clamp Level):	Variable between +0.6 and +13 Vdc with respect to -15 Vdc.

Slice Level: Variable between 0 and -11.6 Vdc with respect to +10 Vdc.

4.3 G010 SENSE AMPLIFIER SELECTOR

This module contains two noninverting driver circuits. Using standard input levels, each drive circuit can drive a large number of base loads and diode loads with levels of -3V and -6V. Refer to Figure 4-3 for the module block diagram.

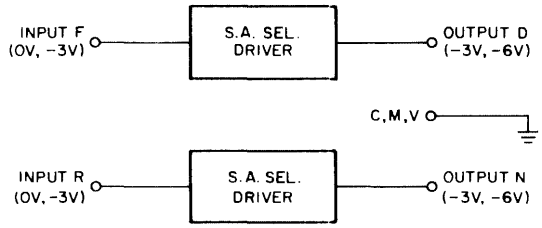


Figure 4-3 Sense Amplifier Selector

Input: Standard DEC logic levels are used; 2 mA of base load are used when the input is -3 Vdc.

Output: A -3 Vdc level occurs when the input level is at ground; a -6 Vdc level occurs when the input level is at -3 Vdc. These levels can drive ±40 mA at 10 Mc. Drive delay is approximately 40 ns.

4.4 G206 MEMORY SELECTOR

This module is used as a selector switch in the read/write matrix of coincident current memories. Refer to Figure 4-4 for the following functional description. Each module contains two read gates and two write gates. Due to the decoding, only one read gate and one write gate may be enabled at one time. Although this module is primarily intended for four-bit decoding, it may be used for three-bit decoding by grounding pin AK.

Input: Standard DEC -3 Vdc levels are used for turn on. The decoding gates and the LSB gate draw one unit of base current. The read or write gates draw two units of base current.

Output: Each output can drive 425 mA for 500 ns at a PRF of 1 MHz. The following specifications refer to the output waveform.

Maximum delay for output fall: 80 ns

Maximum delay for output rise: 50 ns

Minimum cycle time: 1.3 μs

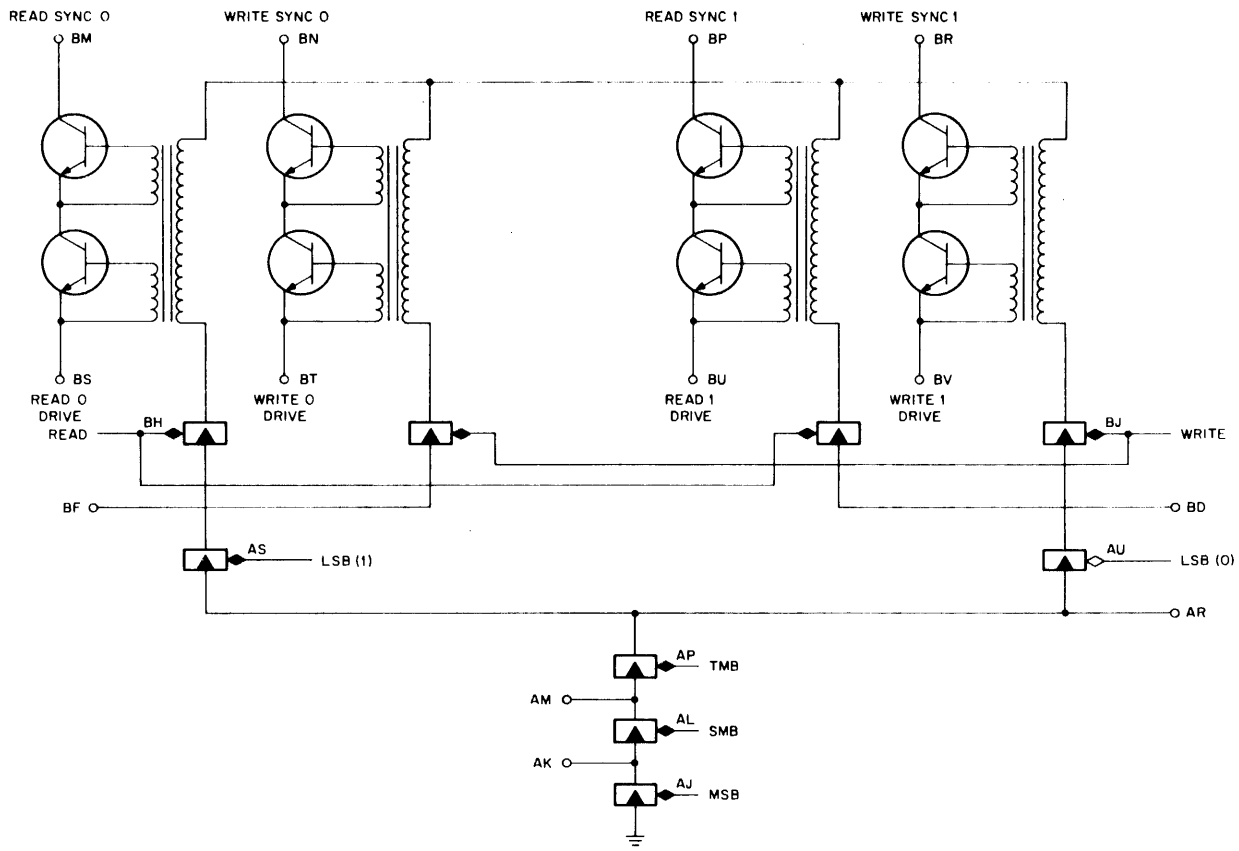


Figure 4-4 G206 Memory Selector

4.5 G207 INHIBIT DRIVE

This module is used to drive the inhibit winding of magnetic core memory planes. Refer to Figure 4-5 for the following functional description. Each module contains four identical gated circuits. A balun (1/1 balanced trap) at the output is used to obtain balanced drive. The switched current is determined by the inhibit resistor.

Input: Standard DEC -3 Vdc levels are used for turn on. The quadrant selection gates draw two units of base current and the MB gate-input draws three units of base current. Each of the inhibit gate inputs draws 1 unit of base current.

Output: Each output can drive 350 mA for 600 ns at a PRF of 750 KHz. The following specifications refer to the output pulse.

Maximum delay for output fall: 60 ns

Maximum delay for output rise: 60 ns

Minimum cycle time: 1.3 μs

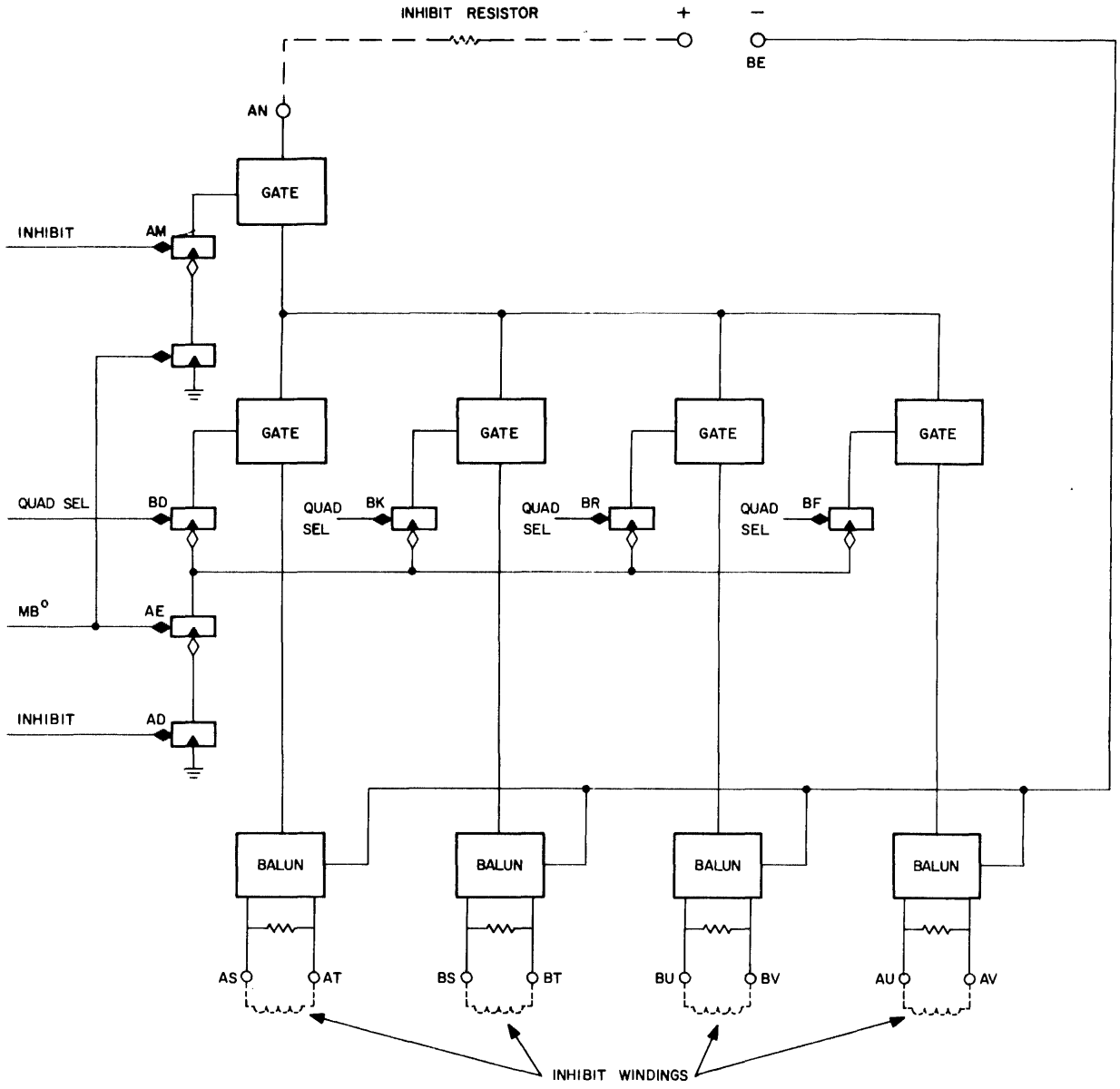


Figure 4-5 Inhibit Drive

4.6 G212 MEMORY COMMON DRIVE

This double height module is used as a common drive in the read/write matrix of coincident current memories. Refer to Figure 4-6 for the following functional description. Each module contains two read gates and two write gates but the decoding gate enables only one read gate and one write gate at a time. The decoding circuit is used to decode a 4-bit address. Clamping diodes are connected to terminals AE and AF to limit transients, and compensating resistors are connected to terminals BE and BK to bias the diode balun matrix.

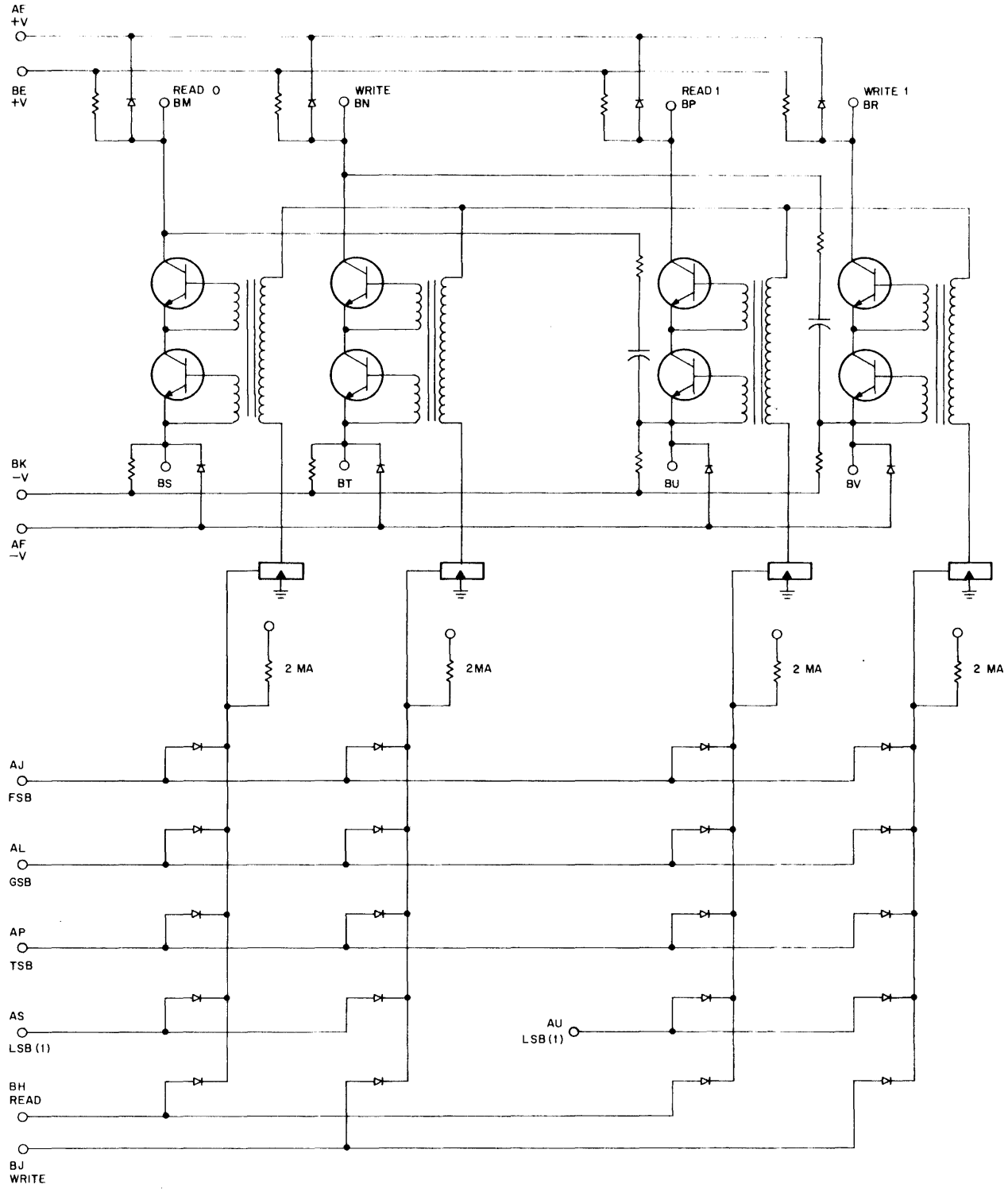


Figure 4-6 Memory Common Drive

Input: Standard DEC -3 Vdc levels are used for turn on. All input gates draw 2 mA of current at ground.

Output: Each output can drive 425 mA for 600 ns at a PRF of 750 KHz. The following specifications refer to the output waveform.

Maximum TTT of output fall: 150 ns

Maximum TTT of output rise: 100 ns

Minimum cycle time: 1.5 μs

4.7 G604 MEMORY SELECTOR MATRIX

This module is used to select the read or write windings of a coincident current memory. Refer to Figure 4-7 for the following functional description. There are four diode-balun networks on each module. Each diode-balun network provides a current path through one winding for read and the reverse current path for write. A balun transformer is used to provide balanced drive.

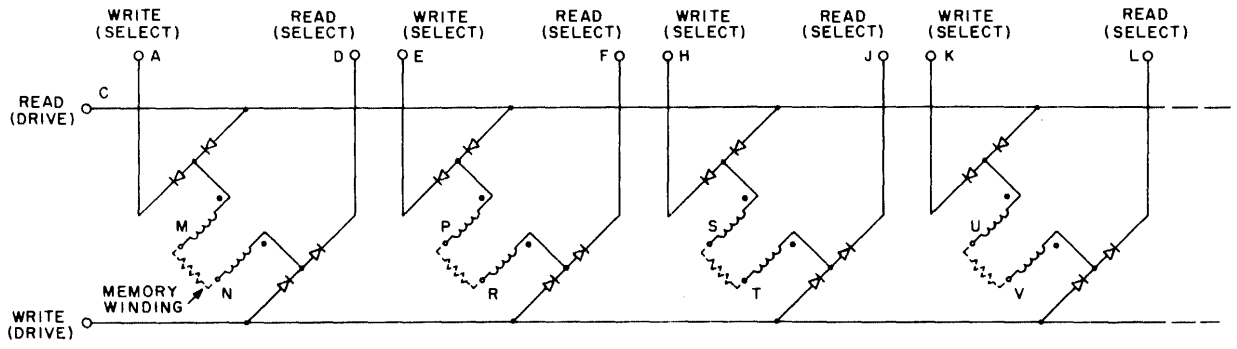


Figure 4-7 Memory Selector Matrix

Input: A 425 mA current for 500 ns at a PRF of 1 MHz. The output pulse from a G206 module.

Output: Each output can drive 425 mA for 500 ns at a PRF of 1 MHz. The following specifications refer to the output waveform.

Maximum TTT output fall: $\leq$ 130 ns

Maximum TTT output rise: $\leq$ 85 ns

CHAPTER 5
MAINTENANCE

Preventive and corrective maintenance at the system level are discussed in this chapter. Since the memory must operate with a Type 166, Arithmetic Processor the discussions presented here, should be used with a full knowledge of the processor and in conjunction with the descriptions in Chapter 9 of the Type 166 Arithmetic Processor Maintenance Manual. Many memory maintenance procedures such as MAINDEC programs use the processor.

The following equipment, with that listed in Chapter 9 of the Type 166 Maintenance Manual, is required to support testing of the Type 164 Memory.

Oscilloscope current probe	Hewlett Packard 1110A or equivalent, calibrated to 1 mV per mA
Dual-channel oscilloscope preamplifier	Tektronix Model CA or equivalent; must have algebraic-add facility.

5.1 MAINDEC DESCRIPTIONS

There are six basic MAINDEC programs fo use in trouble shooting the memory system; all use the processor to exercise either fast or core memory. Each consists of a program tape and a write-up in the 6-MD-DEC reference manual. As an introduction to the MAINDEC write-ups, see Chapter 9 of the Type 166 Processor manual. Always consult the program write-ups for information on start addresses, address modification, or details of program operation. MAINDECS for the memories are as follows.

MAINDEC 603-1 Low-End Address Test - This address test checks registers for unique contents.

MAINDEC 603-2 High-End Address Test - This program, which is loaded into the top of memory (highest available addresses), checks each location for unique contents by loading it with its own address.

MAINDEC 613 Core Data Test - This MAINDEC tests all locations from 40 up, except the top 27, which are occupied by the program. Each time the test word, originally supplied by the DATA switches, is read in and out, the contents of core are tested for accuracy; then the test word is rotated once and the same location is tested again. After 36 rotations the address is indexed and the next location is tested in the same manner. Installations having the two core memories can use this program

most efficiently by modifying its address limits so that the program is contained in one memory while the other is tested completely. The program is then immune to marginal checking or trouble shooting procedures on the memory under test.

MAINDEC 622A High Speed Checkerboard Test - This program, in the presence of a noise-producing data pattern, tests the Type 164 Memory for read-out errors. The program has several options, one of which allows the worst case noise to be generated for any memory or interleaved configuration currently produced. This test should be used to verify correct operation of the memory sense amplifier and inhibit drivers.

MAINDEC 691 Automatic Block Transfer Test - This test detects and diagnoses malfunctions occurring in the block transfer instruction. When run with data switches 19-35 down, it serves as a relocation test for adders and the memory address register. It calculates and prints the time every four or five minutes thereby providing a time record of the program run. The program is executed in the user mode. With data switches 19-35 down, the block transfer buffer size and relocation are generated by random numbers; otherwise, they are controlled by the data switches. The block transfer effective address is a function of the buffer size and the "from" and "to" addresses.

MAINDEC 693 Time Sharing Instruction Test - This test runs Parts 1, 2, 4 and 5 of the PDP-6 instruction tests in the user mode to test the time sharing hardware while concurrently testing the instruction set of the PDP-6.

5.2 PREVENTIVE MAINTENANCE PROCEDURE

Preventive maintenance consists of performing specific procedures at scheduled times, using maintenance programs, marginal checks, and other electrical and mechanical checks, including cleaning and inspections.

5.2.1 Daily Operator Maintenance

Check that all cooling fans are running and cooling air flows freely through the filters.

Run the following MAINDEC programs with no margins; log all error halts, and note the cause, if known. Refer to each MAINDEC program write-up for proper operation.

MAINDEC 603-2 High-End Address Test

MAINDEC 603-1 Low-End Address Test

MAINDEC 613 Core Data Test

MAINDEC 622A High-Speed Checkerboard Test

MAINDEC 691 Automatic Block Transfer Test

MAINDEC 693 T/S Main Test

Replace any noncritical components such as indicators, etc. Note any replacement in the log.

5.2.2 Weekly

Using MAINDEC 622A High-Speed Checkerboard, take moderate margin voltage ±5V on sense amplifier panels 1P and 1S 10V line. Log all error halts, noting cause, if known.

5.2.3 Every Two Weeks

Change lower fan filter for memory stack.

5.2.4 Every 1000 Hours

Run the following MAINDEC programs taking margins in the following panels as shown below. Refer to each MAINDEC program write-up for proper operation.

MAINDEC 603-2 High-End Address Test

MAINDEC 603-1 Low-End Address Test

MAINDEC 613 Core Data Test

MAINDEC 622A High-Speed Checkerboard Test

MAINDEC 691 Automatic Block Transfer Test

MAINDEC 693 T/S Main Test

(+10V margins are taken in the following panels, in groups as shown.)

1A through 1F	(Inhibit and R/W)
1P and 1S	(Sense amplifiers)
1R, 1T, 1U, 1V	
1W, 1X	(MB and control)
1Y and 1Z	(MA and control)

(-15V margins are taken in the following panels, in groups and separately as shown.)

1A through 1F	(Inhibit and R/W)
1P and 1S	(Sense amplifiers)
1R	(MB)
1T	(MB)
1U	(Control and interface)
1V	(Control and interface)

1W	(Control and interface)
1X	(Control and interface)
1Y and 1Z	(MA and control)

NOTE: Log all margins on preventative maintenance voltage charts, form #SA-1, dated 5/25/64.

Change and clean all air filters.

5.3 CORRECTIVE MAINTENANCE

The Type 166 manual contains a general discussion of troubleshooting and repair for DEC logic, with a corrective maintenance procedure assigned, to make most efficient use of field engineering time. It is essential that this material be understood.

5.3.1 Power Supply Checks

Before troubleshooting, always check the power supplies for proper voltage output and ripple content as shown in Table 5-1.

Table 5-1
Power Supply Outputs

Measurement Terminals	Nominal Output Voltage (70 F)	Permissible Range	**Permissible Peak-to-Peak Ripple
	Type 728		
Orange (+), Black (-)	10	9.5 to 11.0	0.7V
Blue (-), Black (+)	-15	-14.5 to -16.0	0.7V
	Type 739		
(Jones Strip)			
2(+), 1(-) - Inhibit -	53*	52 to 55	250 mV
4(+), 3(-) - R/W -	65*	63 to 65	250 mV
	Type 778		
Blue (-), Red (+)	-15	14.5 to 16.5	0.7V
Blue (-), Red (±)			

(*) Exact voltage dependent on stack temperature.
(**) 20% more ripple on 50-cycle types.

The Type 728 and 778 Supplies are not adjustable, therefore, if output or ripple is not within the tolerance specified, the supplies should be considered defective and replaced. The Type 739 Supply is adjustable; but, if outputs are outside tolerance, do not attempt to restore the voltages given above without first checking the core drive currents. If all supply outputs are within tolerance, continue logic troubleshooting procedures.

5.3.2 MAINDEC Testing

The most efficient method of troubleshooting a memory is by tracing signals with an oscilloscope. To set up repetitive, predictable behavior, deposit test routines derived from a MAINDEC into a memory other than the one under test, so that even the most severe malfunctions in the latter do not affect the testing routine. After diagnostics have categorized a malfunction, a small part of the memory logic can be operated repetitively by using the processor EXAMINE or DEPOSIT key with the REPEAT switch ON and the speed controls set to the desired repetition rate. These keys make requests only for read or write access, but read/write access can be requested from the computer console by executing an appropriate instruction, preferably the slowest (e.g., subtract in memory mode).

For apparent malfunctions in memory, examine the processor for faults by checking for the presence of signals on the bus without distrubing it; to do this, wheel the scope down to the last memory where the probe can be attached to the bus without distrubing it. Alternatively, there are many points in the processor wiring where memory return signals or their derivatives appear; signal tracing in that area can determine whether the memory sends back the appropriate responses. When troubleshooting, always turn the DISABLE MEMORY switch at the processor ON so that it will hang up whenever it cannot gain access.

5.3.3 Marginal Checking

Nearly all core memory malfunctions are discovered during marginal check process. As margins are increased from nominal bias values, a single bit generally fails first; such failure may be random or systemic. Check for randomness of failure by increasing margins still further: whenever a very slight increase produces errors in several additional bits, the first error can be considered random. Errors occurring at margin levels substantially below those at which most bits are not random; instead, they signify some systemic malfunction that must be corrected. The remainder of this chapter correlates symptoms to their most probable causative malfunctions, assuming that the malfunction is isolated within a single 16K memory. When errors are truly random, but occur at unsatisfactory margins, the adjustment procedure detailed in the following section should be performed to improved marginal

operation before continuing troubleshooting. When margins for properly operating circuits are wide enough, nonrandom malfunctions are more easily detected. Nonrandom errors always occur in specific data patterns.

5.4 ACCESS MALFUNCTIONS

All access troubles a single symptom; either the processor or the memory hangs up. The fault is always a lack of communication between the two, or a signal ambiguity caused by noise or internal memory control malfunction. With the DISABLE MEMORY switch ON, the system will wait after an access failure, allowing examination of the indicators. If the memory AW RQ indicator is OFF, the memory has failed to complete its cycle. If AW RQ is ON, the hang up is at the processor, which has not received the read restart or address acknowledgment over the bus. These pulses turn off indicators at the top of bay 1 at the processor. If the MC RD indicator is lit, the processor failed to receive the read restart during a read or read/write access. If MC RQ is ON, the address acknowledgment was missing. If either system is confined to a single processor in a multiprocessor system, the fault is almost certainly in the CMPC bus associated with it.

5.5 TROUBLESHOOTING GUIDE FOR MEMORY TYPE 164

5.5.1 Memory Does Not Recognize a Processor Request

Check to make sure that all switches on the memory are in the proper position for memory module address and named processor. Check the CMPC (Core Memory Processor Control) cables for bad connections, or for misalignment. Scope for proper memory module address signals.

5.5.2 Memory Hangs-up With AWAIT REQUEST Zero

A possible cause is multiple selection due to noisy module address lines from the computer. If there are negative-going transients on the module address lines after MC REQUEST has appeared on the bus, one memory may not be completely selected; that is, the active flip-flop in that memory may not get set to 1. If this occurs, then no addressing information and no READ or WRITE REQUEST will get strobed into the memory. T0, will probably have been generated, and will clear the AWAIT REQUEST flip-flop, but that is as far as it will go. The next time the computer tries to reference the memory, the memory will not be available so the non existent memory flag will be set.

Another cause may be the READ-RESTART or ADDRESS ACKNOWLEDGE not getting back to the processor on a READ. If the ACKNOWLEDGE does not get back to the CP on a WRITE, the memory will also hang with AWAIT REQUEST zero. On READ/PAUSE/WRITE the memory will hang

with AWAIT REQUEST zero if the RESTART does not get back, and on either a READ/PAUSE/WRITE or a WRITE, the memory will also hang if the WRITE RESTART does not get from the processor to the memory. The best way of determining which of these things is happening is simply to understand the exact interchange of signals between the memory and the CP, for each type of memory cycle.

The following malfunctions also could cause problems: the failure of the REQUEST SYNC flip-flop to be set to L by either T0 in the single processor mode or REQUEST SYNC in the multi processor mode; the failure of the PSE SYNC flip-flop to set, thus preventing T3 from being generated; the failure of T5 to set the AWAIT REQUEST flip-flop to 1; a noisy W505 in the 739 Power Supply causing transients on the PSOK line; or the self-triggering R303 hangs AW RQ FF to 0.

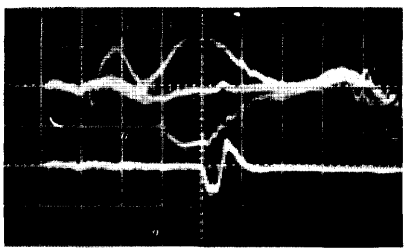
5.6 FULL-WORD FAILURES

Check the pattern of locations that exhibit full-word failures to determine whether they are limited to one or two quadrants or occur throughout a memory. Pickup errors throughout a single quadrant indicate trouble in the sense quadrant selection; check memory address bits 22 and 29 and trace the decoder outputs through. Full-word drop errors in two quadrants indicate that CMABB22 or 29 does not govern quadrant selection in the sense amplifiers; check these levels. Errors affecting the full word in all locations cannot be due to malfunction in the core logic; troubleshooting should thus concentrate on the power supply, memory control, or selection logic. If the power supply outputs are within the tolerances specified at the beginning of this section, malfunction in the supply is doubtful. In general, supply malfunctions can be distinguished from control malfunctions by taking margins on the control panels; poor marginal operation here is due to control-logic faults, since power supply adjustments affect only the sense amplifier margins.

For all types of full-word faults, begin by checking the address logic. Errors in the transfer of addresses from MA in the processor or malfunction in CMA bits or their buffers cause duplication of addressing; a given core location is selected by more than one MA configuration. Thus the same information is read twice, once erroneously, causing data errors in the checkerboard or BTL program (block transfer). If the system is dropping bits, check the logic for transfers into CMB; compare the CMB and processor MB indicators to check data transmission over the bus; check the memory strobe generating logic. For pickup errors check the timing and duration of the inhibit level, and check the CMB clear logic. When the memory exhibits generally poor margins and is both picking up and dropping bits, check the timing of the read and write levels. If necessary, replace the delay lines (W301) to restore this timing requirement.

5.7 SINGLE-BIT FAILURES

A single-bit error that affects only one address results from a single core whose characteristics have drifted out of specifications. Since every memory is checked out thoroughly before shipment, it can be assumed that the cores are not malfunctioning unless they have been subjected to extreme punishment. A single-bit pickup error throughout memory can be caused by a chort or other low impedance path between an X or Y winding and a sense winding, but this is easily recognized by checking the sense preamplifier outputs. Each sense amplifier has test points on the wiring panel for checking internal waveforms. The waveforms at the test point is the differential output of the selected preamp and is shown in Figure 5-1.

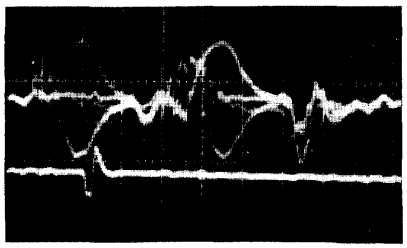


Single ended core read output in reference to strobe.

Time: 100 ns/cm

Core output: 2 volts/cm

Strobe: 5 volts/cm



Single ended core read/write output in reference to strobe.

Time: 200 ns/cm

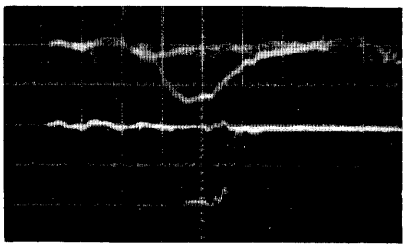
Core output: 2 volts/cm

Strobe: 5 volts/cm

Figure 5-1 Sense Amplifier Waveforms

For a 1, these show a half-sinusoid pulse of either polarity, approximately 350 to 400 MS long at its base, base, and superimposed on noise that occurs at the beginning and end of the read and write current waveforms. For a 0, only the noise is present; absence of the half-sinusoid indicates that the selected core has not changed state.

To check the slice waveform, an oscilloscope is connected to pin AM on the G005 modules. The oscilloscope should show a sense amplifier slice waveform, quiescent at + .05V, which falls to -3.5V whenever the preamp waveform has sufficient amplitude. Consequently, the level is +.05V for a 0 and -3.5V approximately 200 to 300 ns for a 1. Figure 5-2 shows the sense and slice waveforms for both 1 and 0 core outputs. Both of the above single-bit faults require replacement of the entire stock.



Single ended core with relationship to slice.

Time: 100 ns/cm
2 volts/cm

Figure 5-2 Sense Amplifier Slice Waveform

A single-bit pickup error throughout memory can also be caused by a faulty inhibit-driver output section or an open inhibit winding; a drop error similarly confined results from a faulty quadrant sense-amplifier preamps or an open sense line.

Failures in this class turn up initially with a single bit failing at a particular address. Then the problem is to determine whether the failure was due to a faulty bit or a faulty address. To make this determination, some test programs should be run; memory address tests, which sometimes will show up multiple addressing which checkerboard won't; checkerboard with sense amp margins; and block transfer tests.

While running checkerboard, vary sense amp margins until a particular bit begins to fail, then continue to move the voltage and see if the addresses on which the bit fails seem to stay concentrated on a single or a very small group of addresses and if other bits begin to fail quickly as the margin is increased. If they do, then the failure probably is an address failure and not a bit failure. If, however, the failure sticks to a single bit until the margins reach the normal failing points, then the failure is almost certain to be in the sense amplifier or inhibit rather than an address failure.

On the other hand, if many bits and large numbers of consecutive addresses fail, something common to the entire memory system is probably at fault, such as drivers, selectors, power supplies, address register, quadrant selection, segment selection for inhibit, if even or odd bits are dropping or picking up.

In the case of a bit failure, first try changing the sense amplifier and inhibit driver card for that bit.

5.8 ADDRESS FAILURES

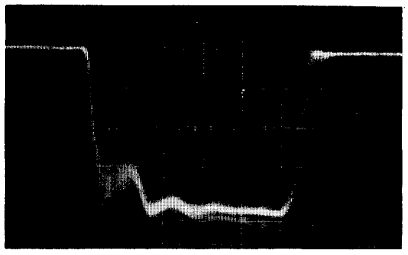
If the failure is always on one and only one address, until margins are varied up to the normal failure points, the failure must be in the stack. With a multiple address failure, the trouble is usually a balun card, G604 or G206. The fault may lie on a balun card other than the one pointed to by the address decoding points, beacuse these cards are connected in a matrix. All the cards on both selection axes should be checked by substitution.

If sense amp margins have no effect on a failure which looks like a bit failure, the failure is usually occurring in the MB flip-flop or the W102 board. If it is a single bit failure occurring only in every address of a single quadrant (determined by address bits 22 and 29) then there is most probably a wire broken; there are four sets of inhibit windings and four sets of sense amp windings to each module.

An open diode on a balun card will be pointed to by failures one every address on that selection line. A shorted or leaky diode will not be pointed to directly, as the address affected will probably appear to be all right and other addresses will appear to be failing. Be sure that you obtain a complete table of every address that fails, far enough into memory to obtain a pattern. This clue will enable you to pinpoint a particular G206 or balun (G604) failure and can be done, quickly and easily, using the checkerboard to inhibit address lines, or by halting on error and pressing CONTINUE. If the errors occur every fourth address or every 1000th address on the Y lines, exercise extreme caution. The error may only be in a single address, but it may be propagated throughout memory because the process by which the program loads the pattern. Address test may help here, but simple deposit examine may also be helpful.

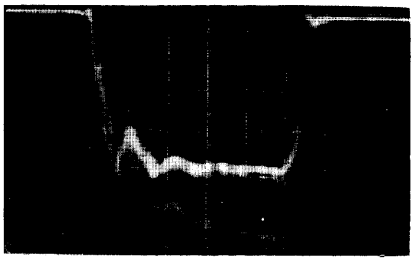
If the address failure pattern is random rather than patterned, there is a chance that the memory common driver G212 may be bad; or, if the failure has occurred throughout the entire word, then it probably is a control failure; i.e. the timing is off or there is a power supply problem. In this case, the bits may not be scattered evenly through the word. It might also be an address bit problem since the MAs are used to feed the G206. If this is the case, expect the failures to show up readily when that bit is inhibited on the checkerboard test.

The best way to test for problems on any of the modules, that have been mentioned so far, is by substitution. A good point at which to start checking bit multi-address failures is to take a current probe and check all the read and write currents while cycling through checkerboard using the test stubs provided for. Open circuits, either in the drivers or selector, will show up as complete absence of current. Photographs provided in Figure 5-3 show waveforms for some typical patterns. These are read and write current waveforms, sense amplifier waveforms and inhibit current waveforms. Generally speaking, the sense amplifier outputs, when looked at differentially, should be approximately 4V peak-to-peak.



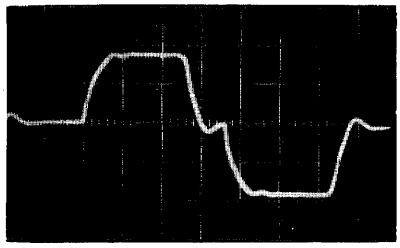
X read select current measured between pins 1F04S and 1F06M white bus.

Time: 100 ma/cm
100 ns/cm



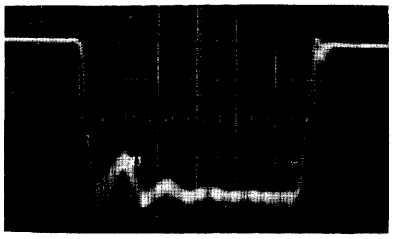
X read drive current measured between pins 1F06U and 1F09M white bus.

Time: 100 ma/cm
100 ns/cm



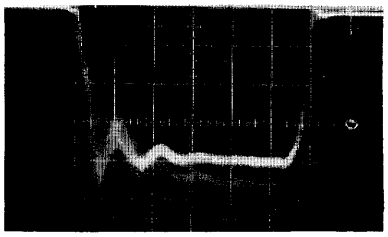
X read-write total current through the stack

Time: 200 ma/cm
200 ns/cm



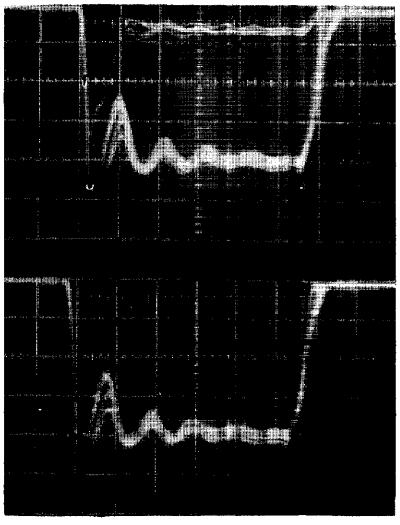
X write select current measured between pins 1F04V and 1F06N brown bus.

Time: 100 ma/cm
100 ns/cm



X write drive current measured between pins 1F06V and 1F09R brown bus.

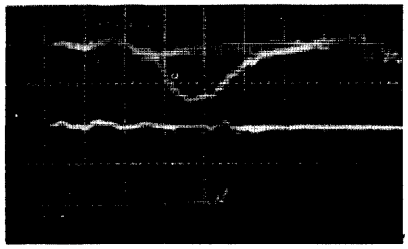
Time: 100 ma/cm
100 ns/cm



Open address on X write drive current.

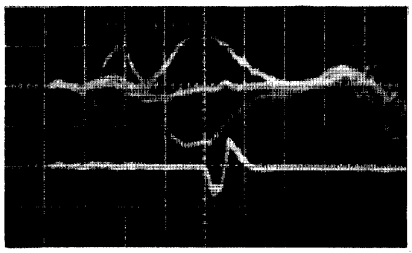
Normal X write drive current

Figure 5-3 Type 164 Memory System Waveforms (Sheet 1 of 2)



Single ended core with relationship to slice.

Time: 100 ns/cm
2 volts/cm

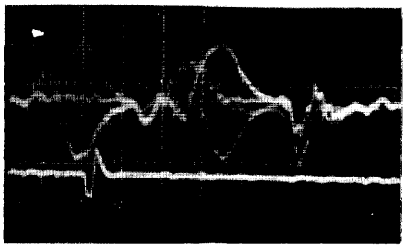


Single ended core read output in reference to strobe.

Time: 100 ns/cm

Core output: 2 volts/cm

Strobe: 5 volts/cm

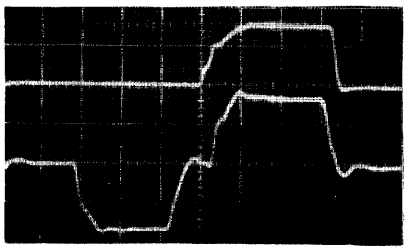


Single ended core read/write output in reference to strobe.

Time: 200 ns/cm

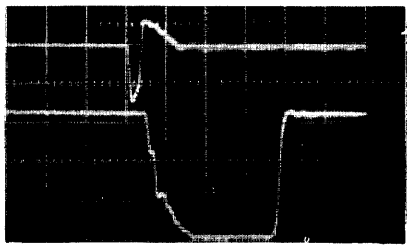
Core output: 2 volts/cm

Strobe: 5 volts/cm



Total inhibit current with relationship to total read/write current.

Time: 200 ma/cm
200 ns/cm



Total inhibit current measured from the resistor panel reference, with time pulse 3.

Time: 200 ns/cm
100 ma/cm

Time Pulse 3: 2 volts/cm

Figure 5-3 Type 164 Memory System Waveforms (Sheet 2 of 2)

5.9 TUNE-UP PROCEDURE

This procedure should only have to be followed, if the stack or power supply regulator cards have been replaced or if someone has misadjusted all the knobs on the power supply; under no circumstances should the balance pots on the sense amplifier cards be touched out in the field.

On the back of the power supply, there is a label giving read-write voltage, inhibit voltage, slice voltage, first and second stage clamp voltages; the memory was set to these when voltages left the plant. A good starting point when reregulating is to set the voltages to these values. Inhibit voltage should be 52-55V, read-write voltage should be 63-65V, and slice should be set at +5.5 to 5.9V at pin AV of the sense amplifiers. The read-write should never be over 70V or below 60V; the inhibit should never be above 60 or below 50V. If these limits are exceeded, the memory simply will run with errors; no damage will be done.

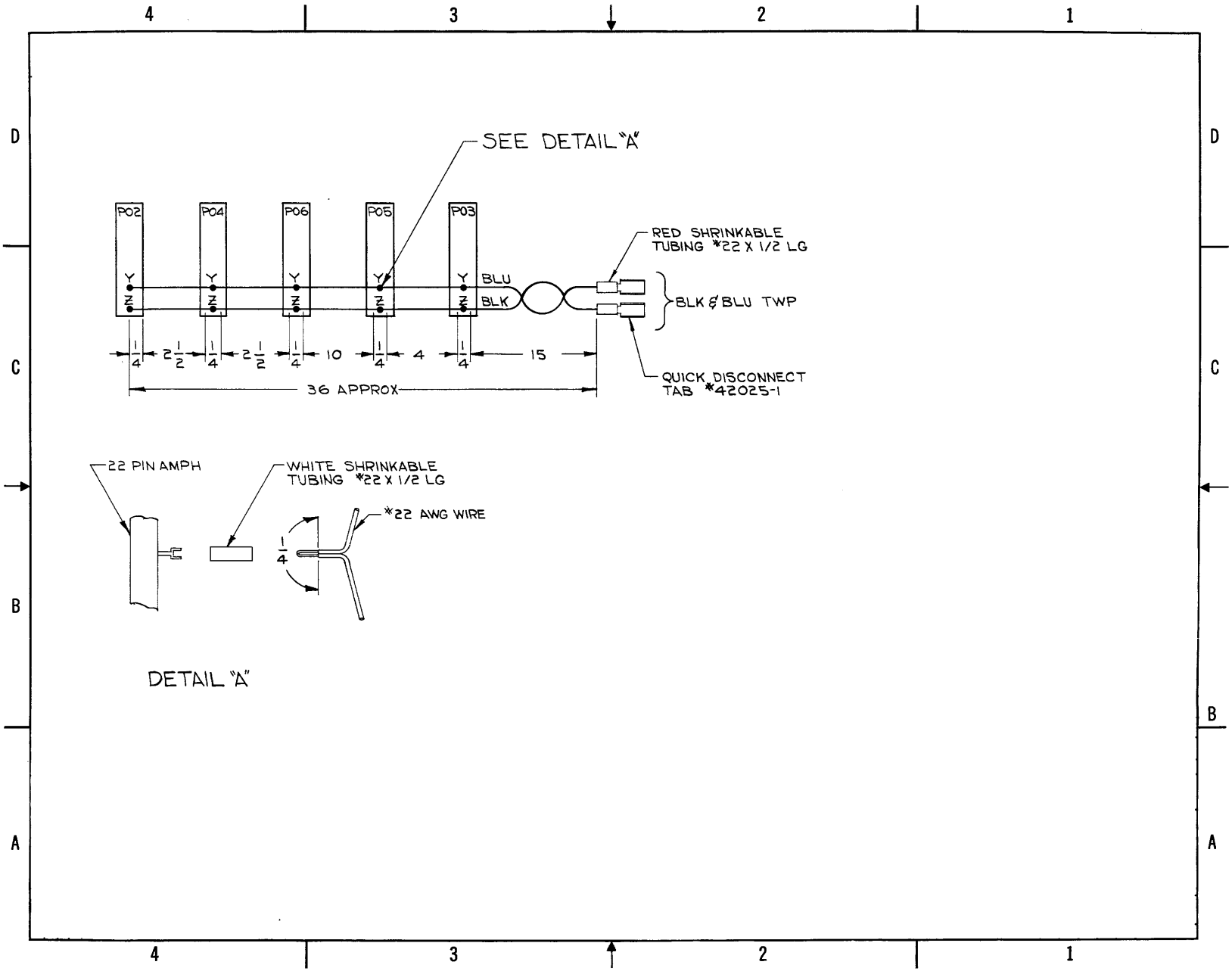
When the voltages are set as described, start moving the sense amp margins using checkerboard. Adjust the read-write voltage for maximum positive margin, and the inhibit voltage for maximum negative margin. Don't worry if the margins are not balanced as long as failure points can be detected on both ends. If failure points cannot be detected, adjust the slice voltage to move the margins so that failure points can be detected and continue tuning. The last thing to be done is adjusting the slice for balanced positive and negative margins. Remember that by using this procedure margins are being optimized for a checkerboard pattern; other patterns will have different margins. Next, look at the slice output pin AM on any sense amp, while simultaneously looking at strobe. Strobe should be about in the middle of the sliced output; if it is offset, the margin range will be too narrow.

CHAPTER 6
ENGINEERING DRAWINGS

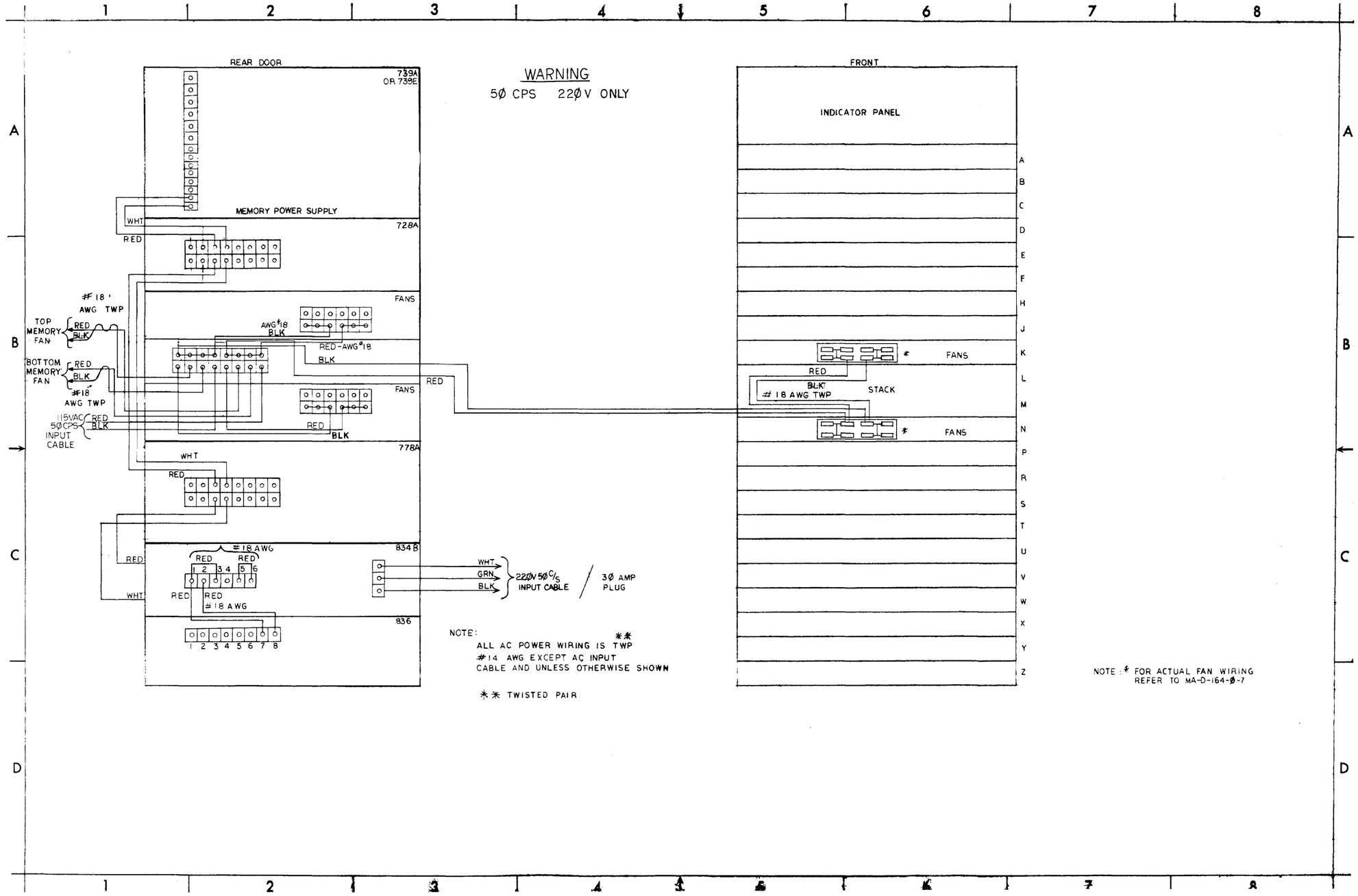
This chapter contains copies of all of the engineering drawings and replacement schematics necessary to understand and maintain the Type 164 Memory System. The engineering drawings in this chapter are in addition to the complete set of full-size drawings supplied with each system; should any discrepancy exist between the drawings in this manual and those supplied with the equipment, assume the drawings supplied with the equipment are correct. The drawings supplied should be used by maintenance personnel when working on the memory, because they show variations peculiar to an individual installation. Replacement schematics are furnished for test and maintenance purposes. The circuits on these drawings are proprietary in nature and should be treated accordingly.

The engineering drawings* appear in the order listed below.

*Standard DEC drawing terminology is used in the drawings contained in this chapter. For an explanation of DEC logic, symbology, and information on the drawings, the reader can refer to Chapter 4 of this manual and Appendix 1 of the Arithmetic Processor 166 Instruction Manual, F-67 (166).



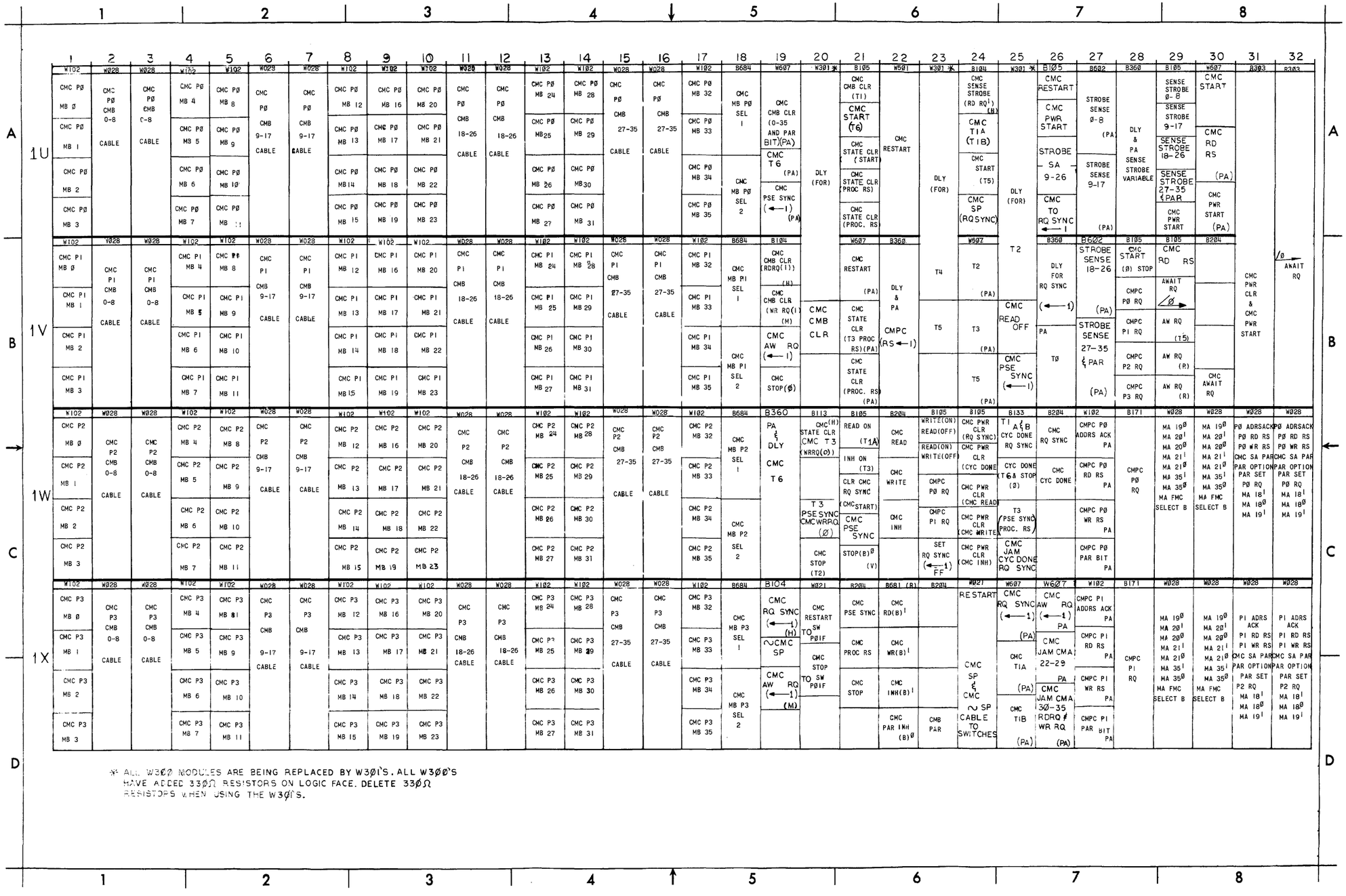
PW-C-164-0-IPW Indicator Power Wiring



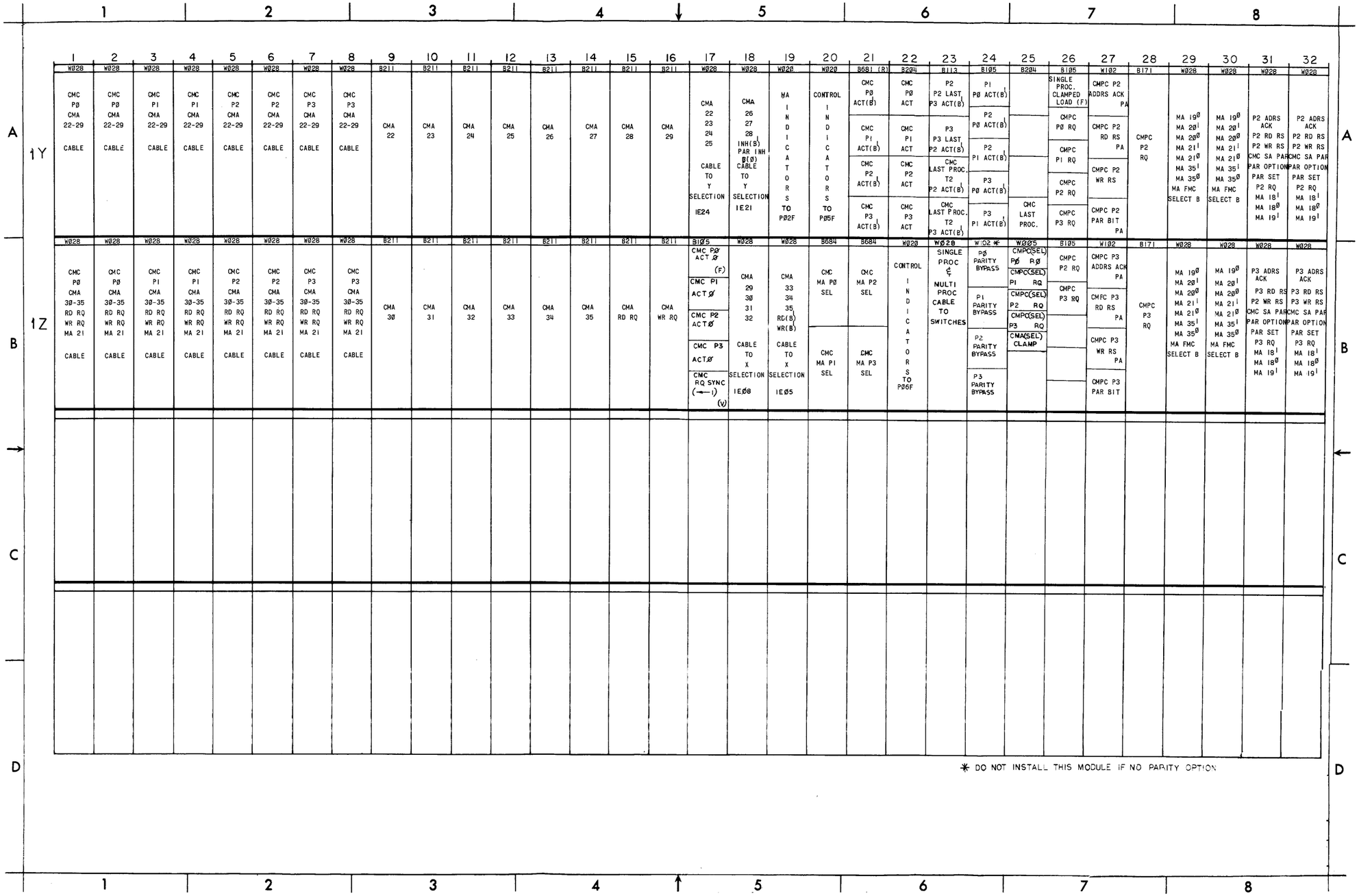
WARNING

50 CPS 220V ONLY

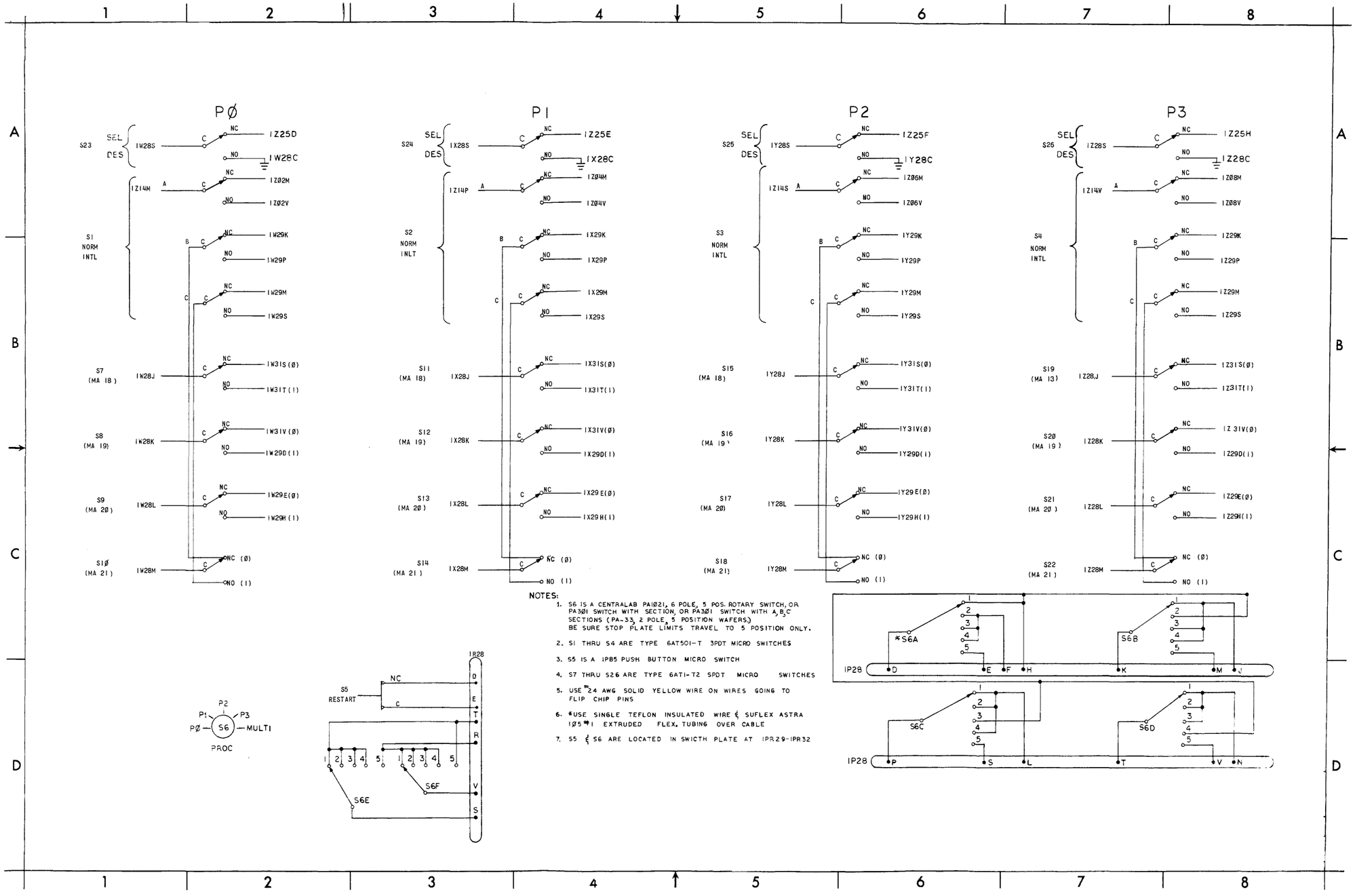
PW-D-164-0-50ACPW AC Power Wiring



UML-D-164-0-11 Utilization Module List Control and Interface (Part 1). Rev. C



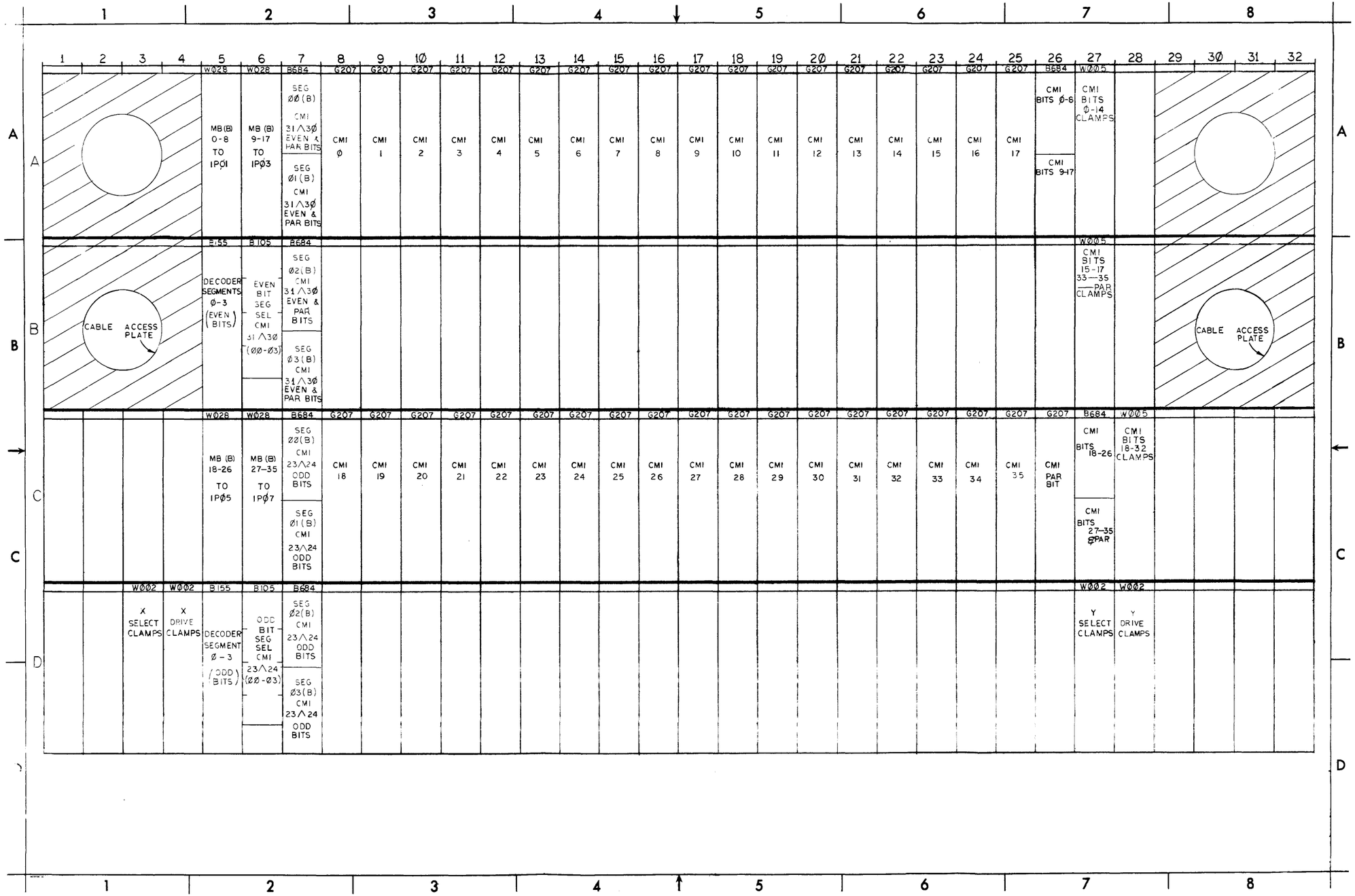
UML-D-164-0-11 Utilization Module List Control and Interface (Part 2). Rev. C



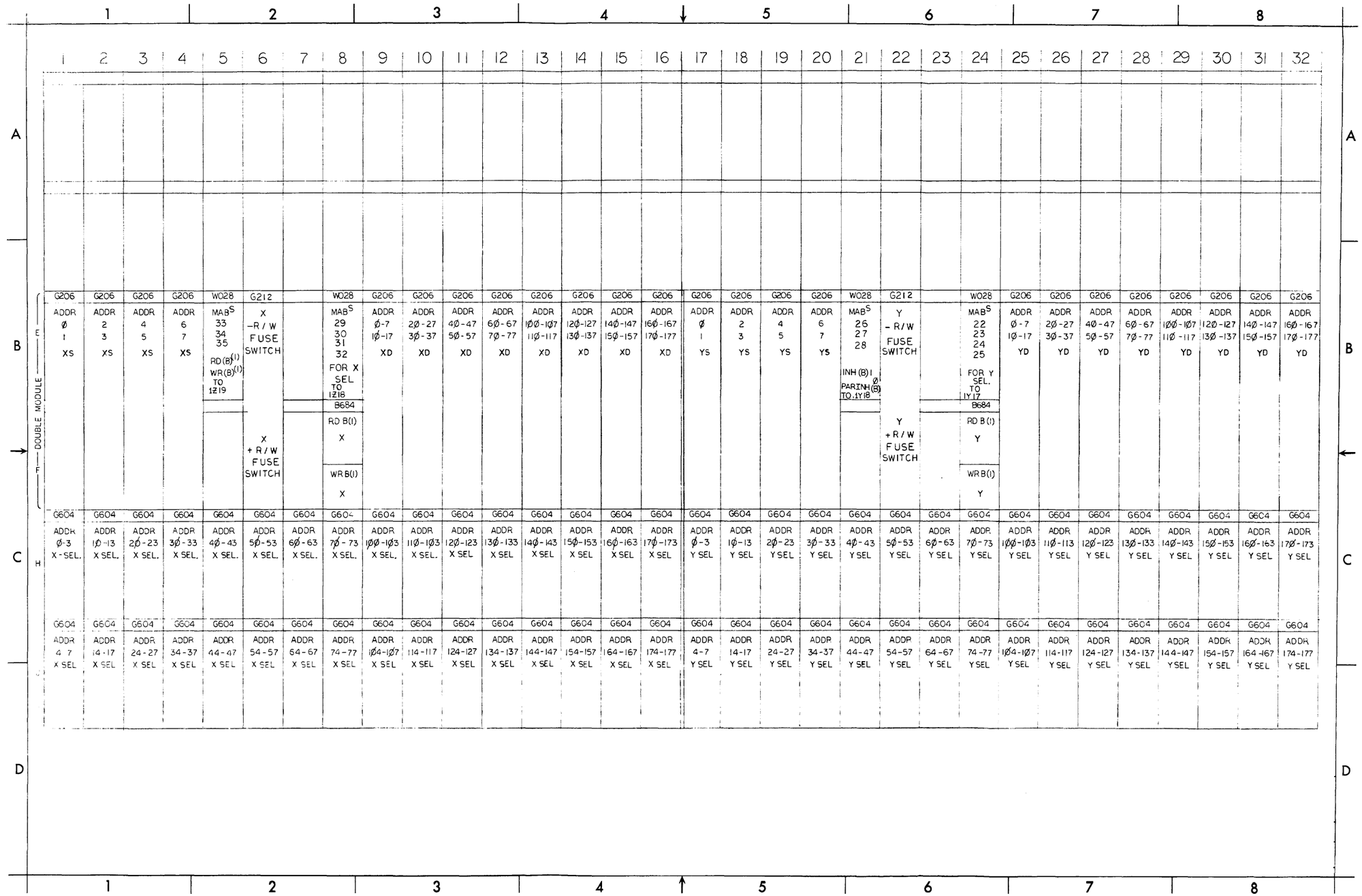
NOTES:

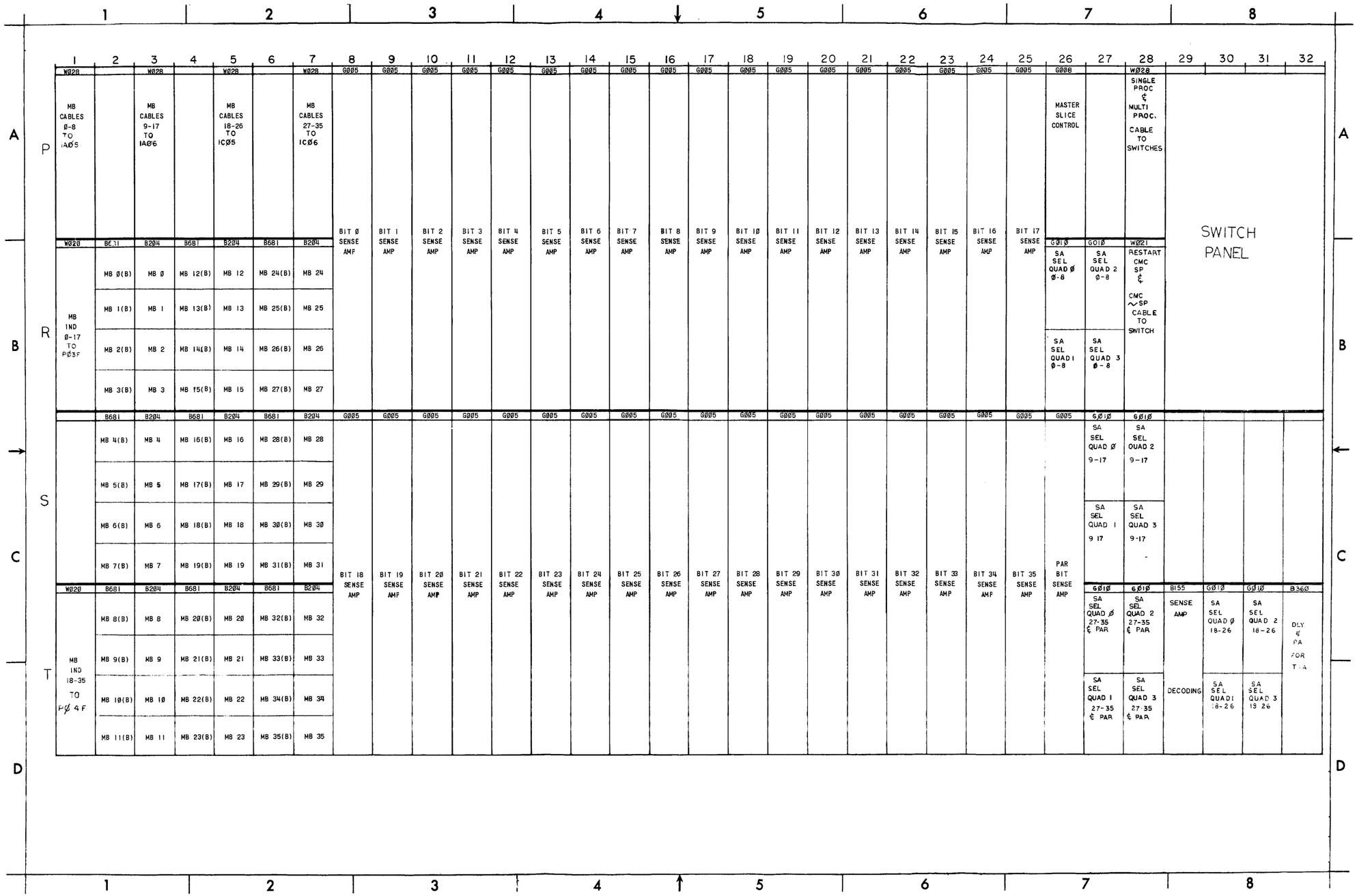
1. S6 IS A CENTRALAB PA1Ø21, 6 POLE, 5 POS. ROTARY SWITCH, OR PA3Ø1 SWITCH WITH SECTION, OR PA3Ø1 SWITCH WITH A,B,C SECTIONS (PA-33, 2 POLE, 5 POSITION WAFERS.) BE SURE STOP PLATE LIMITS TRAVEL TO 5 POSITION ONLY.
2. S1 THRU S4 ARE TYPE 6AT5Ø1-T 3PDT MICRO SWITCHES
3. S5 IS A 1PB5 PUSH BUTTON MICRO SWITCH
4. S7 THRU S26 ARE TYPE 6ATI-T2 SPDT MICRO SWITCHES
5. USE #24 AWG SOLID YELLOW WIRE ON WIRES GOING TO FLIP CHIP PINS
6. *USE SINGLE TEFLON INSULATED WIRE & SUFLEX ASTRA 1Ø5#1 EXTRUDED FLEX. TUBING OVER CABLE
7. S5 & S6 ARE LOCATED IN SWICTH PLATE AT IPR29-IPR32

WD-D-164-0-TSWD Toggle Switch Wiring Diagram for 164 Memory. Rev. A

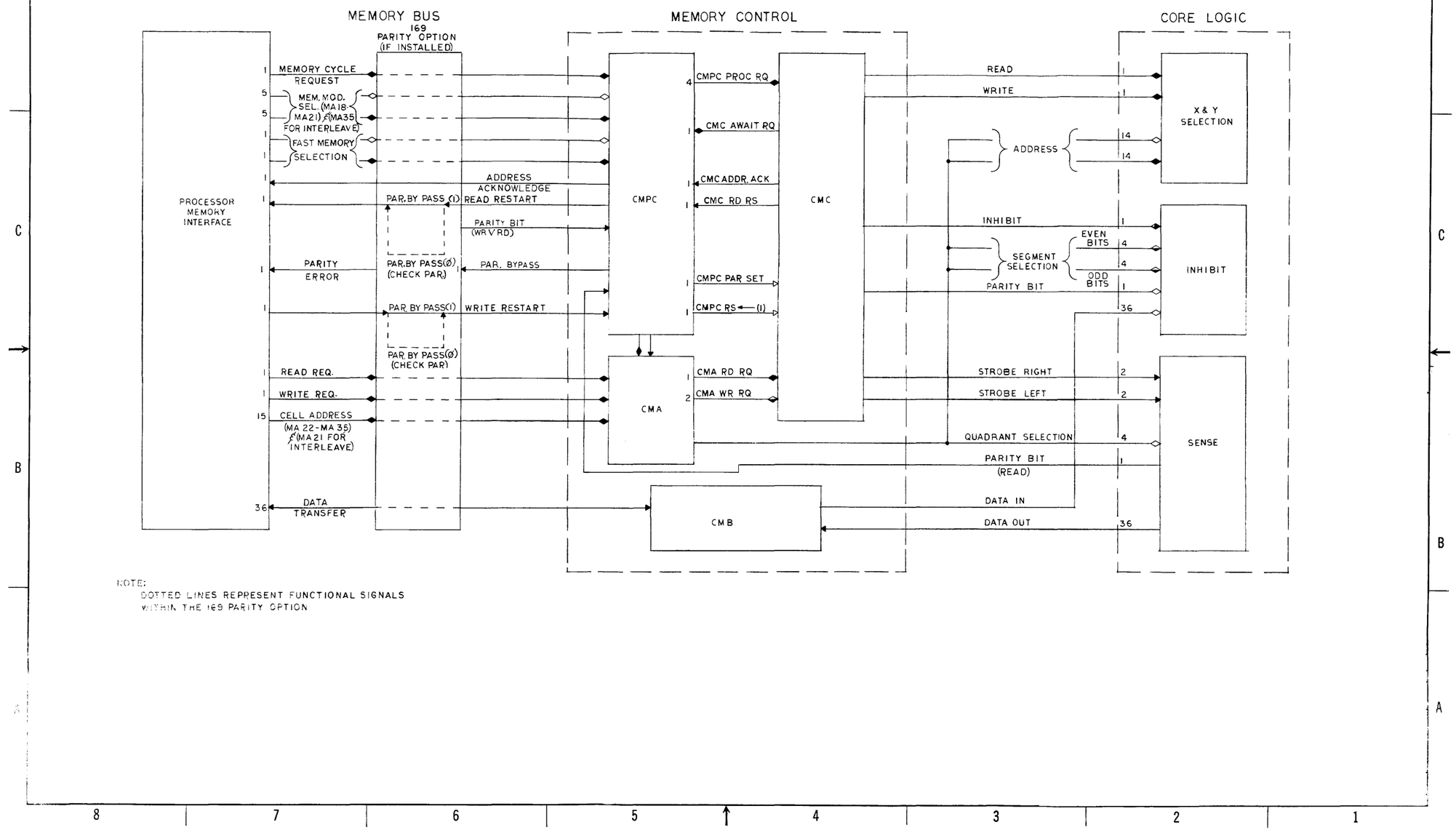


UML-D-164-0-UML-1 Utilization Module
List Inhibit


DOUBLE MODULE
E
F
H
1 2 3 4 5 6 7 8
A B C D
G206 ADDR Ø 1 XS
G206 ADDR 2 3 XS
G206 ADDR 4 5 XS
G206 ADDR 6 7 XS
W028 MAB S 33 34 35 RD(B)(1) WR(B)(1) TO 1Z19
G212 X -R/W FUSE SWITCH
X +R/W FUSE SWITCH
W028 MAB S 29 30 31 32 FOR X SEL TO 1Z18
B684
RD B(1) X
WR B(1) X
G206 ADDR Ø-7 1Ø-17 XD
G206 ADDR 2Ø-27 3Ø-37 XD
G206 ADDR 4Ø-47 5Ø-57 XD
G206 ADDR 6Ø-67 7Ø-77 XD
G206 ADDR 1ØØ-1Ø7 11Ø-117 XD
G206 ADDR 12Ø-127 13Ø-137 XD
G206 ADDR 14Ø-147 15Ø-157 XD
G206 ADDR 16Ø-167 17Ø-177 XD
G206 ADDR Ø 1 YS
G206 ADDR 2 3 YS
G206 ADDR 4 5 YS
G206 ADDR 6 7 YS
W028 MAB S 26 27 28 INH(B) 1 PARINH(B) Ø TO 1Y18
G212 Y -R/W FUSE SWITCH
Y +R/W FUSE SWITCH
W028 MAB S 22 23 24 25 FOR Y SEL. TO 1Y17
B684
RD B(1) Y
WR B(1) Y
G206 ADDR Ø-7 1Ø-17 YD
G206 ADDR 2Ø-27 3Ø-37 YD
G206 ADDR 4Ø-47 5Ø-57 YD
G206 ADDR 6Ø-67 7Ø-77 YD
G206 ADDR 1ØØ-1Ø7 11Ø-117 YD
G206 ADDR 12Ø-127 13Ø-137 YD
G206 ADDR 14Ø-147 15Ø-157 YD
G206 ADDR 16Ø-167 17Ø-177 YD
G604 ADDR Ø-3 X-SEL.
G604 ADDR 1Ø-13 X SEL.
G604 ADDR 2Ø-23 X SEL.
G604 ADDR 3Ø-33 X SEL.
G604 ADDR 4Ø-43 X SEL.
G604 ADDR 5Ø-53 X SEL.
G604 ADDR 6Ø-63 X SEL.
G604 ADDR 7Ø-73 X SEL.
G604 ADDR 1ØØ-1Ø3 X SEL.
G604 ADDR 11Ø-113 X SEL.
G604 ADDR 12Ø-123 X SEL
G604 ADDR 13Ø-133 X SEL
G604 ADDR 14Ø-143 X SEL
G604 ADDR 15Ø-153 X SEL
G604 ADDR 16Ø-163 X SEL
G604 ADDR 17Ø-173 X SEL
G604 ADDR Ø-3 Y SEL
G604 ADDR 1Ø-13 Y SEL
G604 ADDR 2Ø-23 Y SEL
G604 ADDR 3Ø-33 Y SEL
G604 ADDR 4Ø-43 Y SEL
G604 ADDR 5Ø-53 Y SEL
G604 ADDR 6Ø-63 Y SEL
G604 ADDR 7Ø-73 Y SEL
G604 ADDR 1ØØ-1Ø3 Y SEL
G604 ADDR 11Ø-113 Y SEL
G604 ADDR 12Ø-123 Y SEL
G604 ADDR 13Ø-133 Y SEL
G604 ADDR 14Ø-143 Y SEL
G604 ADDR 15Ø-153 Y SEL
G604 ADDR 16Ø-163 Y SEL
G604 ADDR 17Ø-173 Y SEL
G604 ADDR 4 7 X SEL
G604 ADDR 14-17 X SEL
G604 ADDR 24-27 X SEL
G604 ADDR 34-37 X SEL
G604 ADDR 44-47 X SEL
G604 ADDR 54-57 X SEL
G604 ADDR 64-67 X SEL
G604 ADDR 74-77 X SEL
G604 ADDR 1Ø4-1Ø7 X SEL
G604 ADDR 114-117 X SEL
G604 ADDR 124-127 X SEL
G604 ADDR 134-137 X SEL
G604 ADDR 144-147 X SEL
G604 ADDR 154-157 X SEL
G604 ADDR 164-167 X SEL
G604 ADDR 174-177 X SEL
G604 ADDR 4-7 Y SEL
G604 ADDR 14-17 Y SEL
G604 ADDR 24-27 Y SEL
G604 ADDR 34-37 Y SEL
G604 ADDR 44-47 Y SEL
G604 ADDR 54-57 Y SEL
G604 ADDR 64-67 Y SEL
G604 ADDR 74-77 Y SEL
G604 ADDR 1Ø4-1Ø7 Y SEL
G604 ADDR 114-117 Y SEL
G604 ADDR 124-127 Y SEL
G604 ADDR 134-137 Y SEL
G604 ADDR 144-147 Y SEL
G604 ADDR 154-157 Y SEL
G604 ADDR 164-467 Y SEL
G604 ADDR 174-177 Y SEL

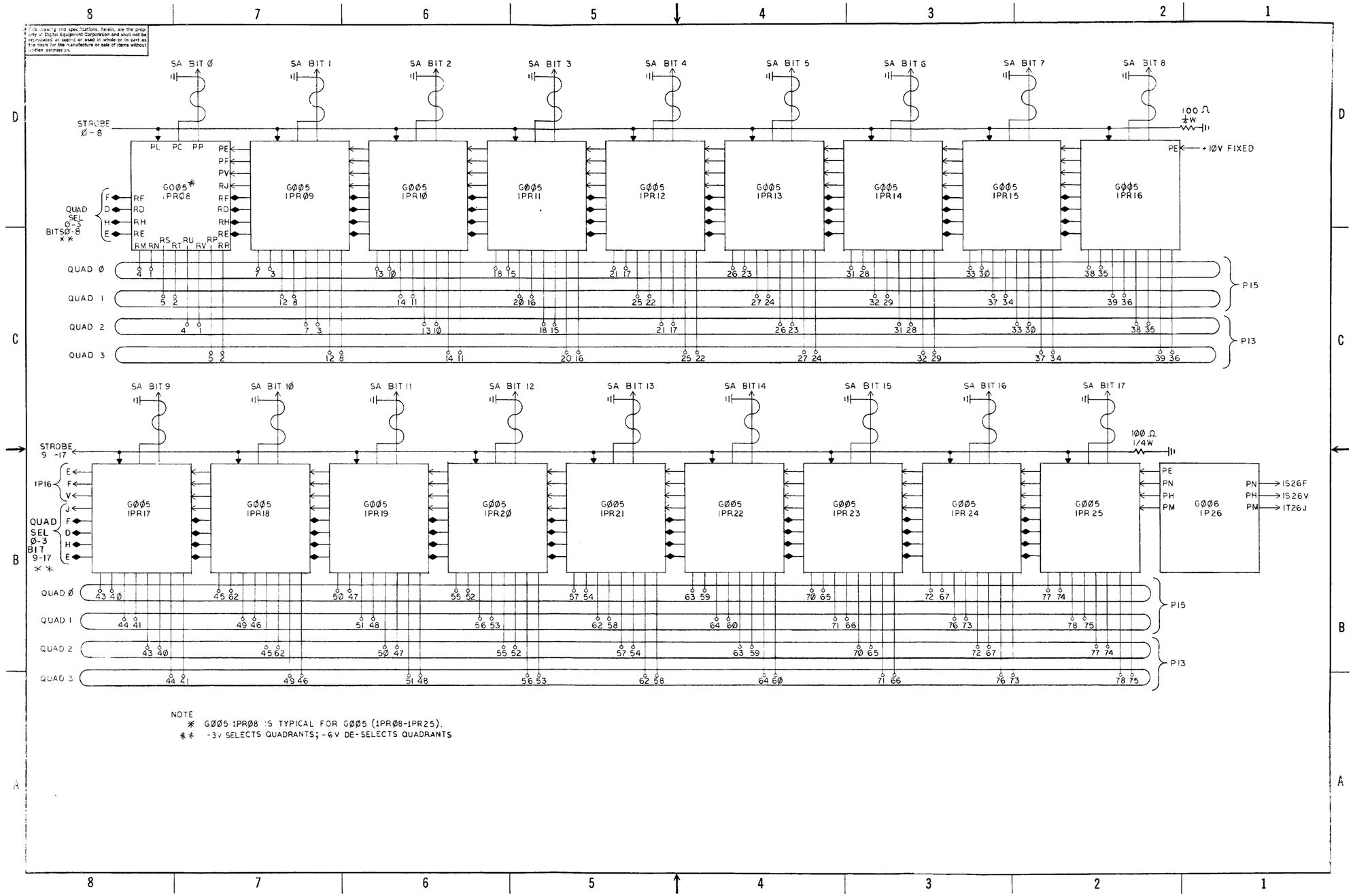


UML-D-164-0-3 Utilization Module List Sense Amps and MBs 164 Memory System. Rev. A



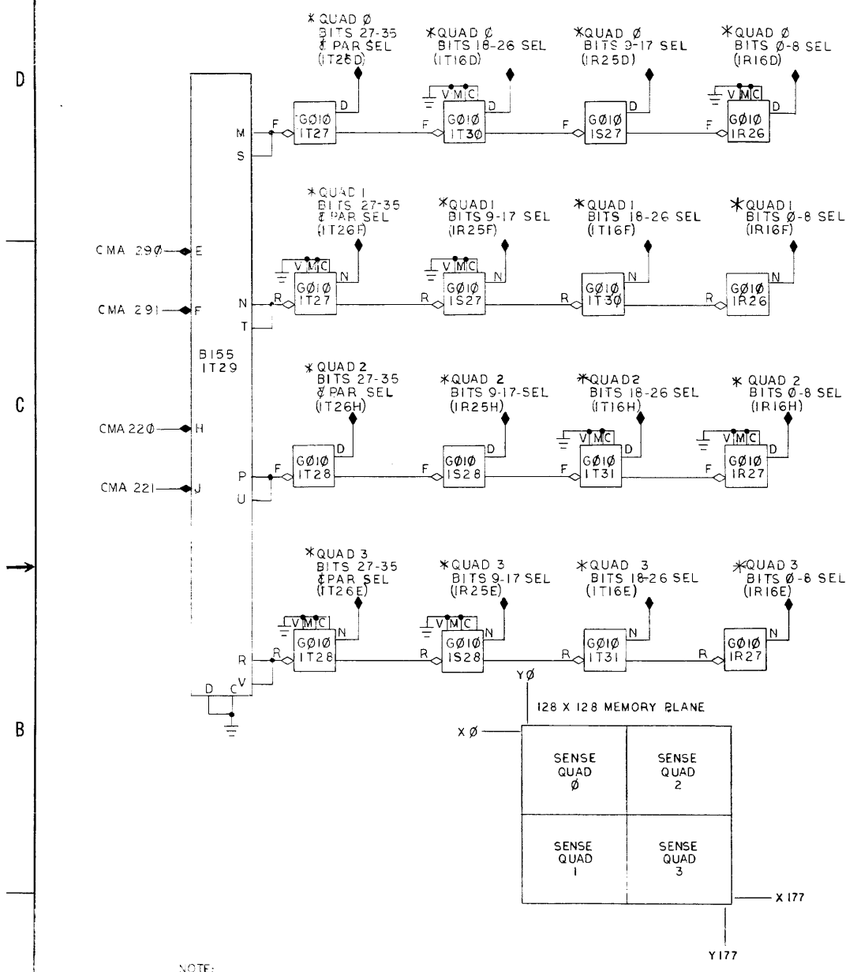
NOTE:
DOTTED LINES REPRESENT FUNCTIONAL SIGNALS
WITHIN THE 169 PARITY OPTION

SD-D-164-0-SBD System Block Diagram



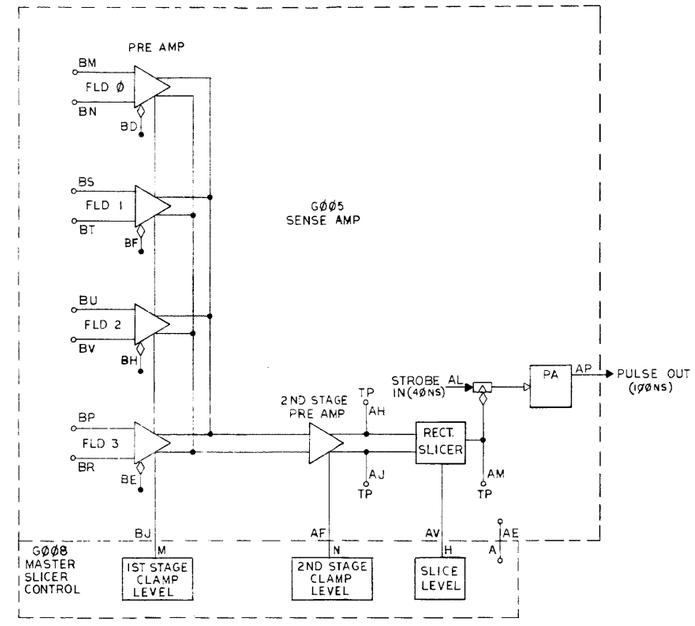
NOTE
* GØØ5 1PRØ8 IS TYPICAL FOR GØØ5 (1PRØ8-1PR25).
** -3V SELECTS QUADRANTS; -6V DE-SELECTS QUADRANTS

BS-D-164-0-MSSA-1 Sense Amplifier Bits 0-17 (Part 1). Rev. A



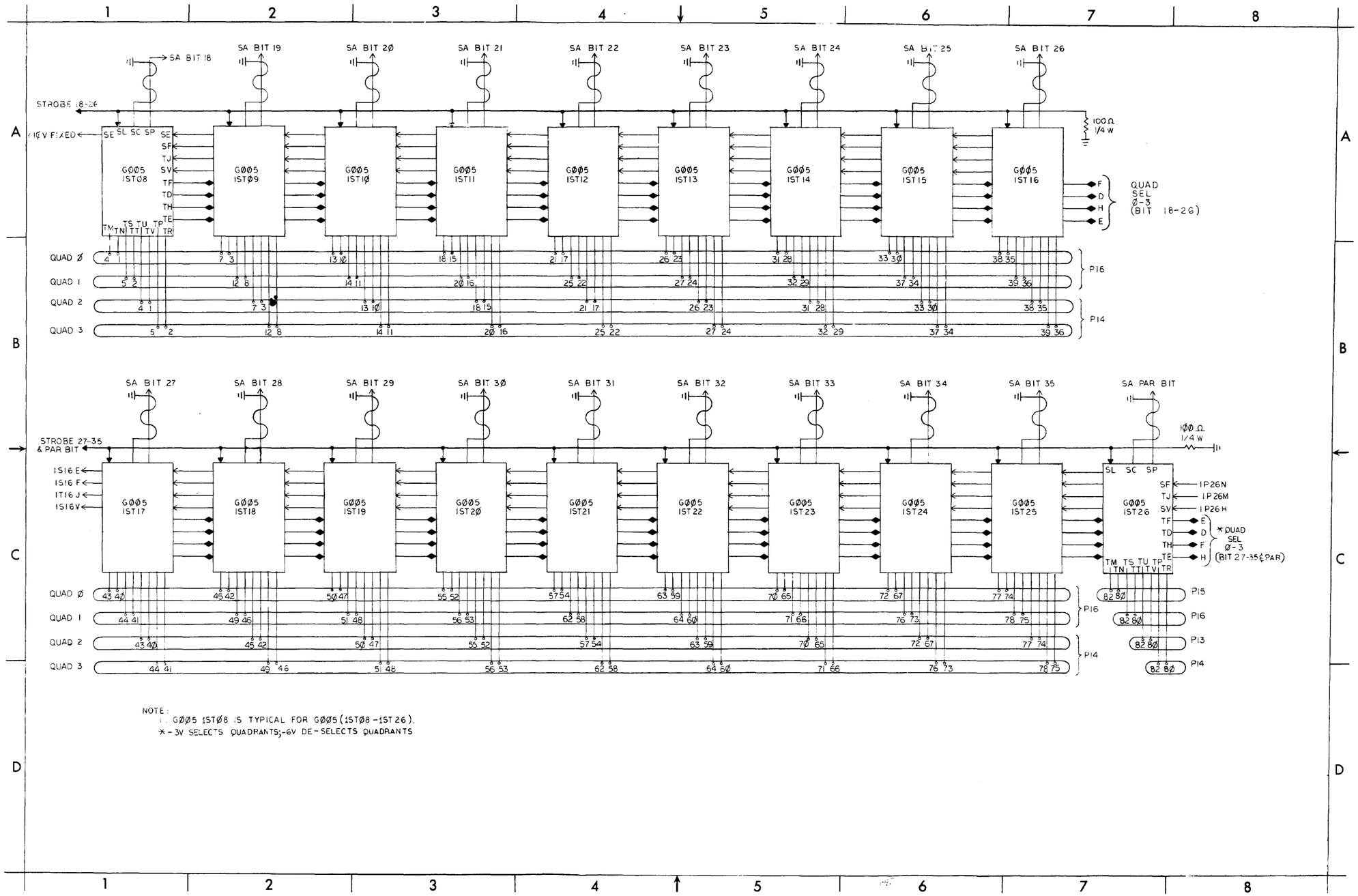
NOTE:

* QUAD SEL. LEVELS ARE -3V
DESELECT LEVELS ARE -6V.

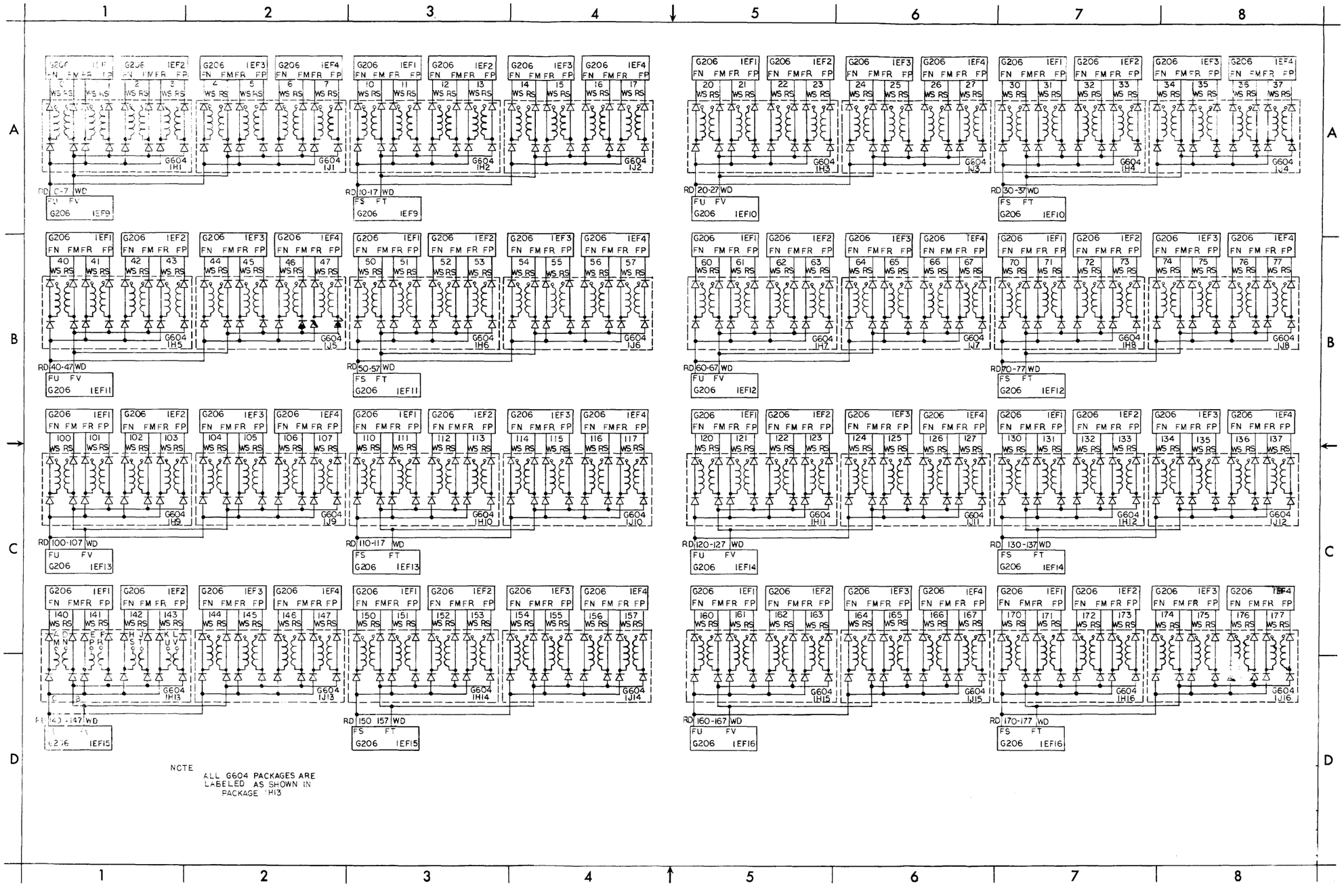


MASTER SLICER CONTROL MEASUREMENTS

	REFERENCE	MEASURE BETWEEN PINS		
*1 STAGE CLAMP	-15V	IP26B	IP26M	3.7V TO 4.0V (FIXED)
*2 STAGE CLAMP	-15V	IP26B	IP26N	8.0 V
SLICE LEVEL	+10V	IP26A	IP26H	5.5V TO 5.9V

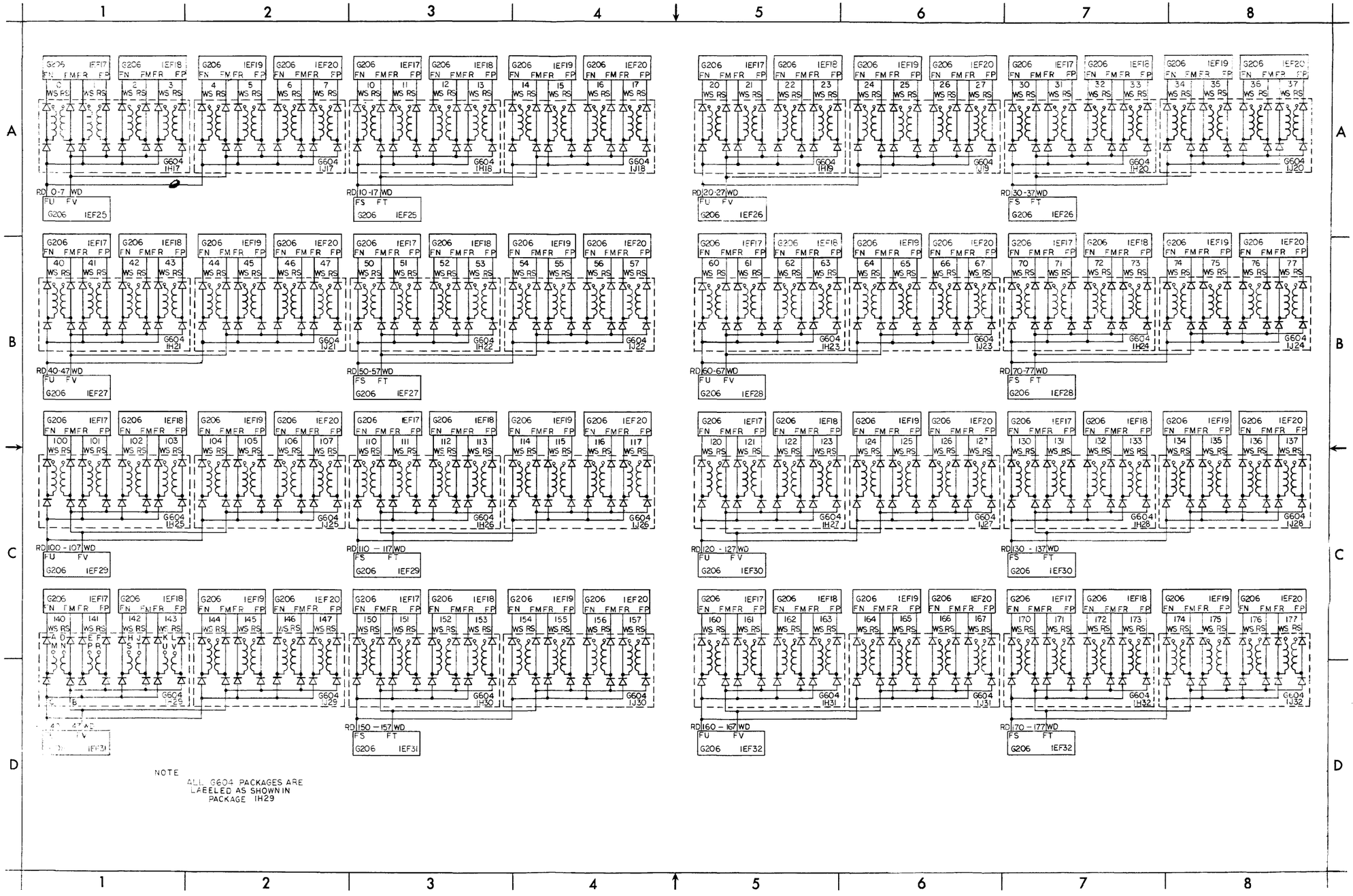


BS-D-164-0-MSSA-2 Sense Amplifier Bits 18-35 and Parity Bit



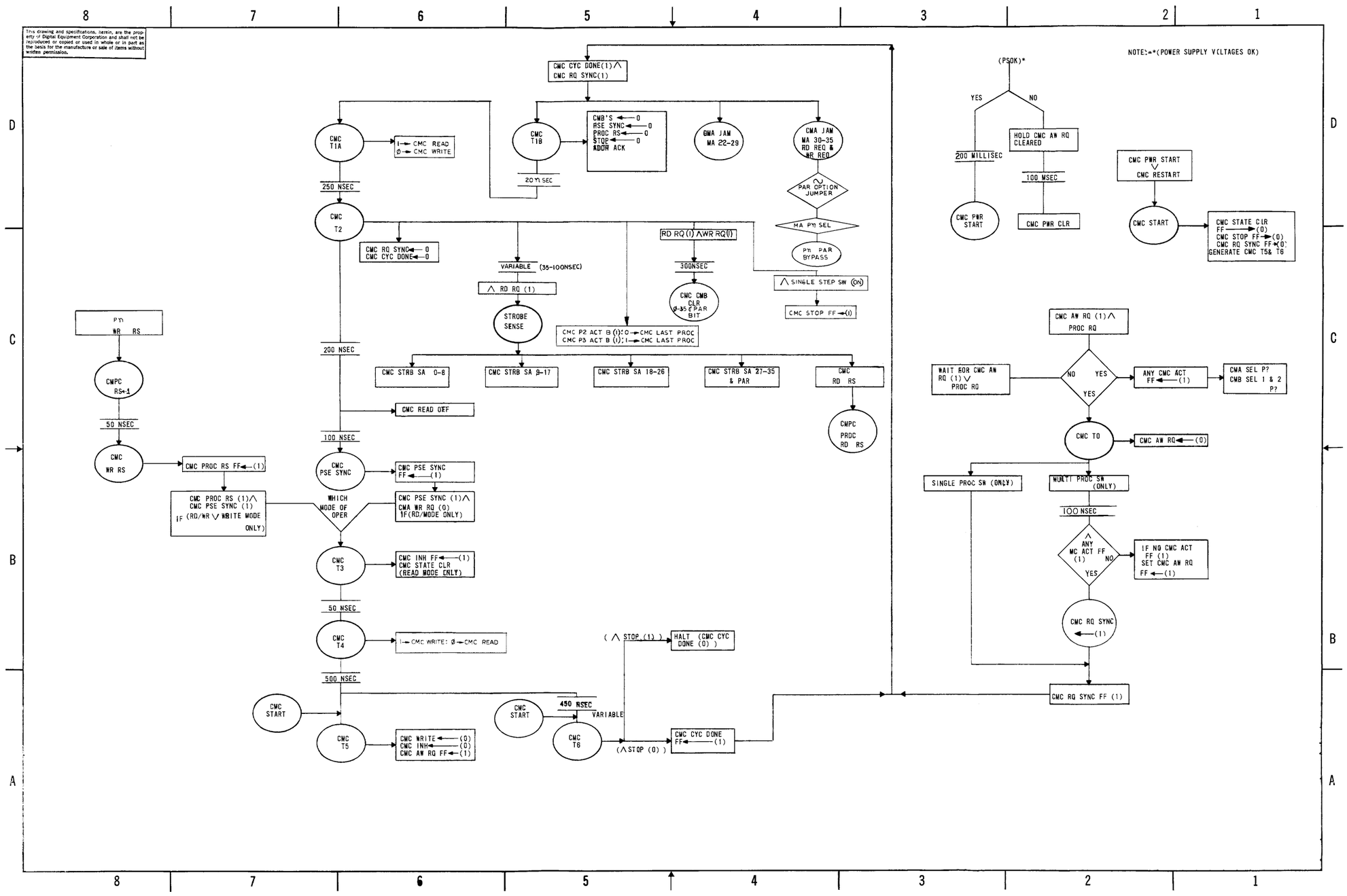
NOTE

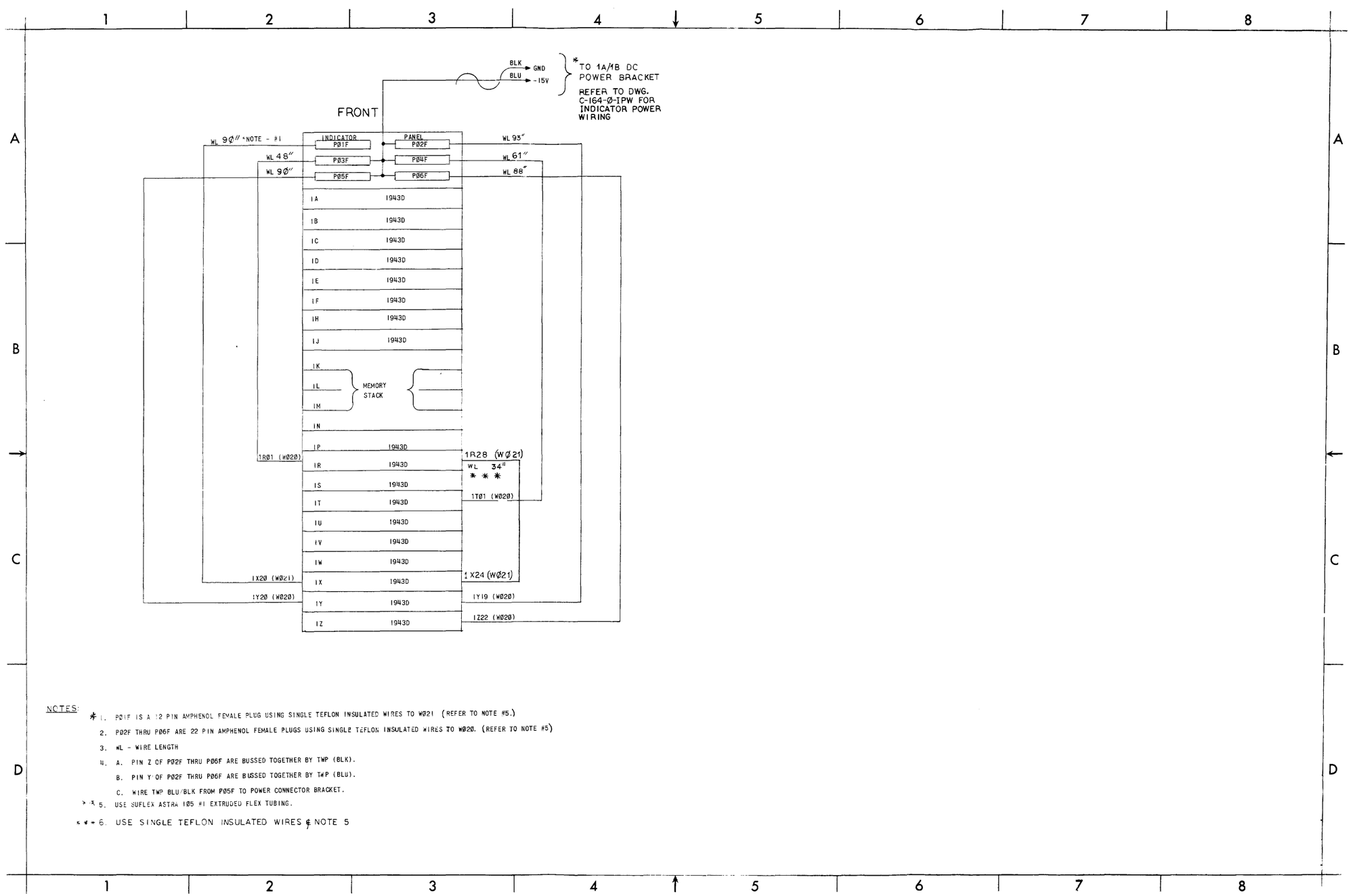
ALL G604 PACKAGES ARE LABELED AS SHOWN IN PACKAGE IH13



NOTE

ALL G604 PACKAGES ARE LABELED AS SHOWN IN PACKAGE 1H29





NOTES:

* 1. P01F IS A 12 PIN AMPHENOL FEMALE PLUG USING SINGLE TEFLON INSULATED WIRES TO W021 (REFER TO NOTE #5.)

2. P02F THRU P06F ARE 22 PIN AMPHENOL FEMALE PLUGS USING SINGLE TEFLON INSULATED WIRES TO W020. (REFER TO NOTE #5)

3. WL - WIRE LENGTH

4. A. PIN Z OF P02F THRU P06F ARE BUSSED TOGETHER BY TWP (BLK).

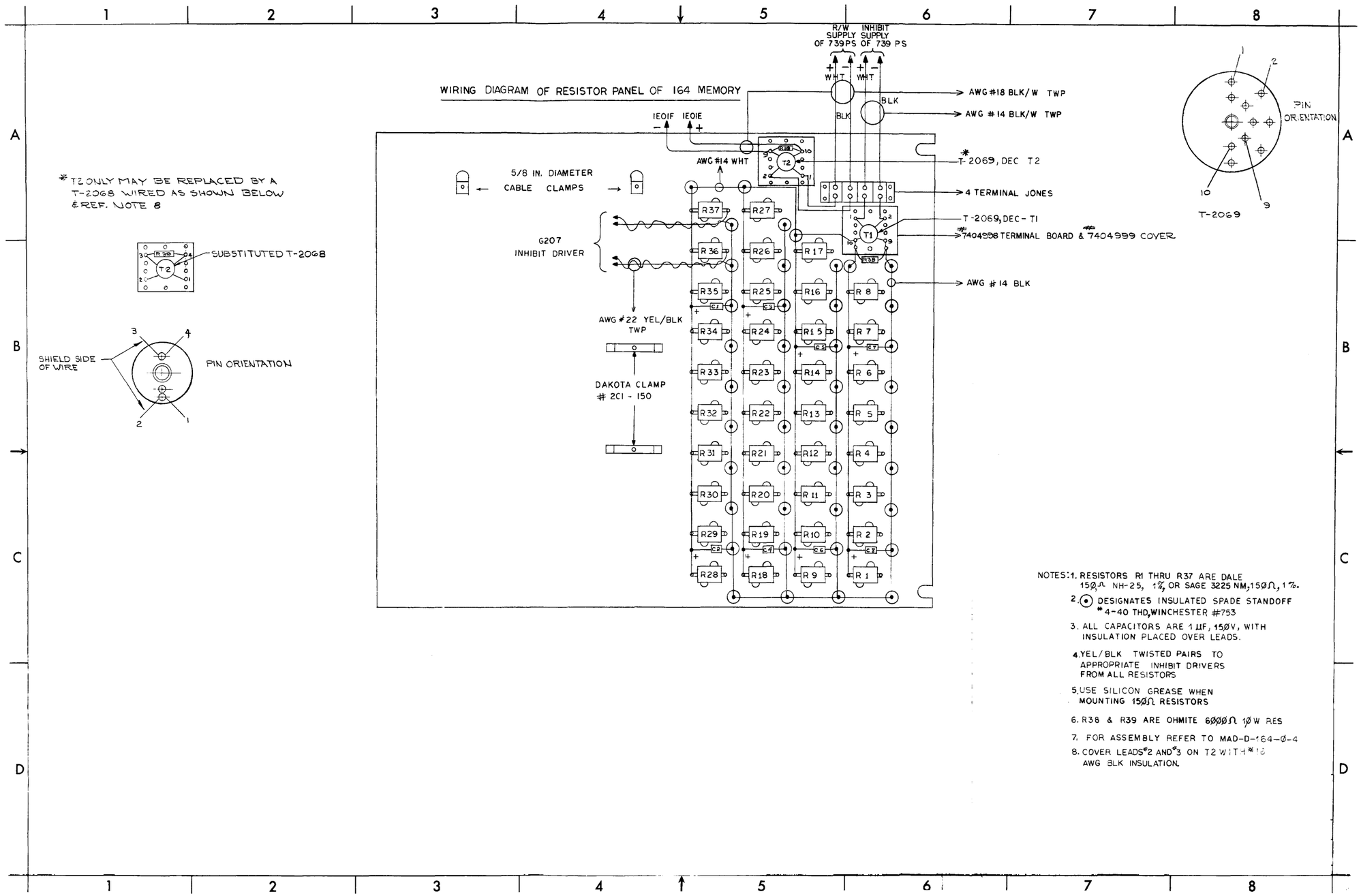
   B. PIN Y OF P02F THRU P06F ARE BUSSED TOGETHER BY TWP (BLU).

   C. WIRE TWP BLU/BLK FROM P05F TO POWER CONNECTOR BRACKET.

* * 5. USE SUFLEX ASTRA 105 #1 EXTRUDED FLEX TUBING.

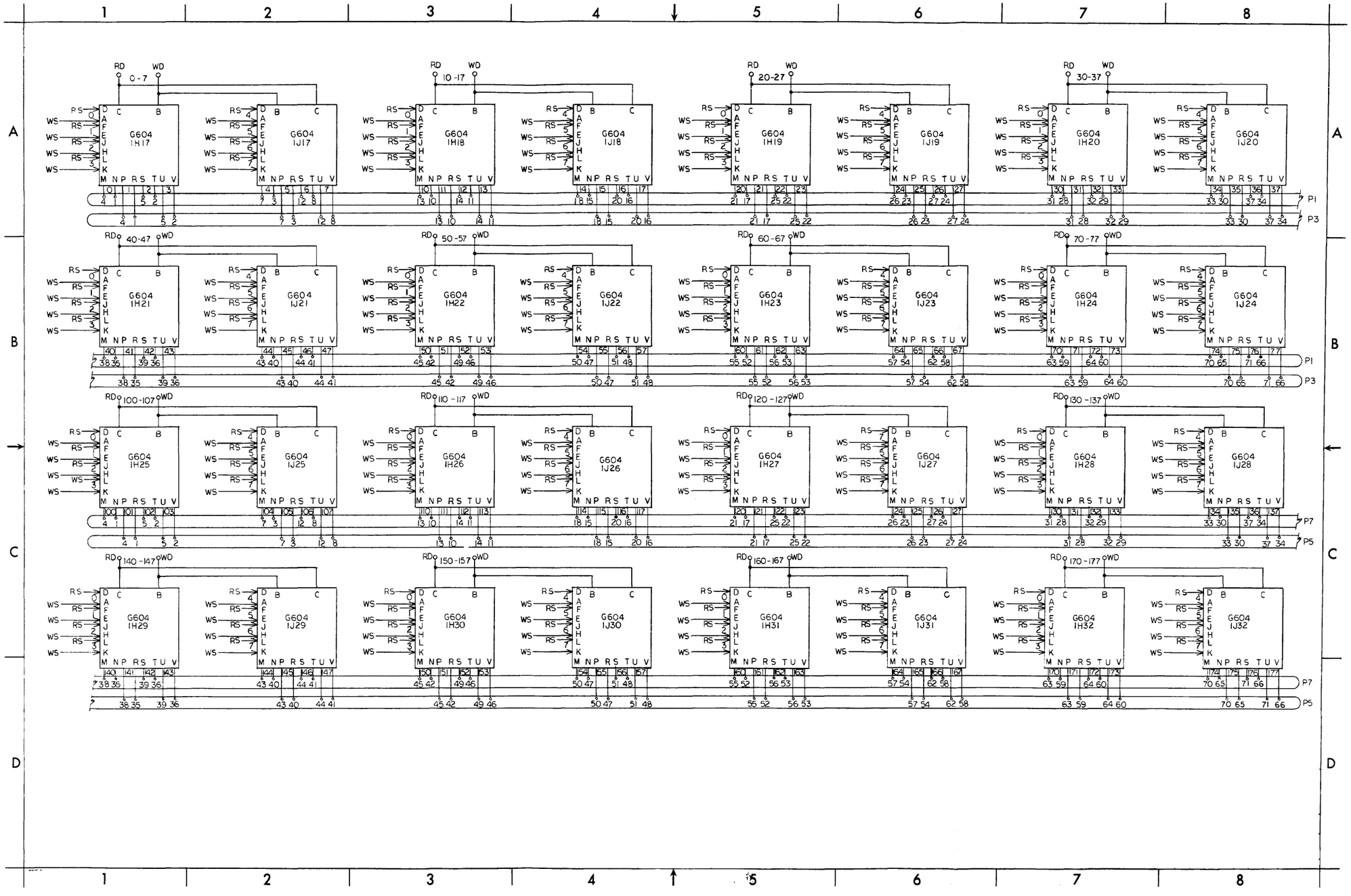
* * * 6. USE SINGLE TEFLON INSULATED WIRES & NOTE 5

CD-D-164-0-ICD Indicator Switch
Cable Diagram. Rev. A

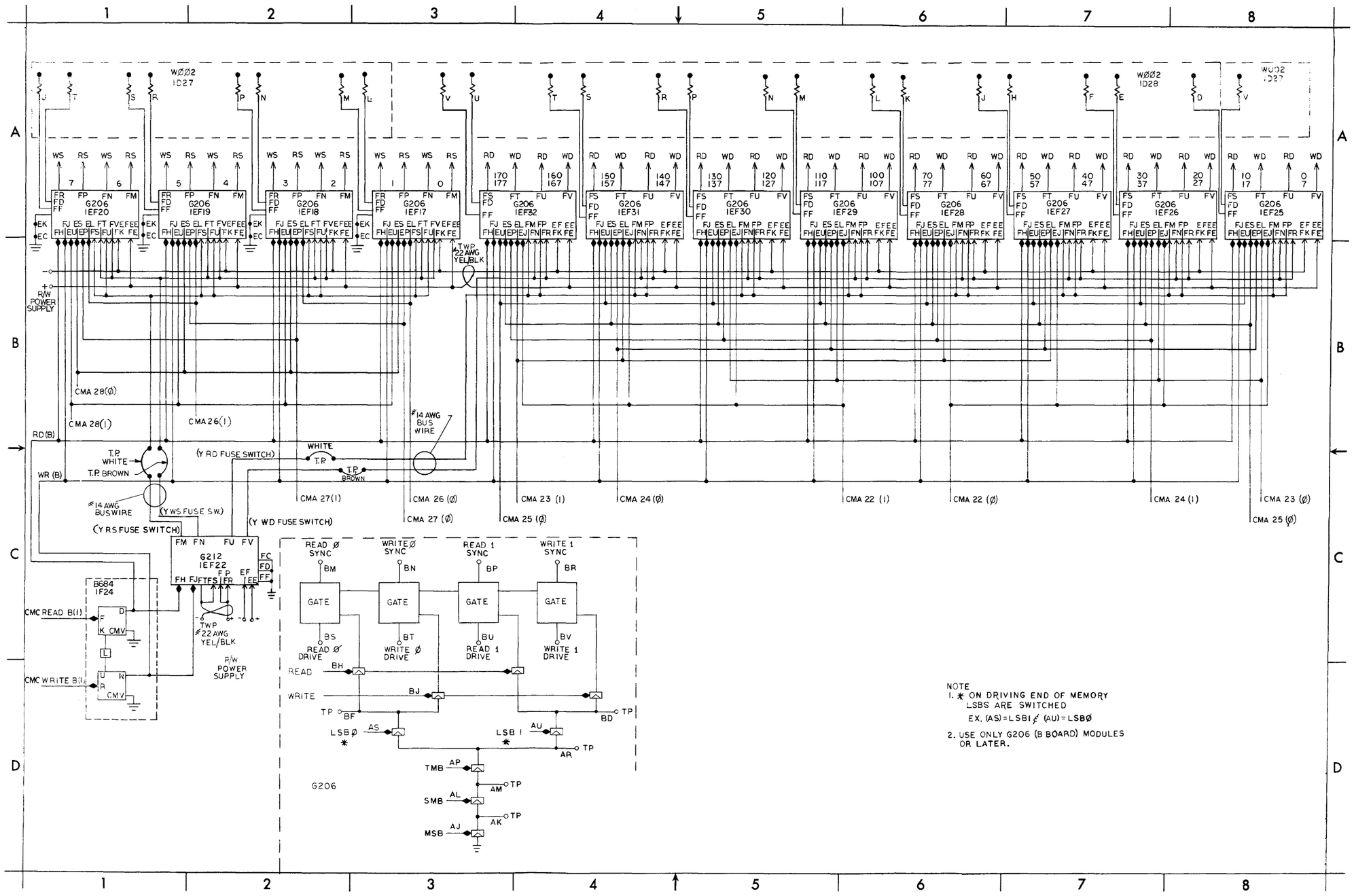


NOTES:
1. RESISTORS R1 THRU R37 ARE DALE 150Ω NH-25, 1%, OR SAGE 3225 NM, 150Ω, 1%.
2. ⊙ DESIGNATES INSULATED SPADE STANDOFF #4-40 THD, WINCHESTER #753
3. ALL CAPACITORS ARE 1 μF, 15ØV, WITH INSULATION PLACED OVER LEADS.
4. YEL/BLK TWISTED PAIRS TO APPROPRIATE INHIBIT DRIVERS FROM ALL RESISTORS
5. USE SILICON GREASE WHEN MOUNTING 15Ø Ω RESISTORS
6. R38 & R39 ARE OHMITE 6ØØØ Ω 1Ø W RES
7. FOR ASSEMBLY REFER TO MAD-D-164-Ø-4
8. COVER LEADS #2 AND #3 ON T2 WITH #18 AWG BLK INSULATION.

WD-D-164-0-IRP Inhibit Resistor Panel
164 Memory. Rev. A

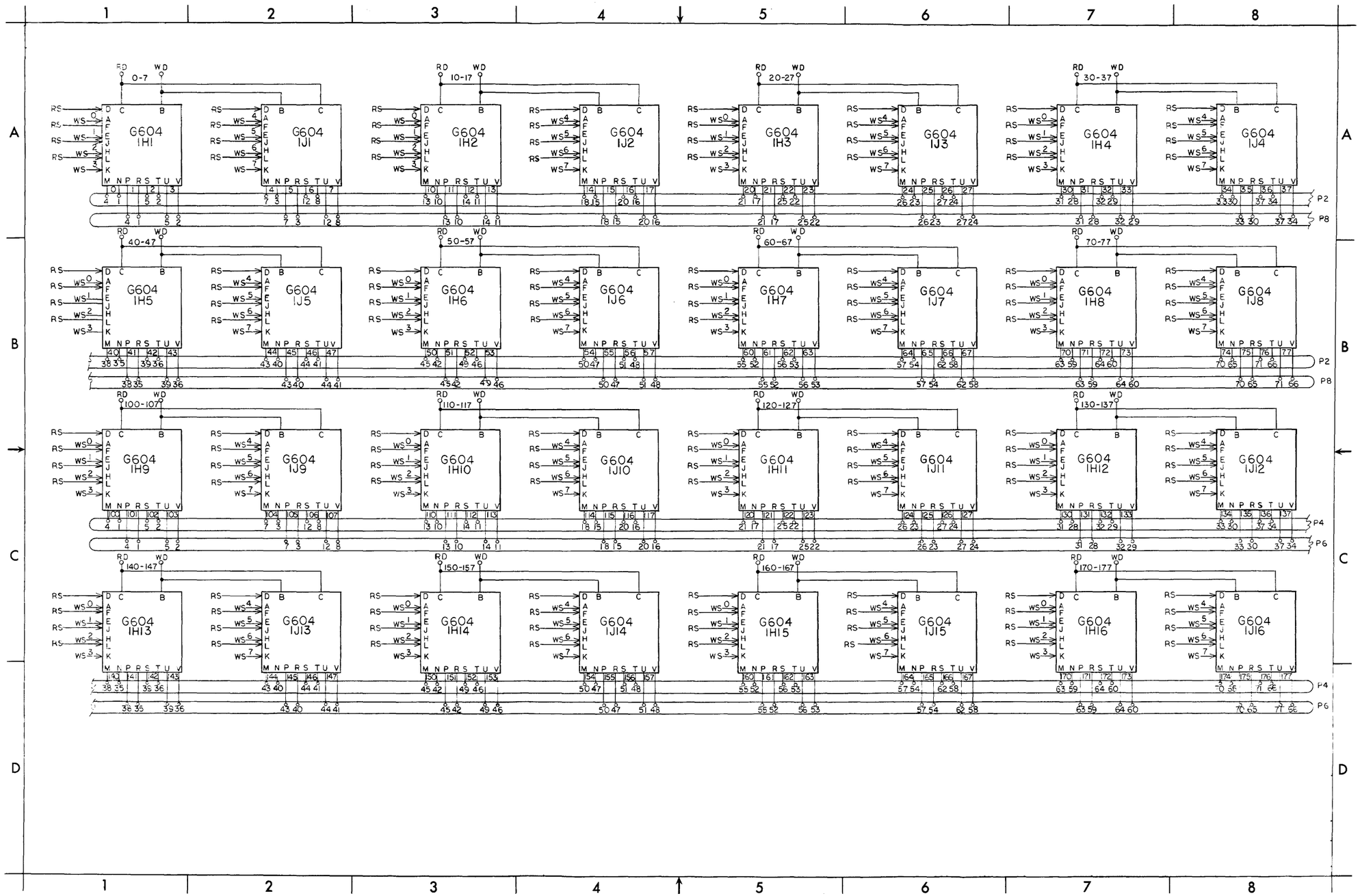


BS-D-164-0-DMSY Diode Matrix
Y-Selection

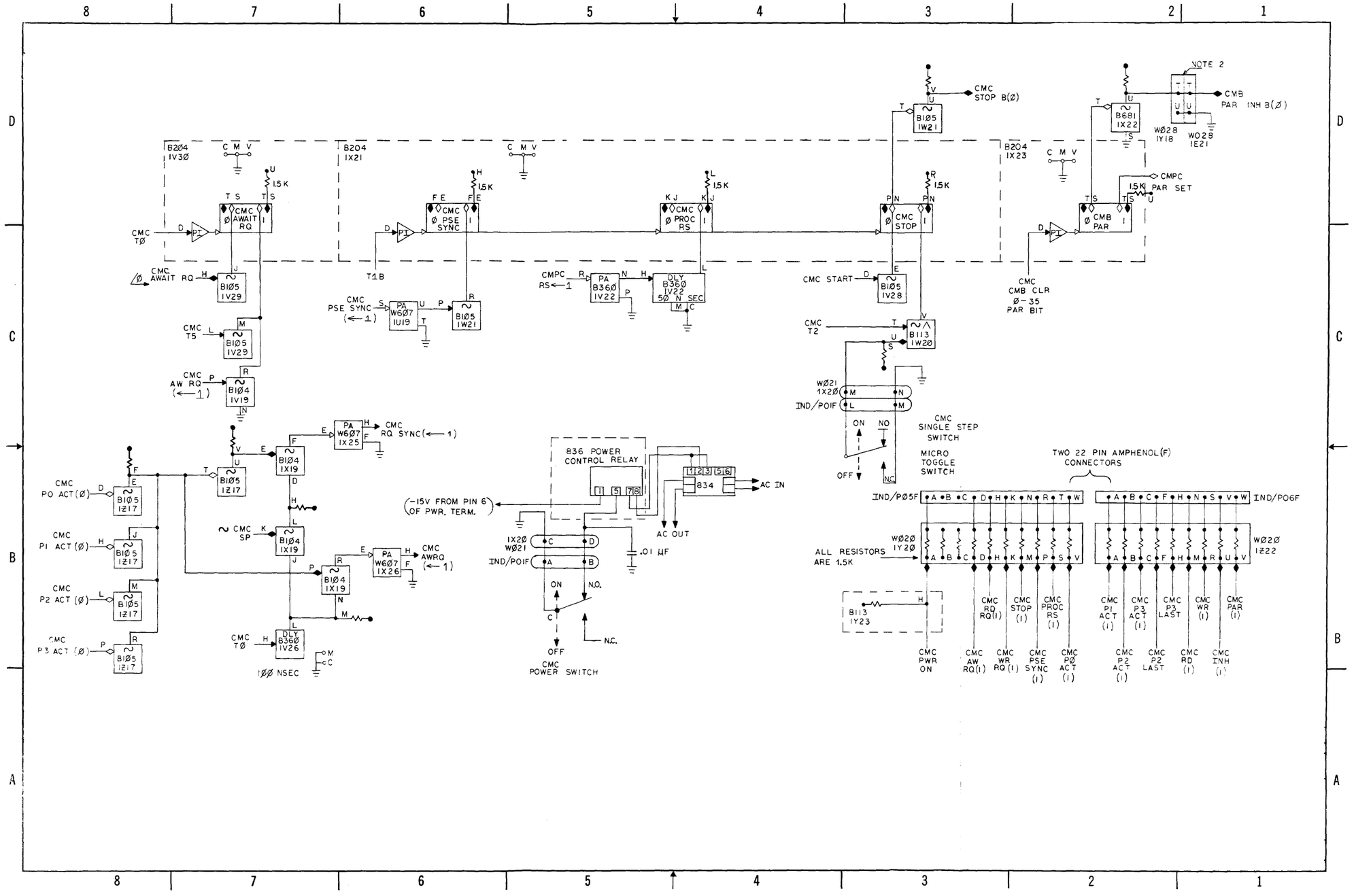


NOTE
1. * ON DRIVING END OF MEMORY LSBS ARE SWITCHED
   EX. (AS) = LSBI ≠ (AU) = LSBØ
2. USE ONLY G206 (B BOARD) MODULES OR LATER.

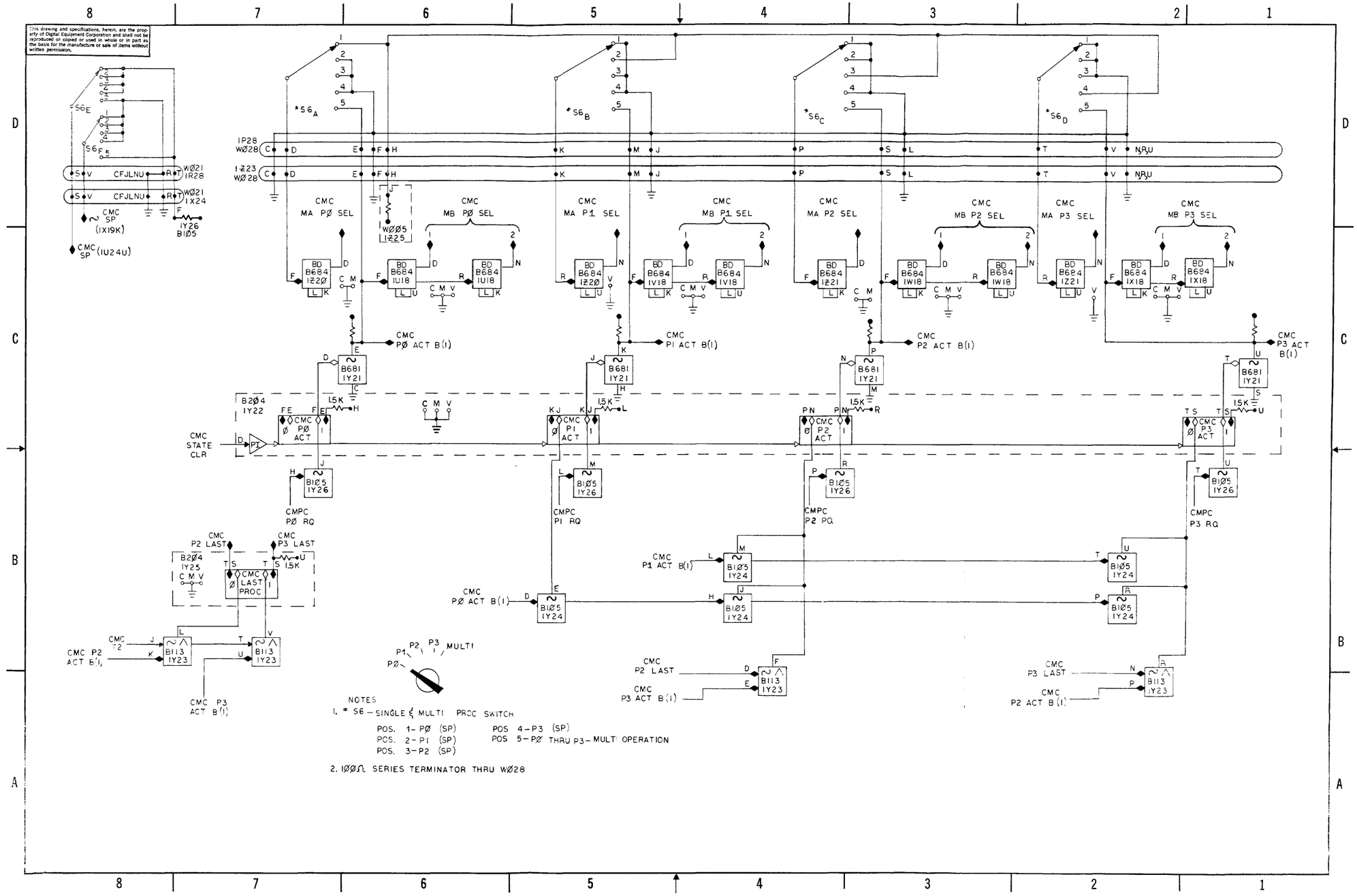
BS-D-164-0-CMY Core Memory Y-Selection Bits 0-35 and Parity Bit. Rev. A



BS-D-164-0-DMSX Diode Matrix
X-Selection



BS-D-164-0-CMC2 Memory Control (Part 1). Rev. B

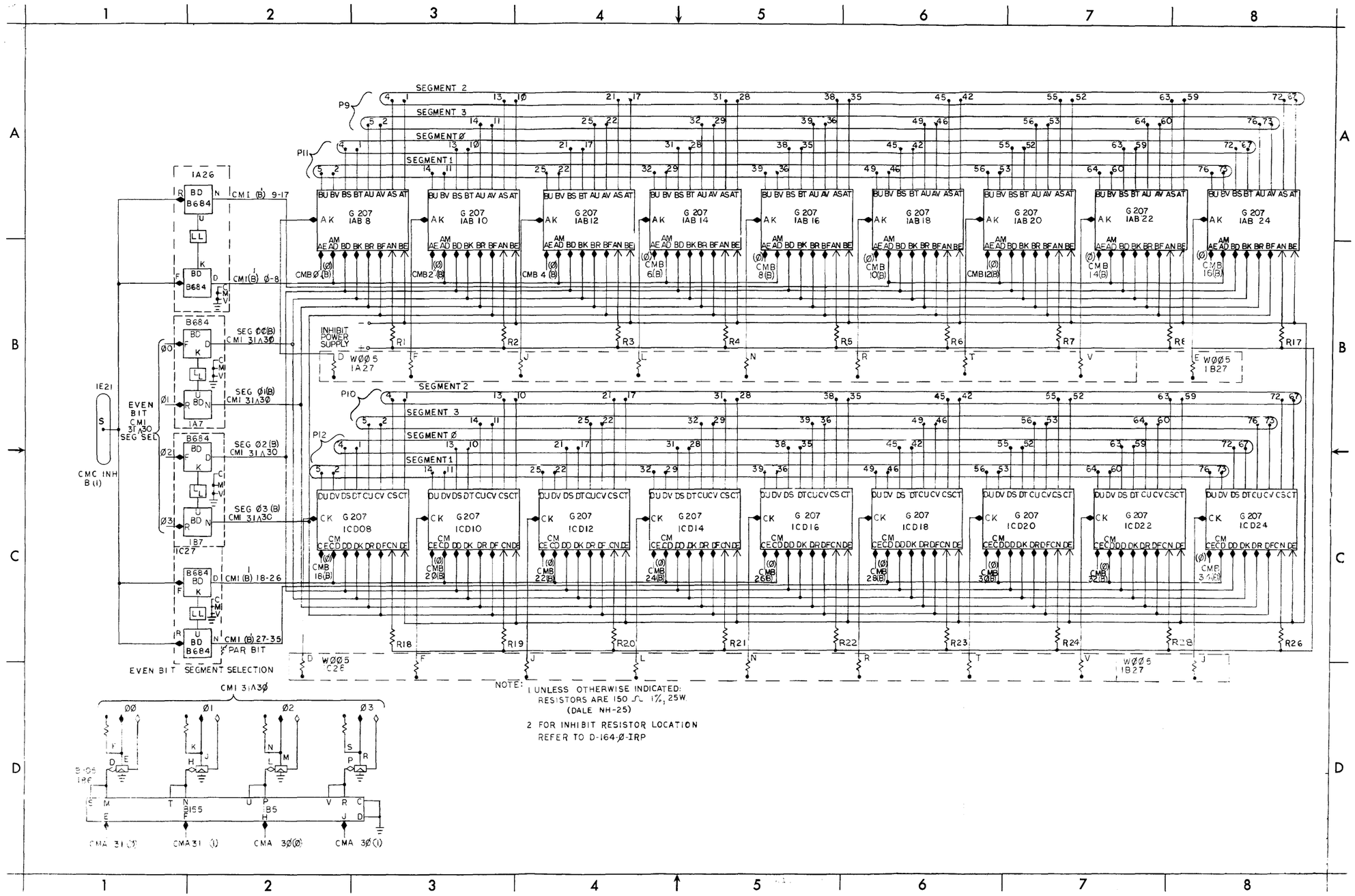


NOTES

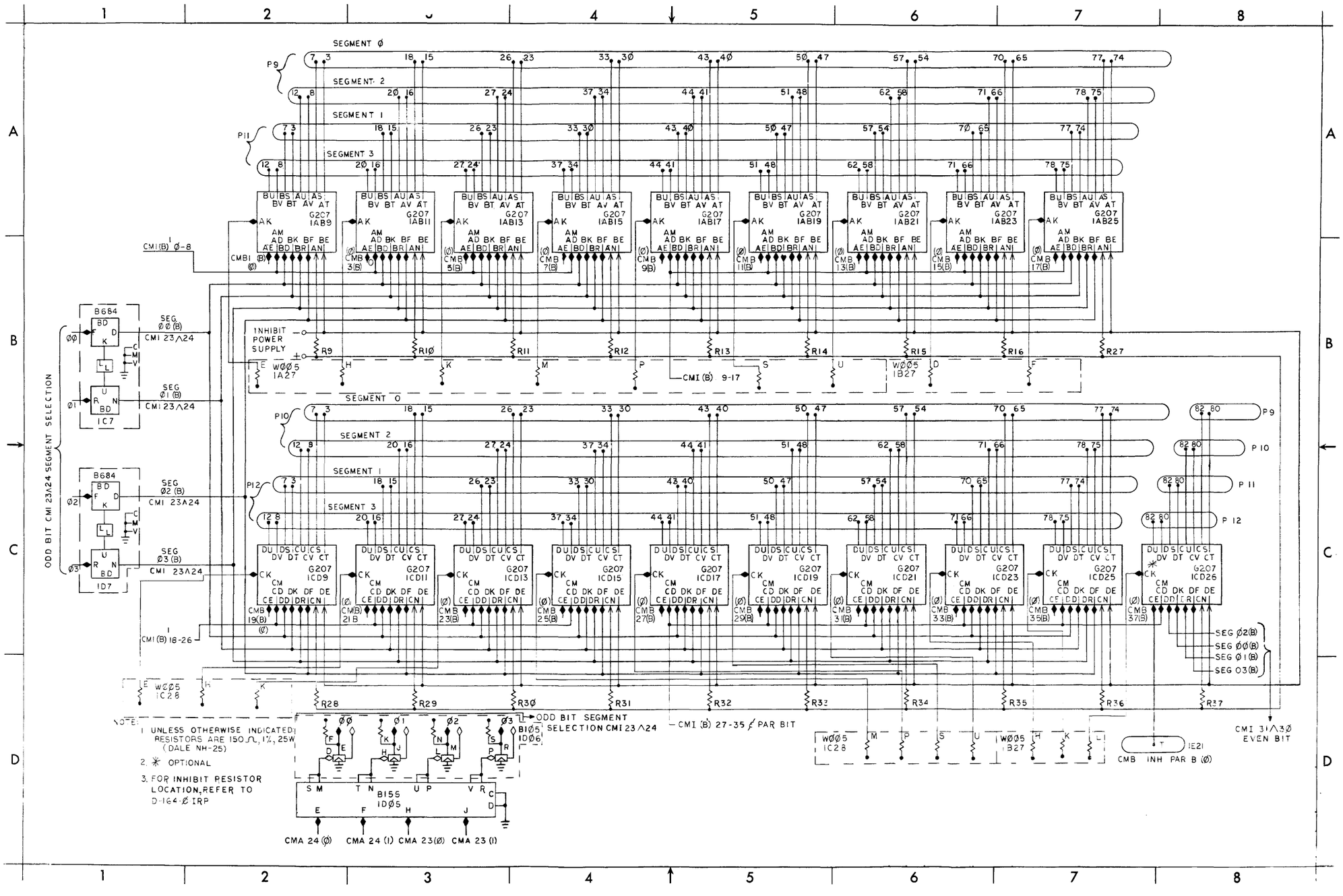
1. * S6 — SINGLE & MULTI PROC SWITCH

POS. 1- PØ (SP)
POS. 2- P1 (SP)
POS. 3- P2 (SP)
POS 4- P3 (SP)
POS 5- PØ THRU P3- MULTI OPERATION

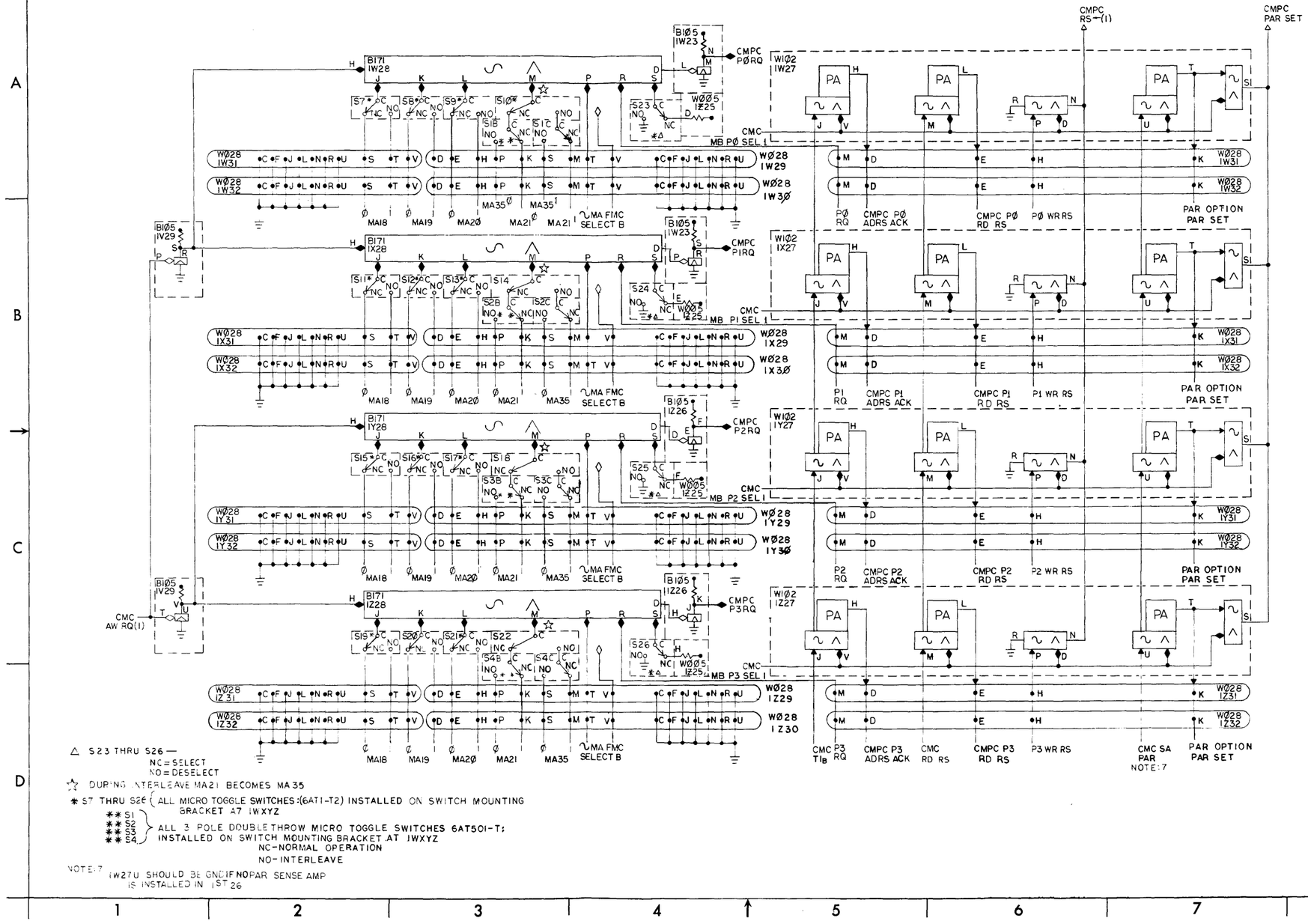
2. 1ØØΩ SERIES TERMINATOR THRU WØ28

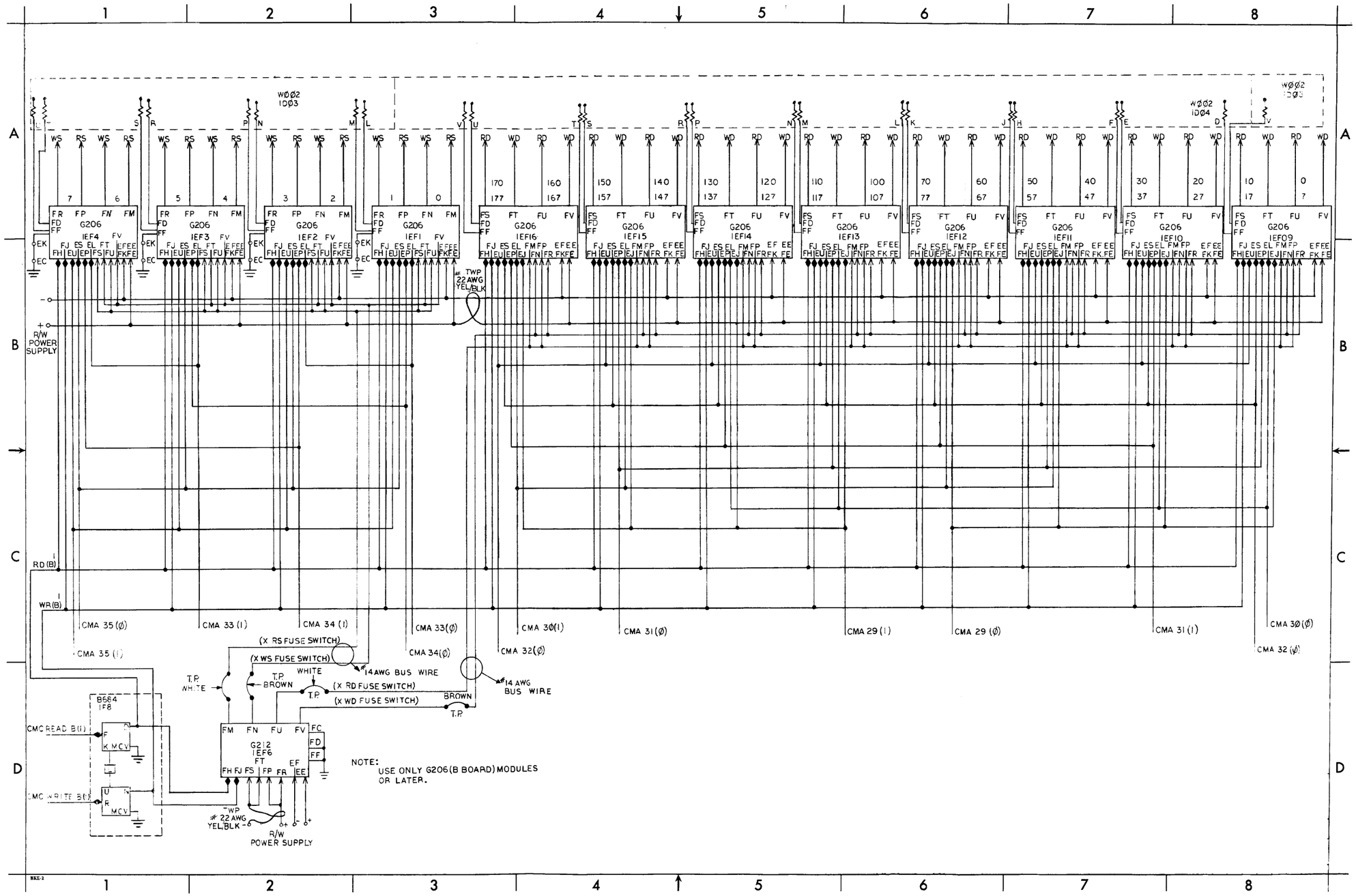


BS-D-164-0-CMI-1 CMI Inhibit Even Bits
Rev. A

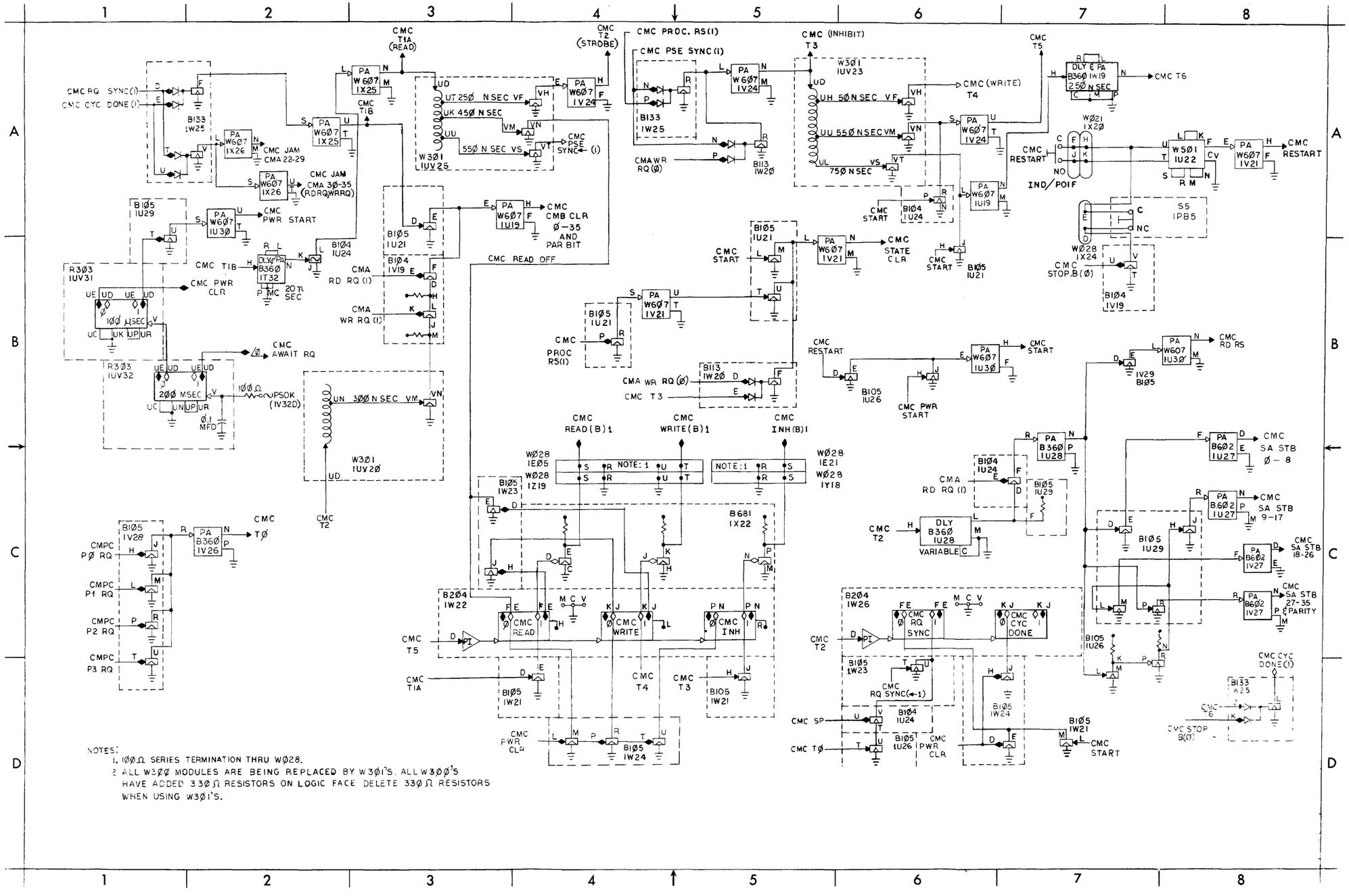


BS-D-164-0-CMI-2 CMI Inhibit Odd Bits and Parity Bits


CMPC RS-(1)
CMPC PAR SET
CMPC PØRQ
CMPC P1RQ
CMPC P2RQ
CMPC P3RQ
CMC AW RQ(1)
CMC MB PØ SEL 1
CMC MB P1 SEL 1
CMC MB P2 SEL 1
CMC MB P3 SEL 1
MA FMC SELECT B
PØ RQ
CMPC PØ ADRS ACK
CMPC PØ RD RS
PØ WR RS
P1 RQ
CMPC P1 ADRS ACK
CMPC P1 RD RS
P1 WR RS
P2 RQ
CMPC P2 ADRS ACK
CMPC P2 RD RS
P2 WR RS
CMC P3 TIB RQ
CMPC P3 ADRS ACK
CMC RD RS
CMPC P3 RD RS
P3 WR RS
CMC SA PAR NOTE:7
PAR OPTION PAR SET
△ S23 THRU S26 — NC=SELECT NO=DESELECT
☆ DURING INTERLEAVE MA21 BECOMES MA35
* S7 THRU S26 (ALL MICRO TOGGLE SWITCHES:(6AT1-T2) INSTALLED ON SWITCH MOUNTING BRACKET A7 IWXYZ
** S1 ** S2 ** S3 ** S4 } ALL 3 POLE DOUBLE THROW MICRO TOGGLE SWITCHES 6AT501-T; INSTALLED ON SWITCH MOUNTING BRACKET AT IWXYZ
NC-NORMAL OPERATION
NO-INTERLEAVE
NOTE:7 IW27U SHOULD BE GND IF NOPAR SENSE AMP IS INSTALLED IN 1ST 26

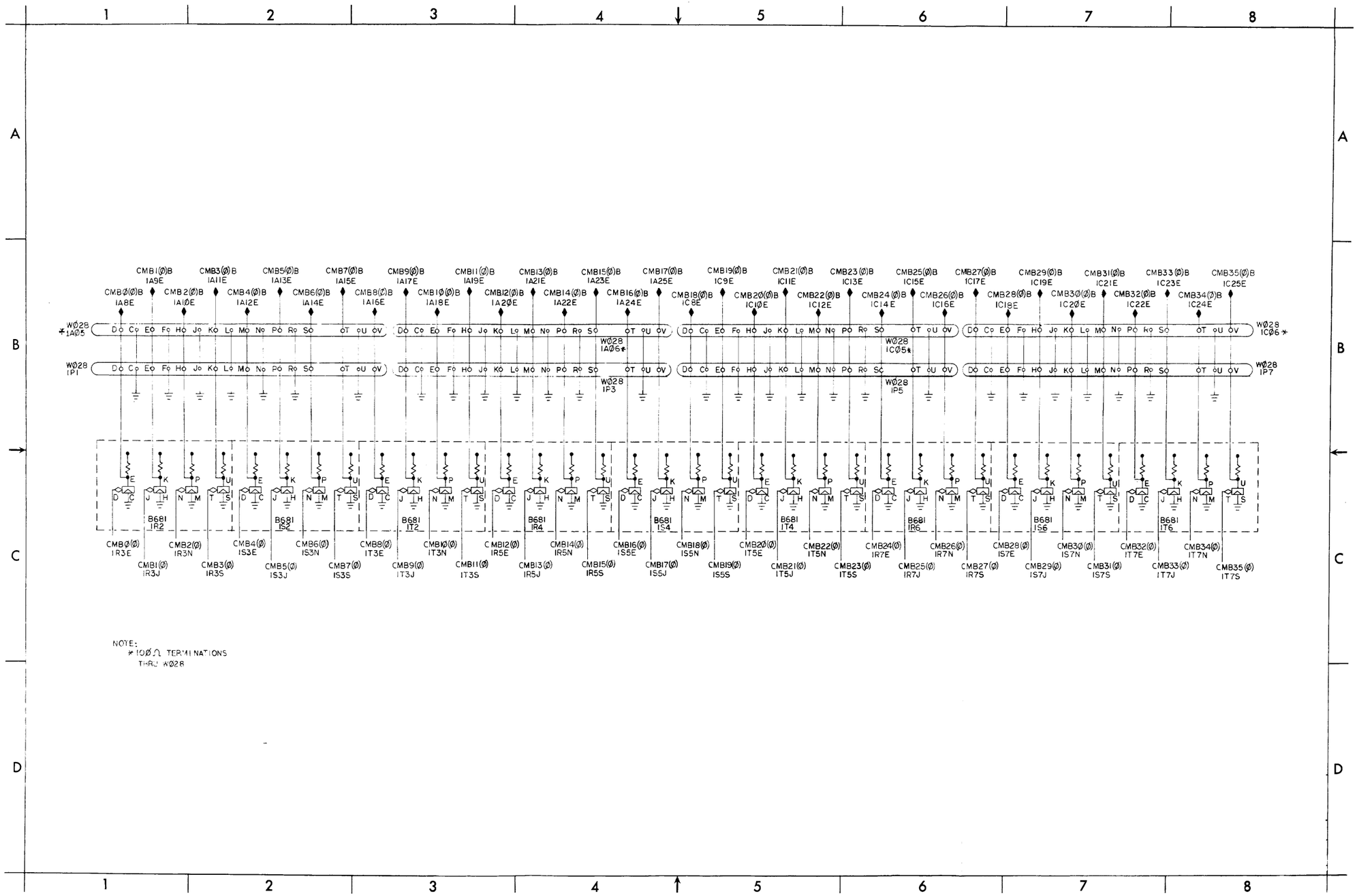


BS-D-164-0-CMX Core Memory X-Selection
0-35 and Parity Bit. Rev. A

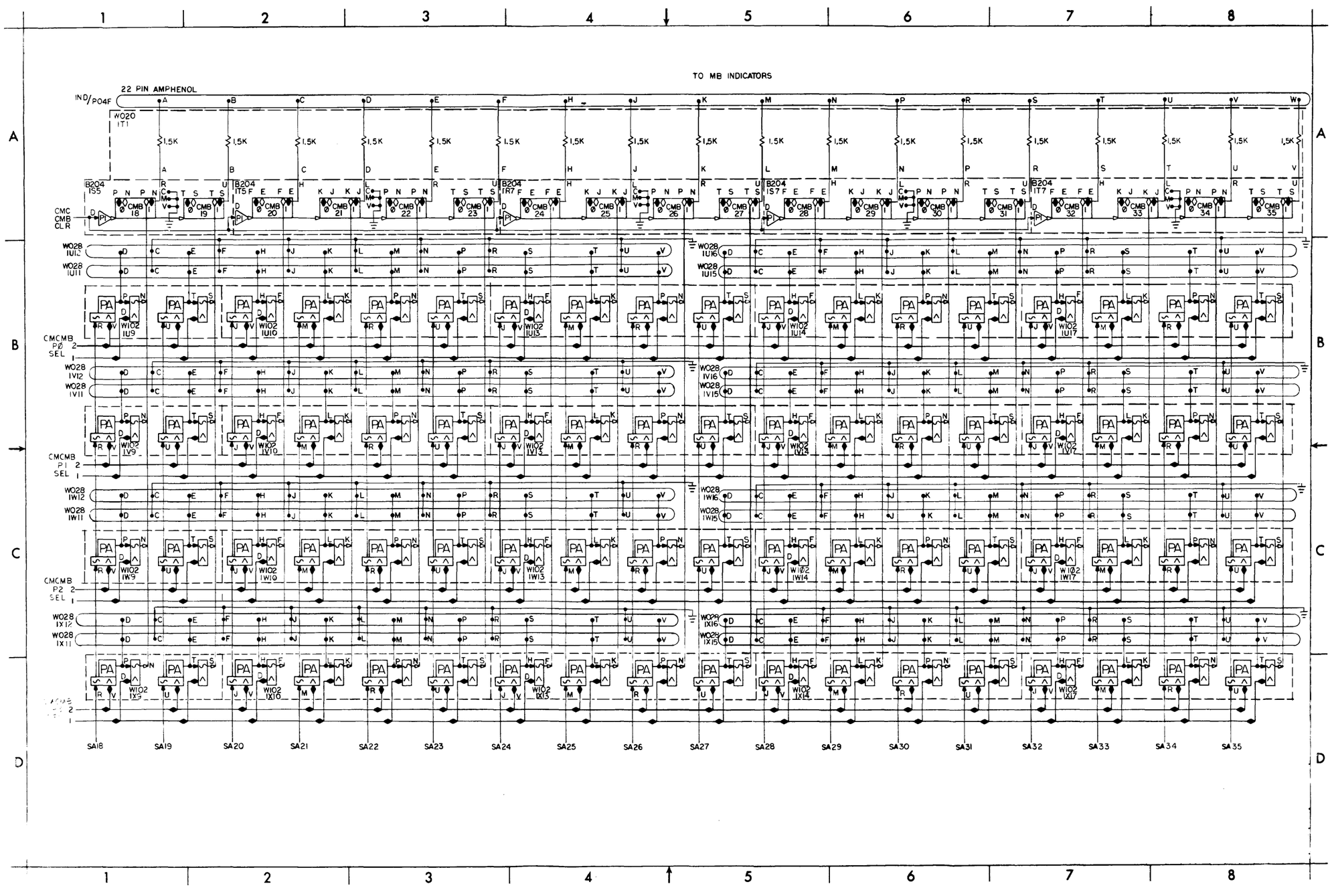


NOTES:
1. 1ØØΩ SERIES TERMINATION THRU WØ28.
2. ALL W3ØØ MODULES ARE BEING REPLACED BY W3Ø1'S. ALL W3ØØ'S HAVE ADDED 33ØΩ RESISTORS ON LOGIC FACE DELETE 33ØΩ RESISTORS WHEN USING W3Ø1'S.

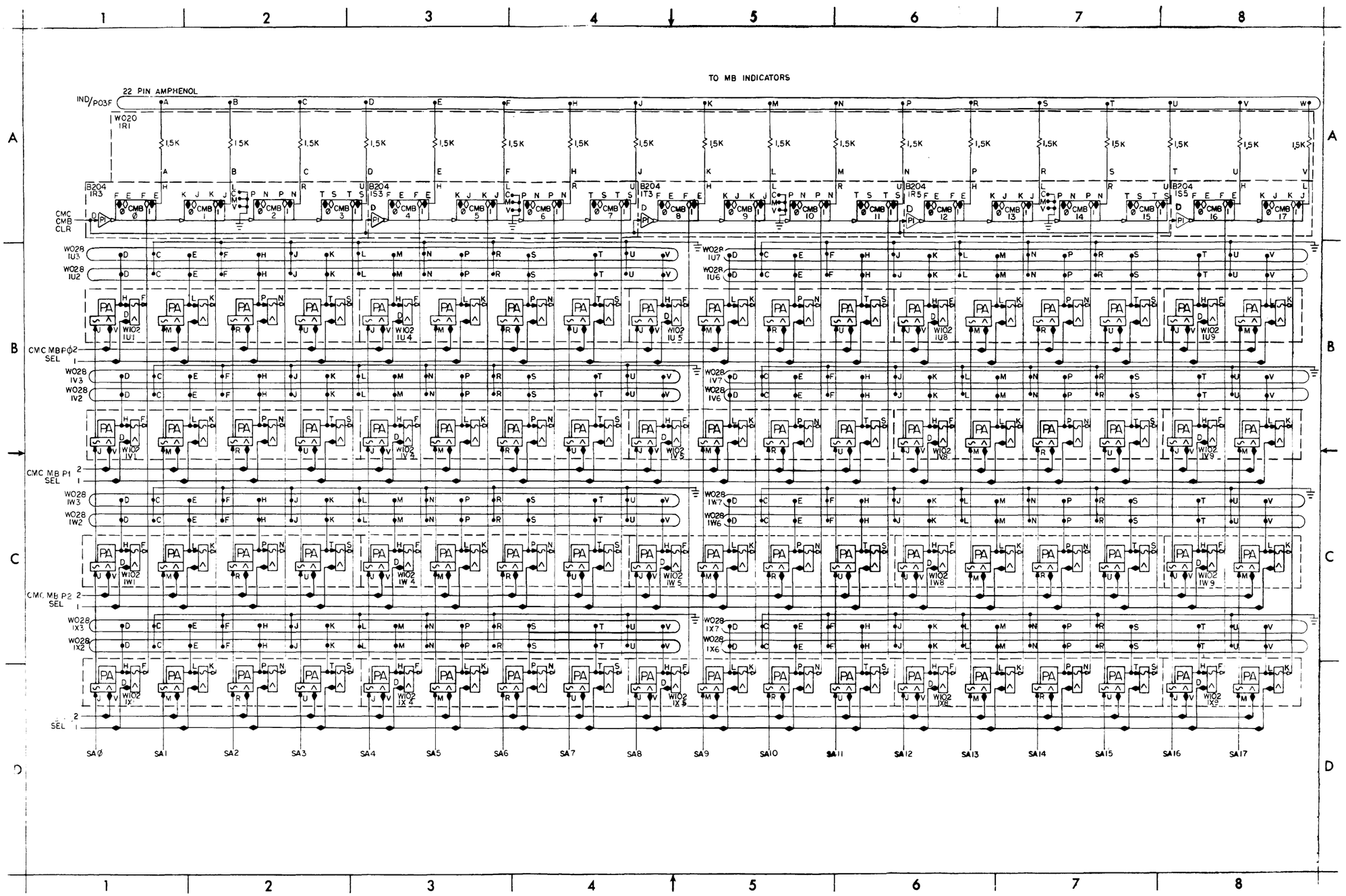
BS-D-164-0-CMC1 Core Memory Control Timing. Rev. D



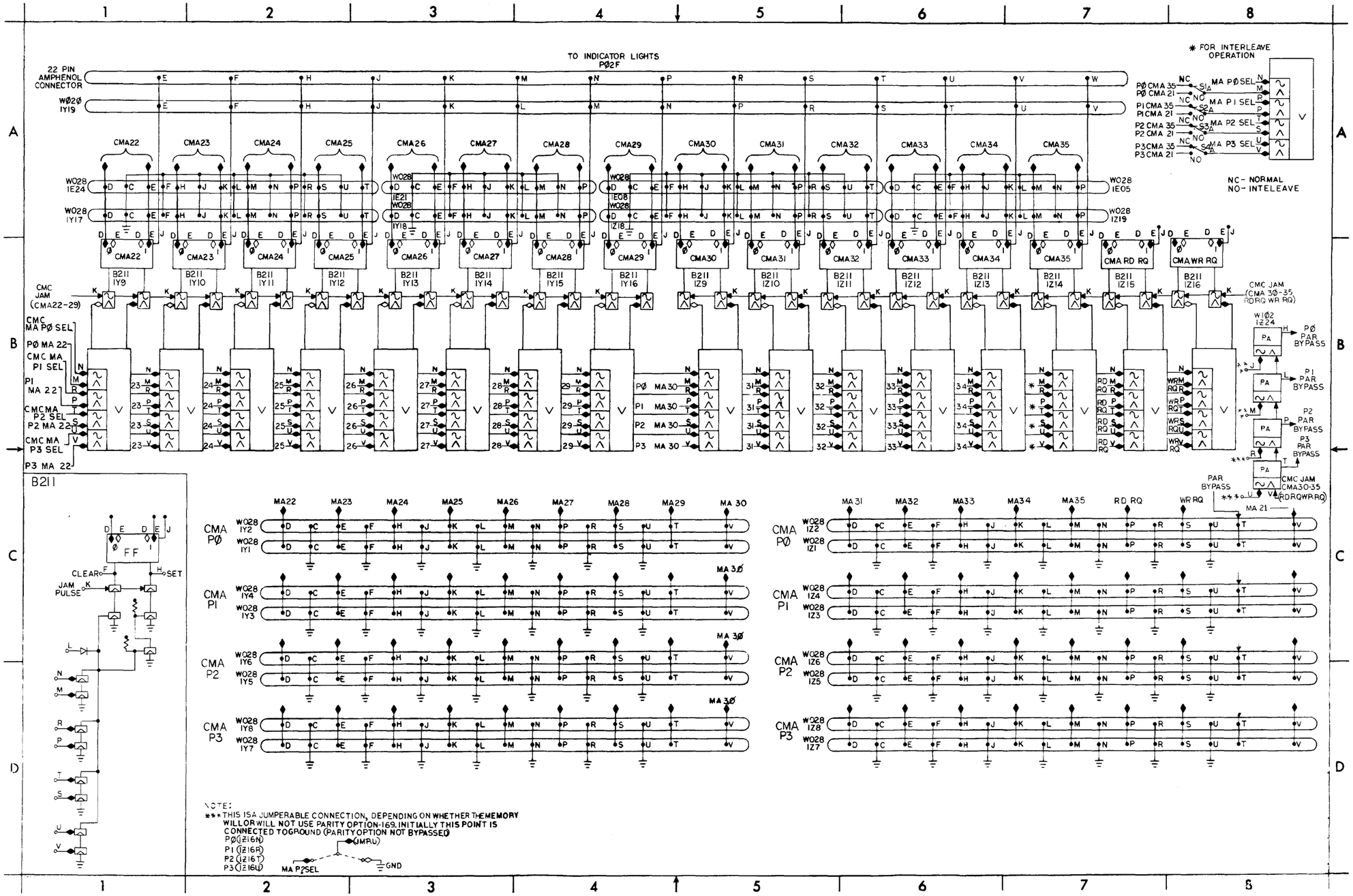
NOTE:
*100Ω TERMINATIONS THRU WØ28



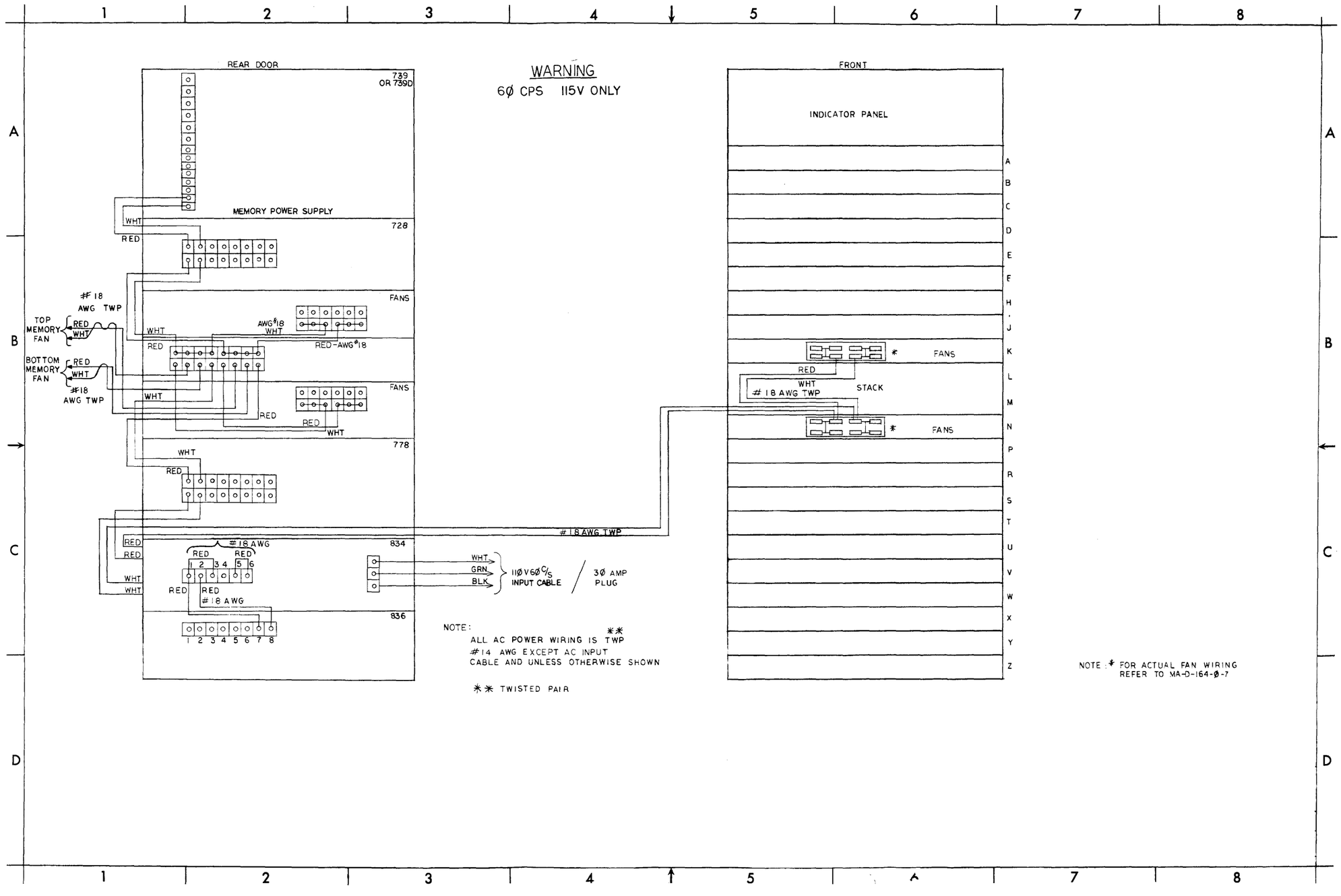
BS-D-164-0-CMBR Core Memory Buffer
18-35 Type 164

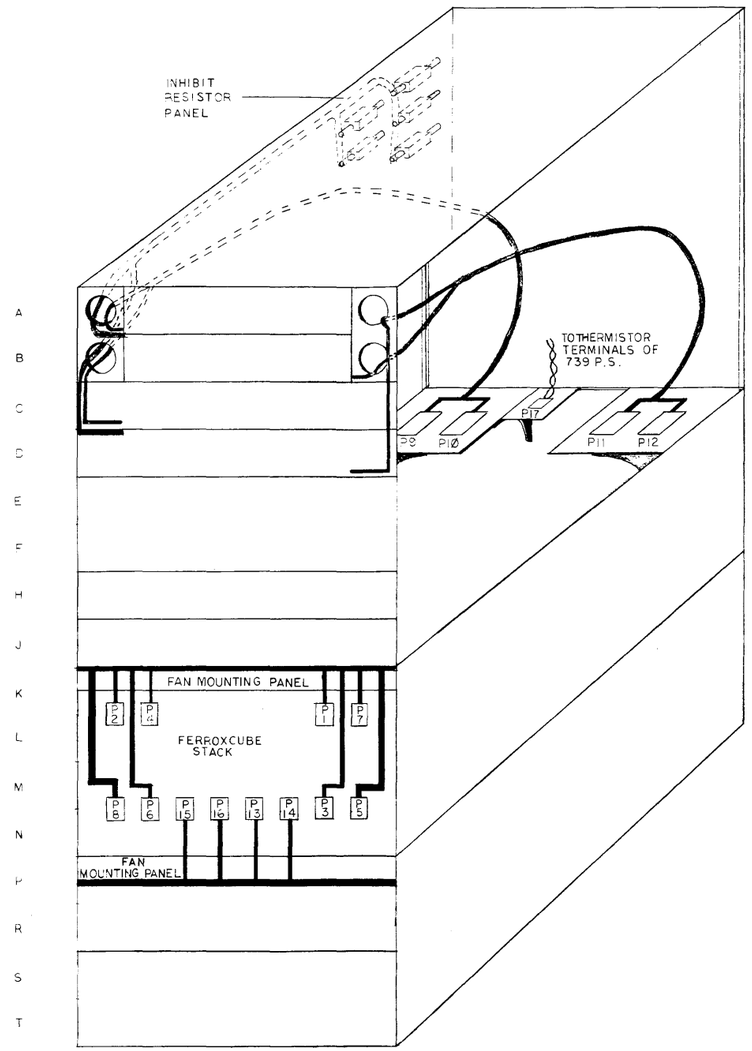


BS-D-164-0-CMBL Core Memory Buffer
0-17 Type 164



BS-D-164-0-CMA CMA. Rev. A

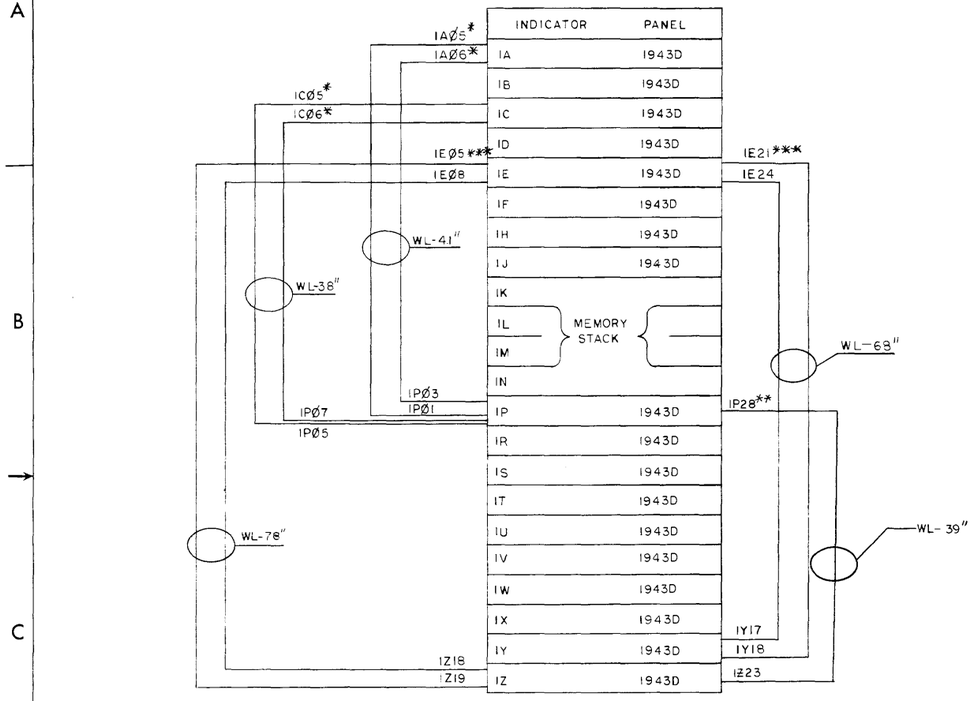

REAR DOOR
739
OR 739D
MEMORY POWER SUPPLY
728
FANS
778
834
836
WARNING
60 CPS 115V ONLY
#18 AWG TWP
TOP MEMORY FAN
BOTTOM MEMORY FAN
RED
WHT
GRN
BLK
#18 AWG
RED-AWG #18
1 2 3 4 5 6
1 2 3 4 5 6 7 8
110V60C/S INPUT CABLE
30 AMP PLUG
NOTE:
** ALL AC POWER WIRING IS TWP #14 AWG EXCEPT AC INPUT CABLE AND UNLESS OTHERWISE SHOWN
** TWISTED PAIR
FRONT
INDICATOR PANEL
* FANS
STACK
NOTE: * FOR ACTUAL FAN WIRING REFER TO MA-D-164-0-7



NOTES:
1. P1 THRU P16 ARE 75 PIN BURNDY CONNECTORS # MS75R-124
2. P17 IS A 14 PIN BURNDY CONNECTOR # MS14R-1

CABLE LENGTHS			CABLE LISTS	
FROM	TO	42 IN	DWG NO.	CODE
IHJ	P1	↑	A-164-Ø-CBLS-9	CL
IHJ	P2		A-164-Ø-CBLS-10	CL
IHJ	P3		A164-Ø-CBLS-11	CL
IHJ	P4		A164-Ø-CBLS-12	CL
IHJ	P5		A164-Ø-CBLS-13	CL
IHJ	P6		A164-Ø-CBLS-14	CL
IHJ	P7	↓	A164-Ø-CBLS-15	CL
IHJ	P8	42 IN	A164-Ø-CBLS-16	CL
IA	P9	72 IN	A164-Ø-CBLS-6	CL
IB	P10	↑	A164-Ø-CBLS-5	CL
IC	P11	↓	A164-Ø-CBLS-4	CL
ID	P12	72 IN	A-164-Ø-CBLS-3	CL
IT	P13	48 IN	A-164-Ø-CBLS-17	CL
IT	P14	↑	A-164-Ø-CBLS-18	CL
IR	P15	↓	A-164-Ø-CBLS-7	CL
IR	P16	48 IN	A164-Ø-CBLS-8	CL
IA	RES	66 IN	A-164-Ø-12	CL
IC	RES	66 IN	A-164-Ø-13	CL
739 PS.	P17	36 IN	A-164-Ø-21	CL

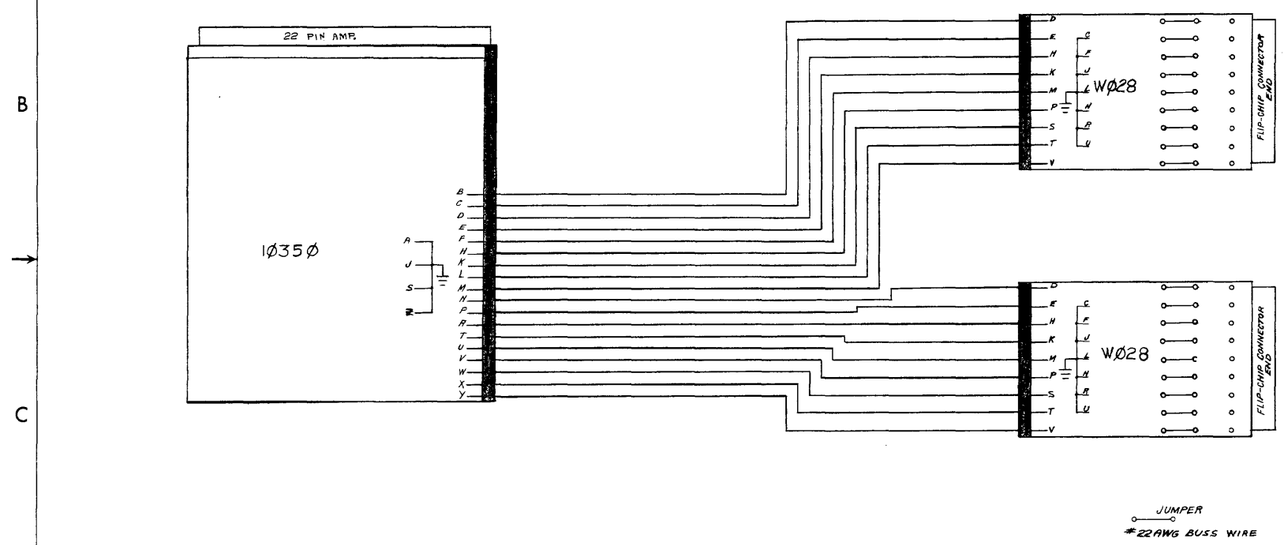
CD-D-164-0-CD Cable Diagram Core Memory
164 (Part 1). Rev. A



NOTE:

1. WL-WIRE LENGTH
2. * — DESIGNATES END OF CABLE WHICH HAS SERIES TERMINATION OF ØØΩ, 1/4W 10% RESISTORS
3. ** — SAME AS NOTE 2 EXCEPT PIN E, M, & V ONLY WILL HAVE SERIES TERMINATION OF 100Ω 1/4W RESISTORS & THE OTHER PINS WILL BE JUMPERS.
4. *** — SAME AS NOTE 2 EXCEPT PIN S, T ONLY WILL HAVE SERIES TERMINATIONS OF 100Ω, 1/4W RESISTORS & THE OTHER PINS WILL BE JUMPERS.

CD-D-164-0-CD Cable Diagram Core Memory
164 (Part 2). Rev. A

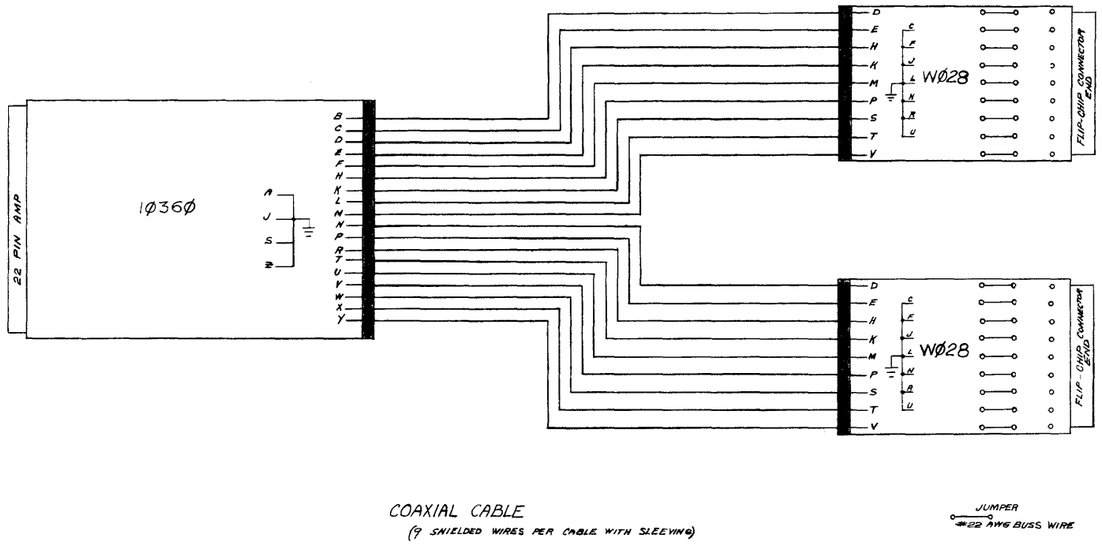


COAXIAL CABLE
(9 SHIELDED WIRES PER CABLE WITH SLEEVING)

*FOR USE BETWEEN:
TYPE 164 AND C.P.
TYPE 164 TO TYPE 163C
TYPE 169 AND MEMORY EXERCISER

CABLE LENGTH WILL BE SPECIFIED WHEN ORDERED

CD-D-164-0-CCD Coaxial Cable
Diagram (Part 1)



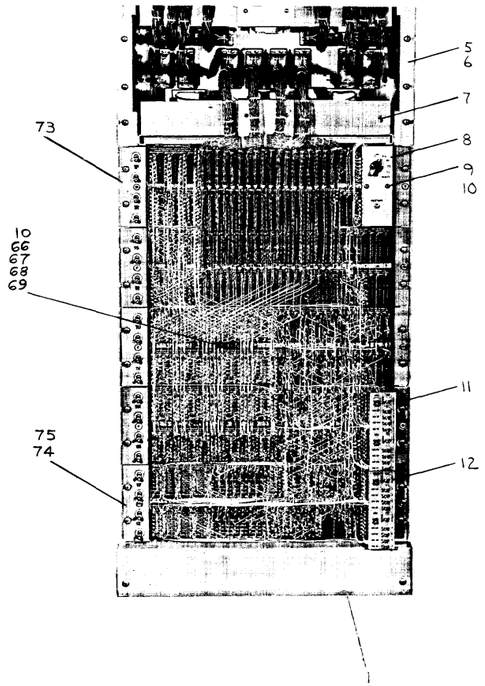
COAXIAL CABLE
(9 SHIELDED WIRES PER CABLE WITH SLEEVING)

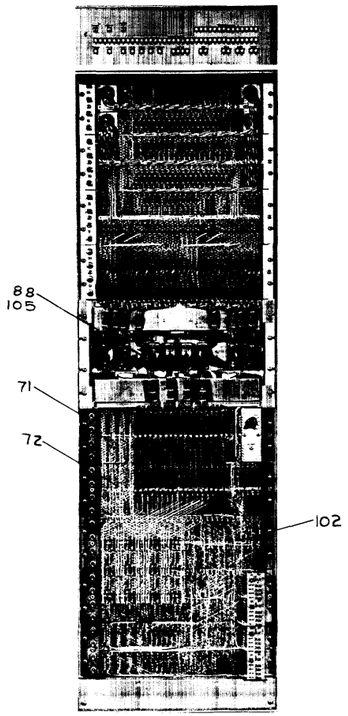
JUMPER
#22 AWG BUSS WIRE

*FOR USE BETWEEN:
TYPE 164 AND C.P.
TYPE 164 TO TYPE 163C
TYPE 164 AND MEMORY EXERCISER

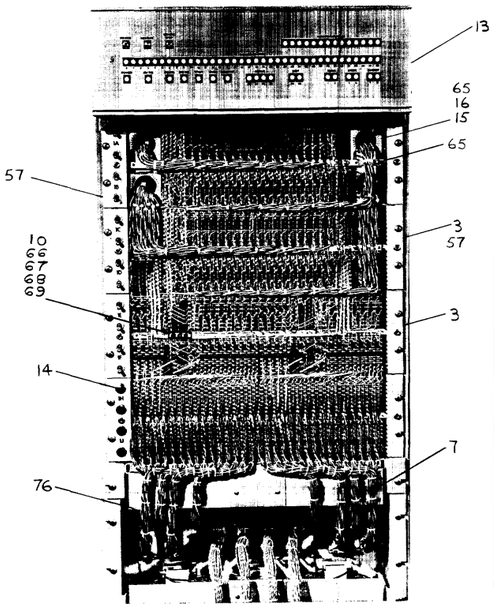
CABLE LENGTH WILL BE SPECIFIED WHEN ORDERED

CD-D-164-0-CCD Coaxial Cable
Diagram (Part 2)



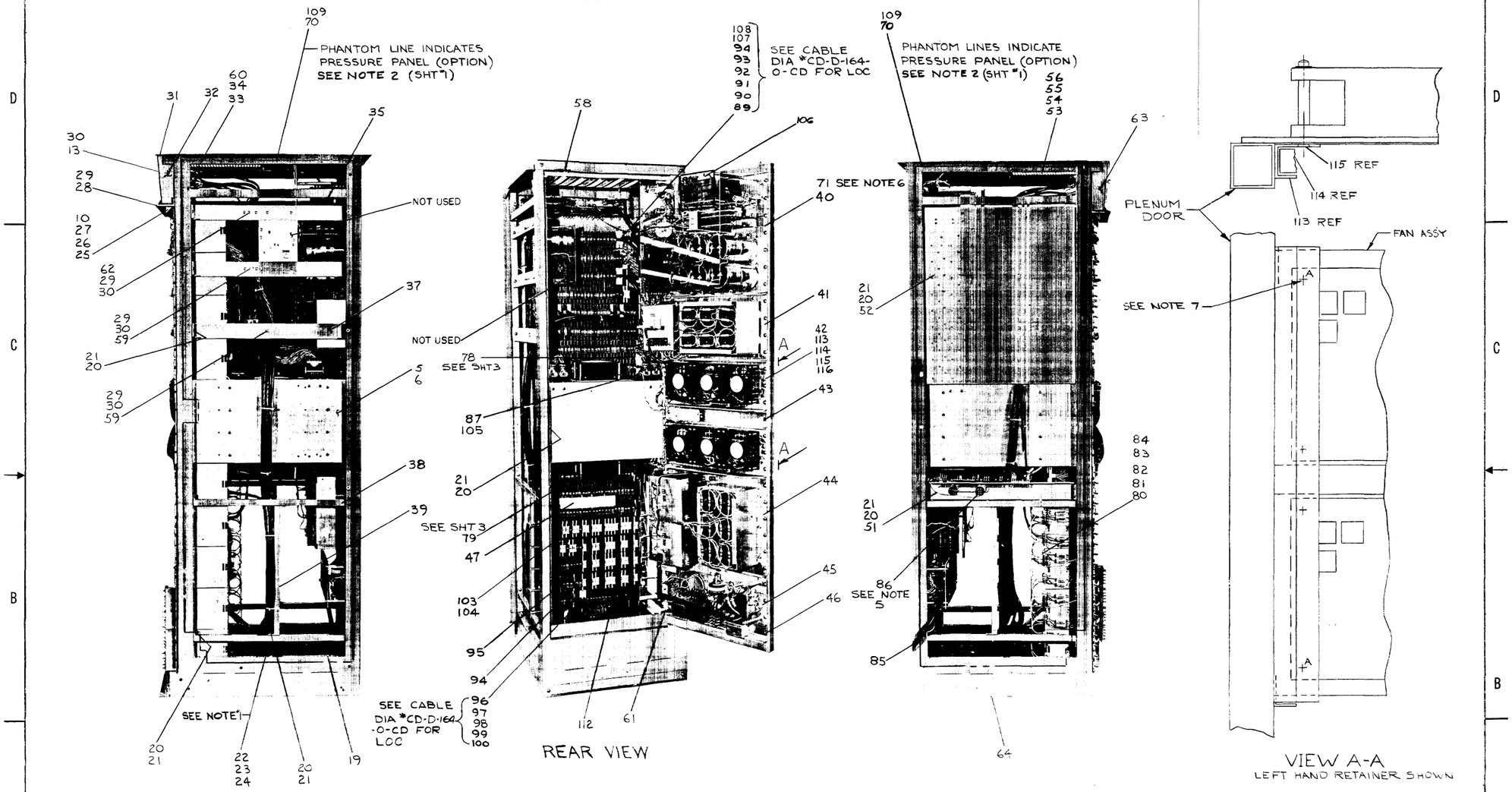


FRONT VIEW WITH DOORS
AND END PANELS REMOVED

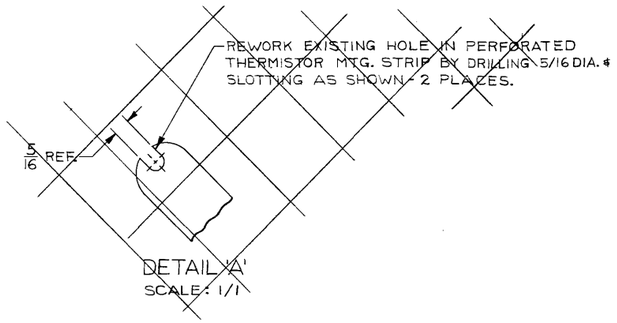


NOTES:

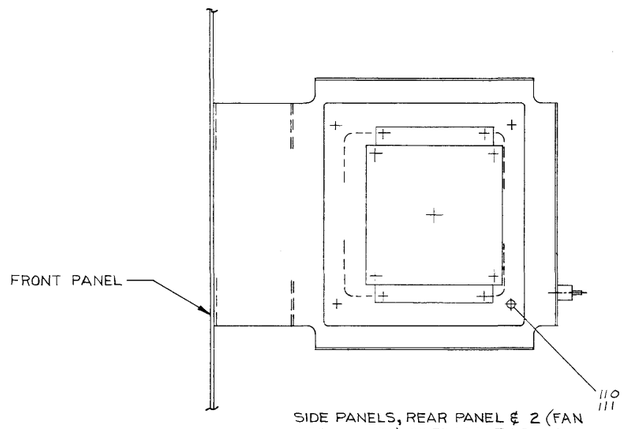
1. DRILL THRU (3) 32 DIA HOLES IN CABINET FRAME, THRU SKID DWG NO. MD-D-0142 & INSTALL PART NO. 22, 23 & 24
2. IF #70 (PRESSURE PANEL) IS USED, YOU MAY BY USING 2 HOLES AT TOP OF CABLE DUCT RIVET (109) IN PLACE. DRILL REMAINING HOLES #20 (161) DRILL USING PANEL AS TEMPLATE & POP RIVET (109), [illegible] PLACE.
3. ~~IF #70 PRESSURE PANEL IS USED PART TO HAVE 5/32 HOLES CSK 82° TO .55 DIA FOR #6-32 X 1/4 LG. FLAT HEAD SCREWS (MAKE SURE HOLES ARE CSK ON CORRECT SIDE) WHICH DEPENDS ON WHETHER LEFT OR RIGHT SIDE MTG IS REQD.~~
4. #101 (164 BUS TERMINATOR. DWG NO. B-IA-5403930) NOT SHOWN ON THIS ASSEMBLY. LOC TO BE SPECIFIED BY PROJ ENG.
5. LENGTH OF CABLE TO BE SPEC BY PROJ ENG.
6. LABEL TO BE PLACED IN THIS LOCATION WHICH READS "THIS UNIT REQ A G800D BOARD."
7. REMOVE 4 "A" SCREWS, 2 LEFT SIDE, 2 RIGHT SIDE, & ASSEMBLE ITEMS #113 & 116 USING FLT H'D SCREWS ITEM #115 TO REPLACE "A" SCREWS.



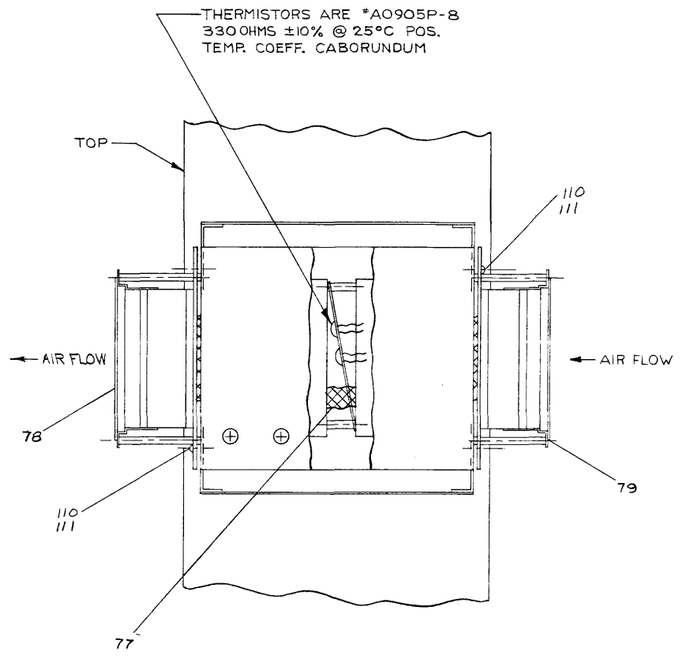
D-UA-164-0-1 Core Memory Type 164 (Part 2). Rev. J



DETAIL 'A'
SCALE: 1/1



SIDE PANELS, REAR PANEL & 2 (FAN HSG. ASSY.) OMITTED IN THIS VIEW.



NOTES:
1. THIS DWG WAS FORMERLY CALLED 164 MEMORY REWORK. DWG # D-AD-164-0-15, BUT HAS BEEN MADE SHT 3 OF 164 ASSY.

* 739A POWER SUPPLY MODIFICATIONS FOR TYPE 164 MEMORY

REFER TO MA-D-739-0-1

WIRE CHANGES

CHANGE	LOCATION	COLOR & SIZE	LOCATION
DELETE	DI & D4 CATHODE	RED AWG 14	C1 (+)
DELETE	DI & D4 CATHODE	RED AWG 18	PIN AD OF R/W G800
ADD	PIN AD OF R/W G800	RED AWG 18	C4 (+)
ADD	DI & D4 CATHODE	BLUE AWG 14	TERM. # TO JONES STRIP)
ADD	C1 (+)	YELLOW AWG 14	TERM. # 4 JONES STRIP)
ADD	PIN AS OF R/W G800	WHT AWG 22	PIN BK OF R/W G800
ADD	PIN AS OF INH G800	WHT AWG 22	PIN BK OF INH G800

TRANS	TAP CHANGES (T157144 TRANS. ONLY)
FROM TAP	TO TAP
15	20
11	25

**

NOTE: INSTALL G800-D BOARDS IF PRIOR BOARDS ARE PREVIOUSLY IN

COMPONENT CHANGES ON FLIP CHIP MOUNTING BLOCK

CHANGE	LOCATION	COMPONENT	LOCATION
DELETE	PIN BM OF R/W G800	R9-1.62K, $\frac{1}{8}$W, 1%, MF	PIN BN OF R/W G800
ADD	PIN BM OF R/W G800	R9-2K, $\frac{1}{8}$W, 1%, MF	PIN BN OF R/W G800
DELETE	PIN BM OF INH G800	R10-1.62K, $\frac{1}{8}$W, 1% MF	PIN BN OF INH G800
ADD.	PIN BM OF INH G800	R10-2K, $\frac{1}{8}$W, 1% MF	PIN BN OF INH G800
DELETE	PIN AE OF R/W G800	R3-909Ω, $\frac{1}{8}$W, 1% MF	PIN AC OF R/W G800
ADD	PIN AE OF R/W G800	R3-3.3K, $\frac{1}{4}$W, 5%	PIN AC OF R/W G800

* NOTE: THESE CHANGES APPLY ONLY TO 739A POWER SUPPLIES AND NO CHANGES ARE NECESSARY ON 739E'S.

** THIS CHANGE ALSO APPLIES TO 739E POWER SUPPLIES ALREADY USED IN THE FIELD

PW-D-164-0-50PSM 739A Power Supply Modification for 50 cps, 220V

✳ 739 POWER SUPPLY MODIFICATIONS FOR TYPE 164 MEMORY

REFER TO MA-D-739-0-1

WIRE CHANGES

CHANGE	LOCATION	COLOR & SIZE	LOCATION
DELETE	DI E D4 CATHODE	RED AWG 14	C1 (+)
DELETE	DI E D4 CATHODE	PED AWG 18	PIN AD OF R/W G800
ADD	PIN AD OF R/W G800	RED AWG 18	C4 (+)
ADD	DI E D4 CATHODE	BLUE AWG 14	TERM # 10 (JONES STRIP)
ADD	C1 (+)	YELLOW AWG 14	TERM # 4 (JONES STRIP)
ADD	PIN AS OF R/W G800	WHT AWG 22	PIN BK OF R/W G800
ADD	PIN AS OF INH G800	WHT AWG 22	PIN BK OF INH G800

TRANS	TAP CHANGES (T1 57109 TRANS. ONLY)
FROM TAP	TO TAP
3	17
6	11

✳✳

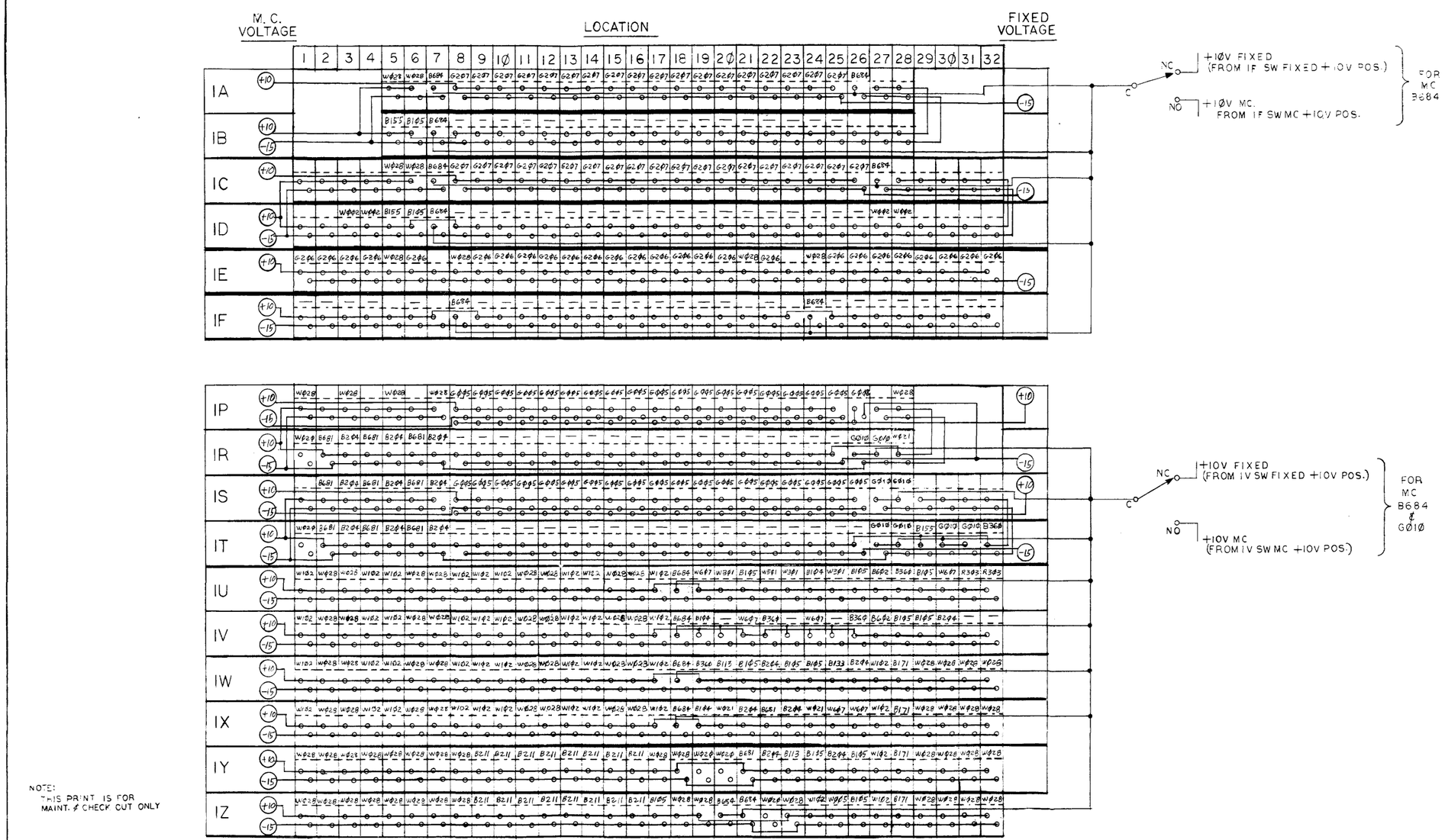
NOTE: INSTALL G800-D BOARDS IF PRIOR BOARDS ARE PREVIOUSLY IN

COMPONENT CHANGES ON FLIP CHIP MOUNTING BLOCK

CHANGE	LOCATION	COMPONENT	LOCATION
DELETE	PIN BM OF R/W G800	R9-1,62K, $\frac{1}{8}$W, 1% MF	PIN BN OF R/W G800
ADD	PIN BM OF R/W G800	R9— 2K, $\frac{1}{8}$W, 1% MF	PIN BN OF R/W G800
DELETE	PIN BM OF INH G800	R10-1,62K, $\frac{1}{8}$W, 1% MF	PIN BN OF INH G800
ADD	PIN BM OF INH G800	R10— 2K, $\frac{1}{8}$W, 1% MF	PIN BN OF INH G800
DELETE	PIN AE OF R/W G800	R3-909Ω, $\frac{1}{8}$W, 1% MF	PIN AC OF R/W G800
ADD	PIN AE OF R/W G800	R3-3,3K, $\frac{1}{4}$W, 5%	PIN AC OF R/W G800

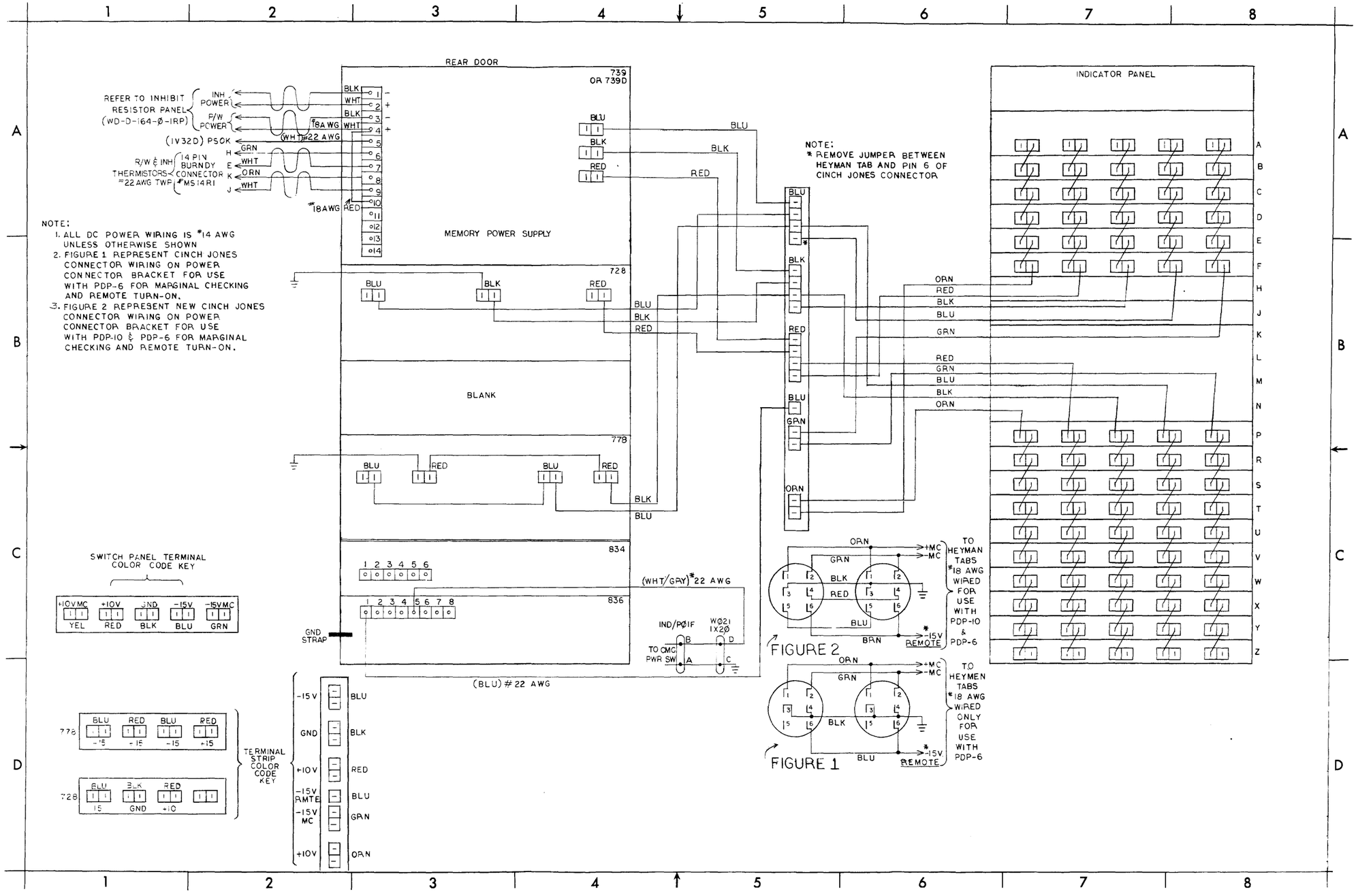
✳ NOTE: THESE CHANGES APPLY ONLY TO 739 POWER SUPPLIES AND NO CHANGES ARE NECESSARY ON 739DS

✳✳ THIS CHANGE ALSO APPLIES TO 739D POWER SUPPLIES ALREADY USED IN THE FIELD

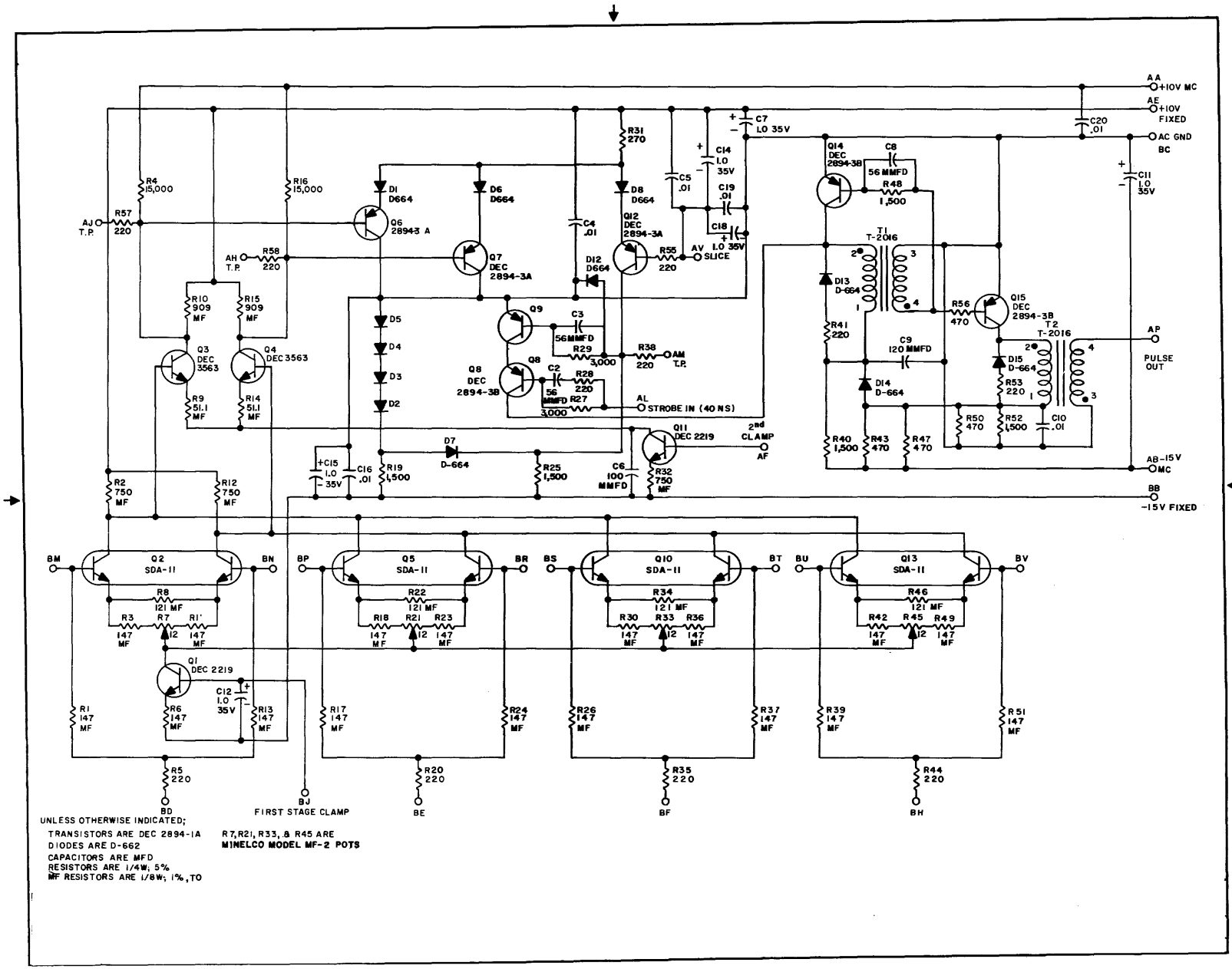


NOTE:
THIS PRINT IS FOR MAINT. & CHECK OUT ONLY

CD-D-164-0-MCR Margin Check Routine with Module Location. Rev. C



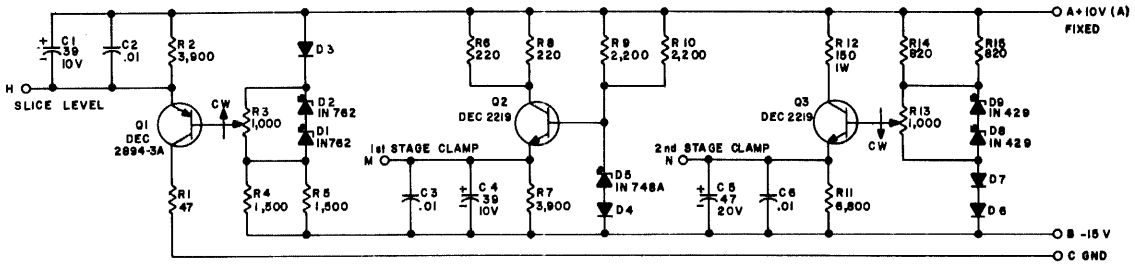
PW-D-164-0-DCPW DC Wiring for Type 164 Memory System. Rev. B



UNLESS OTHERWISE INDICATED;
TRANSISTORS ARE DEC 2894-1A
DIODES ARE D-662
CAPACITORS ARE MFD
RESISTORS ARE 1/4W; 5%
MF RESISTORS ARE 1/8W; 1%, TO

R7, R21, R33, & R45 ARE MINELCO MODEL MF-2 POTS

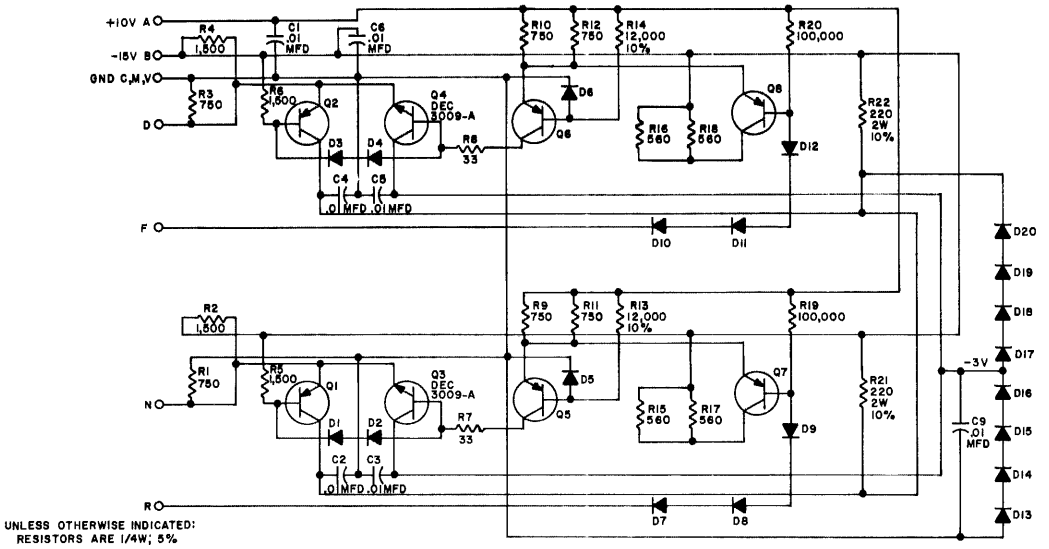
CS-C-G005-0-1 4 Input DC Sense Amplifier, G005. Rev. L



UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W, 5%
CAPACITORS ARE MFD
DIODES ARE D-662
R3 & R13 ARE A #275P

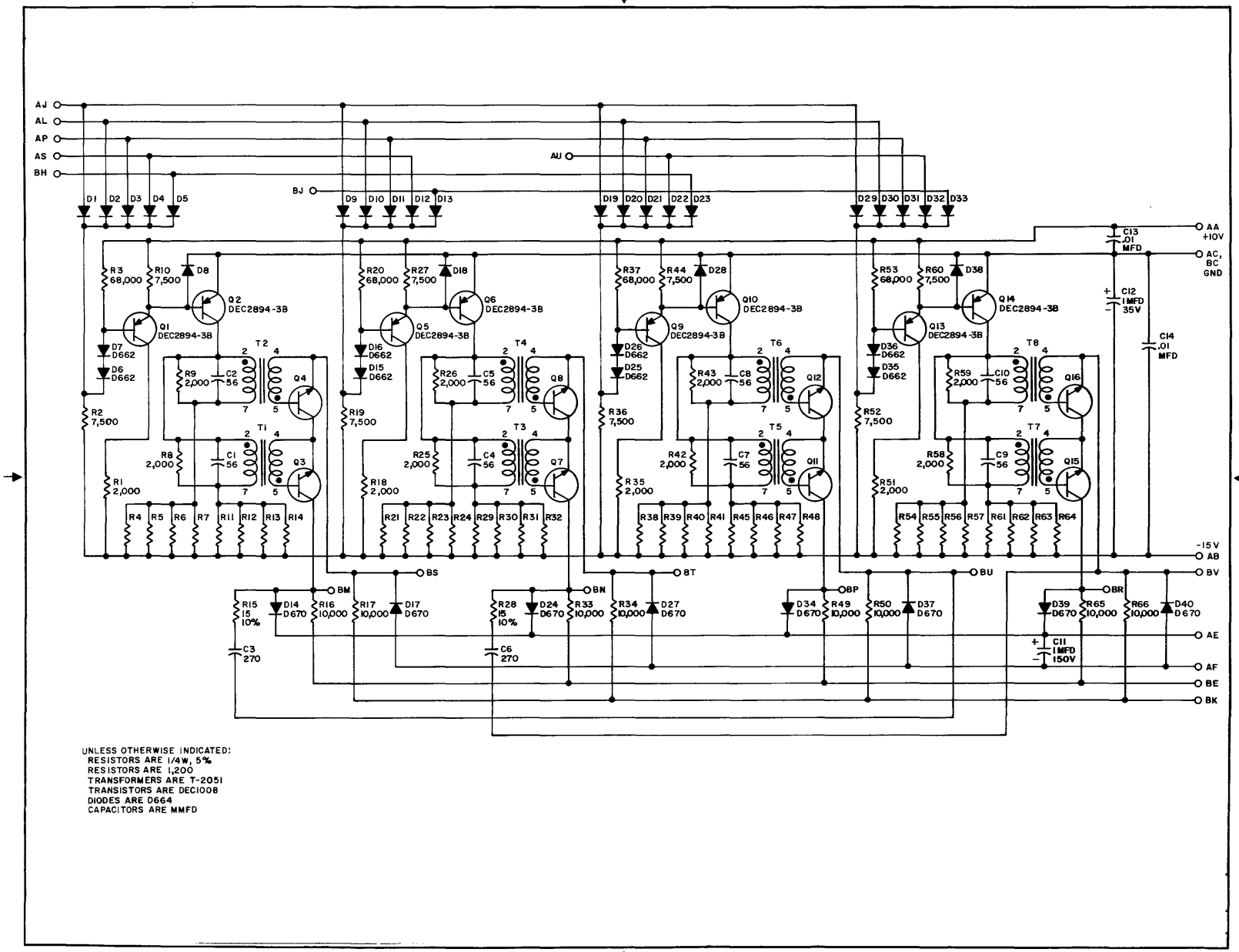
NOTES: * IN429 6.2V 5%
** IN762 6.5V 250mw 5%

CS-C-G008-0-1 Master Slice,
G008. Rev. C

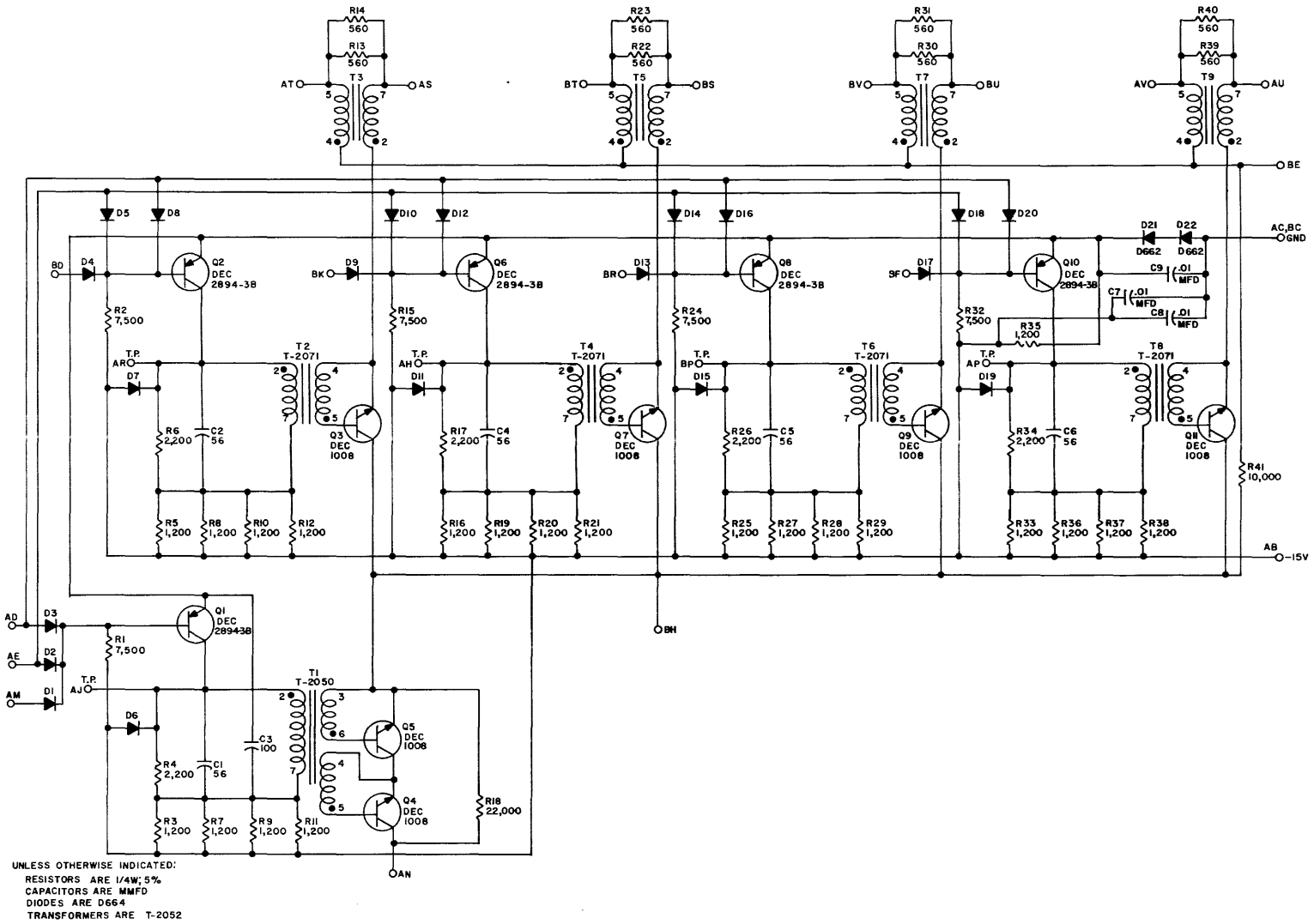


UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 5%
DIODES ARE D662
TRANSISTORS ARE DEC 2894-3A

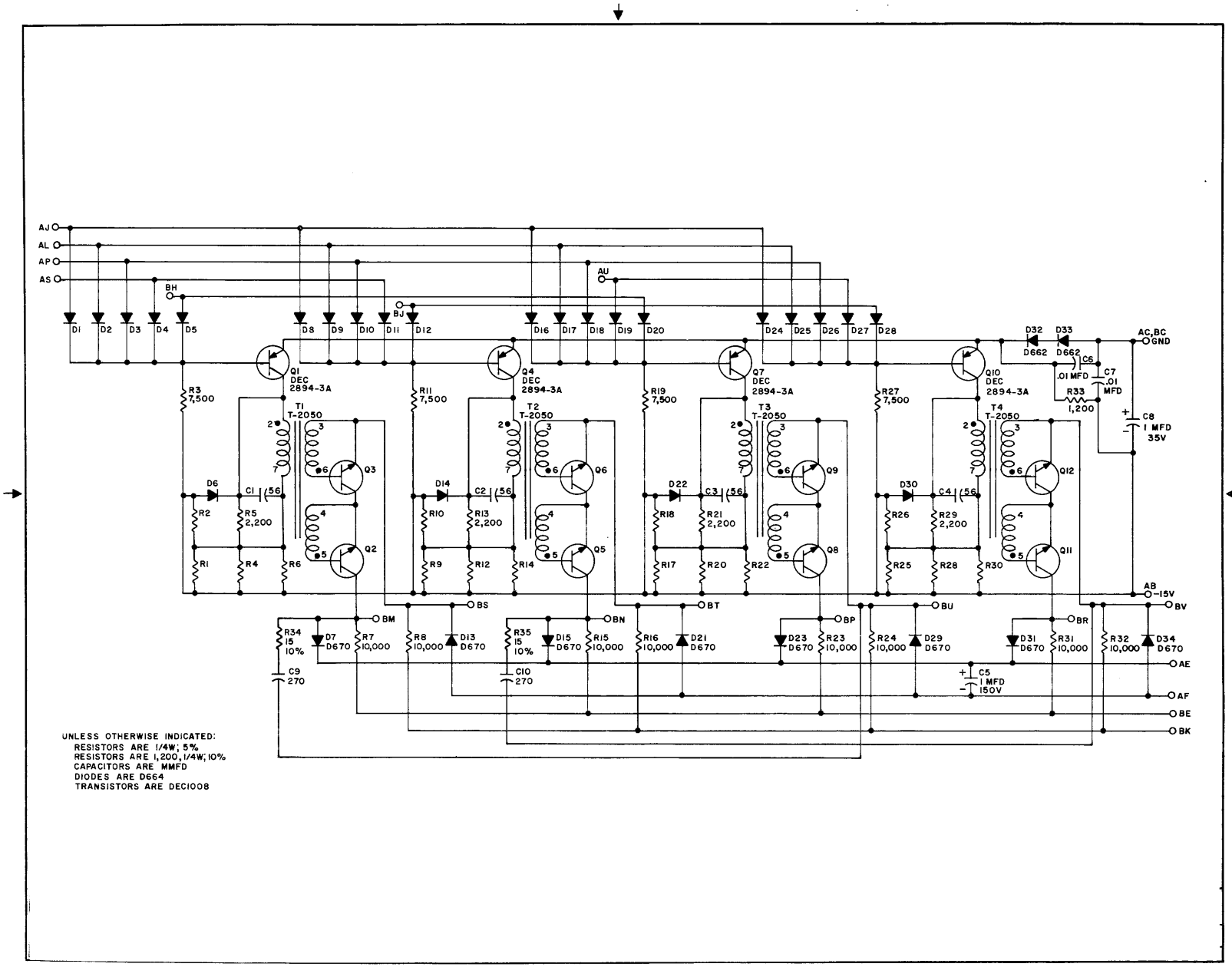
CS-B-G010-0-1 Sense Amplifier
Selector, G010. Rev. A



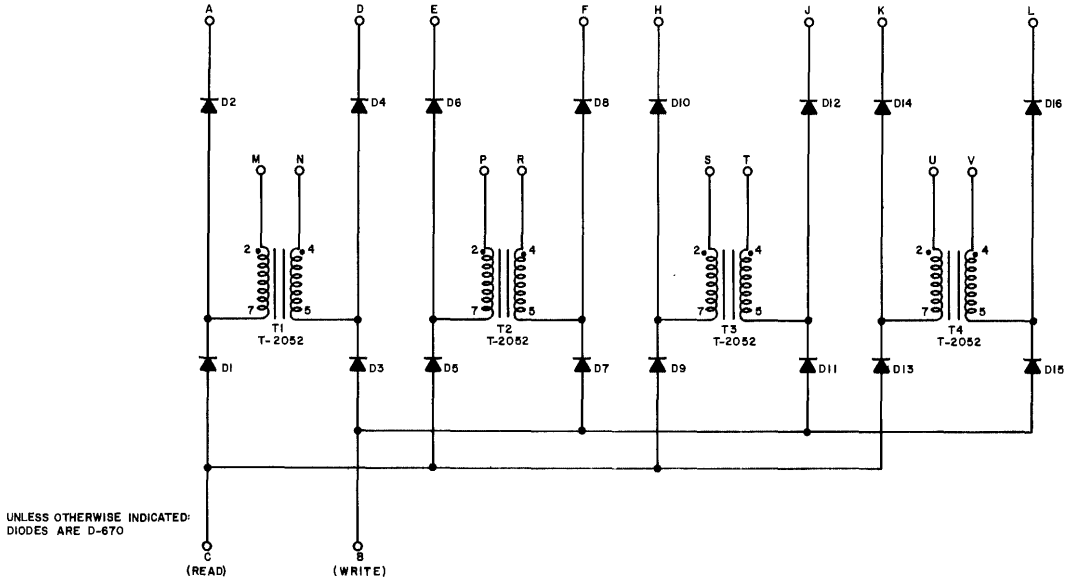
CS-C-G206-0-1 Memory Selector,
G206. Rev. C



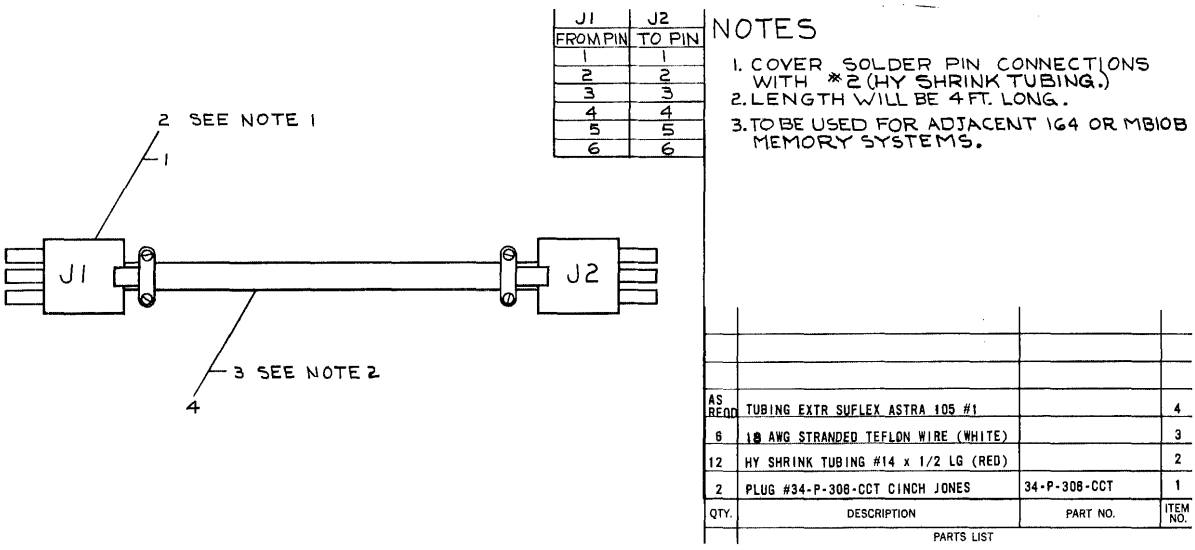
CS-C-G207-0-1 Inhibit Driver, G207. Rev. D



CS-C-G212-0-1 Memory Common Driver, G212



CS-B-G604 Memory Selector Matrix, G604



IA-B-7405423-0-0 Marginal Check and Remote Control Cables for Installation Only. Rev. A

APPENDIX A
TYPE 169 PARITY OPTION

The Type 169 Parity Option is connected across the PDP-6 memory bus (MBUS). It is used to check the parity of each data word transferred between the central processor and the memory. In the write cycle, the parity of each data word being transferred to memory is checked. If the parity of the data word is even, a parity bit is generated and is routed to memory along with the data word. The data word including the parity bit is stored in memory with odd parity.

When a data word is transferred out of memory during a read cycle, the data word is checked for odd parity. If an even parity condition is located, an error condition exists. The parity option detects the error condition and reacts according to the selected operating mode.

The parity option will not check the data word parity or generate a parity bit when a fast memory control is referenced or when a memory plane returns a bypass signal.

A1 INSTALLATION

To install the parity option, the MBUS cable connectors from the central processor are disconnected at the memory and the cables are inserted into the parity option, according to Table A-1. Record the memory slot number from which the cables were removed in Table A-1. Then, install the supplied jumper cables between the parity option and the memory according to Table A-1.

Table A-1
Cable Installation

Processor End	Parity Option End	Parity Option End	Memory End
2L5 (DEC 10360)	1A1 (W028)	1A2 (W028)	
	1B1 (W028)	1B2 (W028)	
2L24 (DEC 10360)	1C1 (W028)	1C2 (W028)	
	1D1 (W028)	1D2 (W028)	
2E25 (DEC 10350)	1A31 (W028)	1A32 (W028)	
	1B31 (W028)	1B32 (W028)	

Table A-1 (cont)
Cable Installation

Processor End	Parity Option End	Parity Option End	Memory End
2J25 (DEC 10350	1C31 (W028) 1D31 (W028)	1C32 (W028) 1D32 (W028)	

A2 CONTROLS AND INDICATORS

The controls and indicators for the parity option are contained on the indicator panel. Figure A-1 shows the indicator panel and Table A-2 contains a brief description of the function of each.

Table A-2
Controls and Indicators

Number	Nomenclature	Type	Function
1	STOP	Micro toggle switch	Selects Normal or Stop-on-Error operating mode.
2	PAR ODD	Indicator	Indicates odd parity
3	CLEAR ERROR	Push button switch	Clears PAR ERR flip-flop
4	PARITY	Indicator	Indicates the presence of a parity bit.
5	RESTART	Push button switch	Clears PAR ERR flip-flop, clears ERROR stop flip-flop, clears Memory Buffer flip-flops and generates RDRS signal.
6	ERROR	Indicator	Indicates a parity error
7	FMC SEL	Indicators	Indicates when a fast memory control is selected.
8	MEMORY BUFFER	36 Indicators	Displays the data word being transferred
9	MEMORY ADDRESS	18 Indicators	Displays the memory address being referenced.

A3 THEORY OF OPERATION

In this section, the circuit operation of the parity option during the write cycle, read cycle, and the read/pause/write cycle is discussed. When only a zone number is referenced, refer to drawing BS-D-169-0-POC. The other references in the text will refer to the drawing number containing the circuits under discussion.

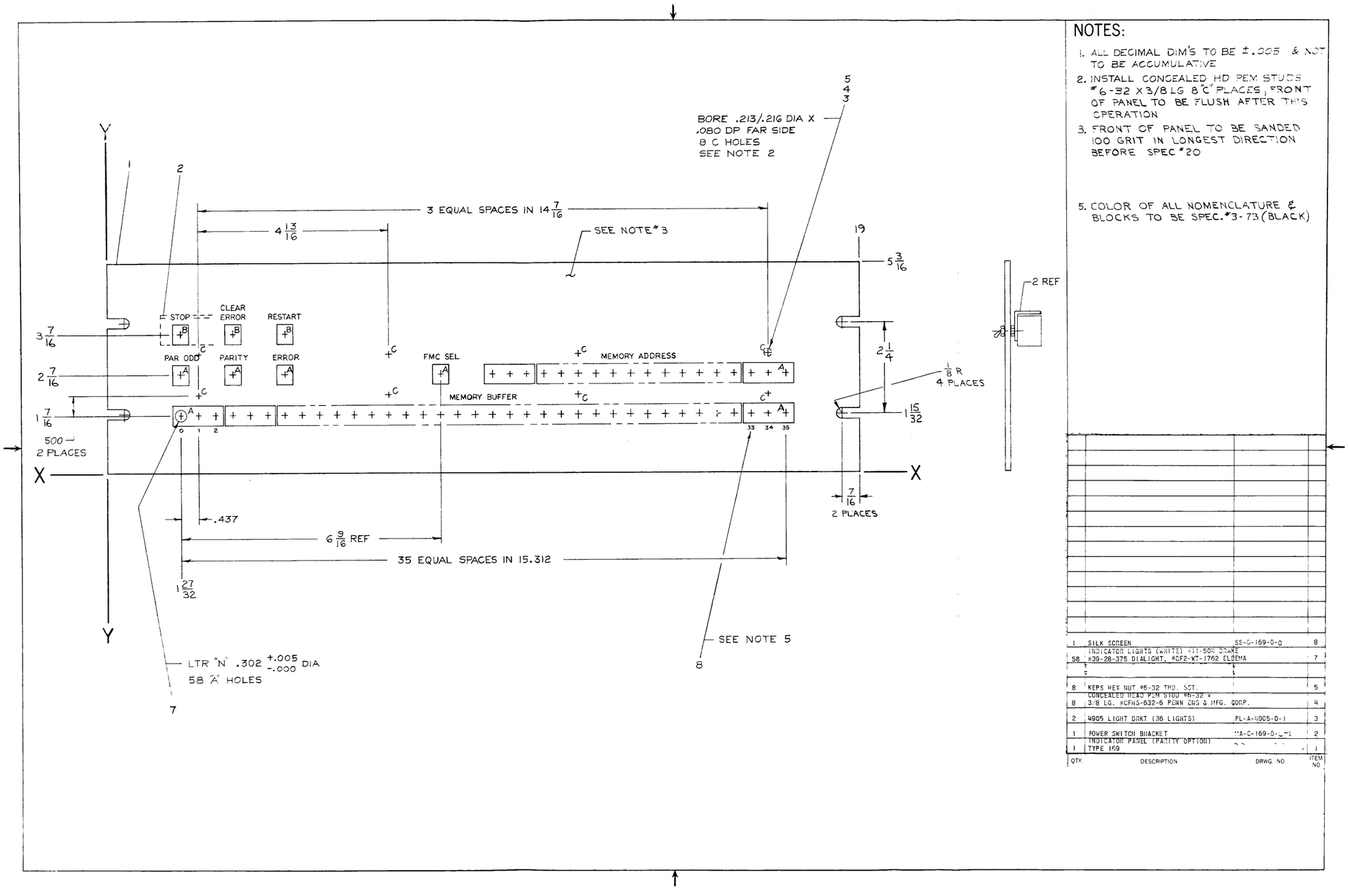


Figure A1 Indicator Panel (Parity Option) Type 169

A3.1 Write Cycle

When the central processor requests a write cycle, the PMA 18 through PMA 35 flip-flops (drawing BS-D-169-0-POMA) are cleared by the CLR PAR MEM ADRS signal (zone A2) which is generated from the PROC RQ signal (zone A1). After a 100-ns delay, the STRB PAR MEM ADRS signal (zone A3) is generated and the PMA flip-flops are set equal to the address presently being referenced.

In addition, at the start of the write cycle, the PAR MB 0 through PAR MB 35 flip-flops (drawing BS-D-169-0-POMB) and the parity flip-flop (zone A7) are cleared by the CLR PAR MEM BUF signal (zone B8) which is generated from the PROC RQ signal (zone B7). Then, the PAR MB flip-flops are connected to the parity option circuit (drawing BS-E-169-0-PO).

In the parity option circuit, the 1 and 0 sides of each PAR MB flip-flops are connected to a parity checking circuit. When odd parity is detected, the desired condition, no parity bit is generated. If even parity is detected, the WR RS (write restart) signal (zone B7), which signifies the end of the central processor write-access time, is ANDed with the PAR EVEN signal. The resultant signal, PAR BIT (zone B7), is transferred via the MBUS and stored in the parity plane of the memory.

A3.2 Read Cycle

When the central processor requests a read cycle, the PMA 18 through PMA 35 flip-flops are cleared and set equal to the address presently being referenced. At the same time, the parity flip-flop and the PAR MB 0 through PAR MB 35 flip-flops are cleared and a data word is placed on the MBUS. If the data word being transferred to the central processor via the MBUS contains a parity bit, the PAR BIT signal (zone B7) is ANDed with the PAR EVEN level and the resultant signal is used to set the parity flip-flop.

After the PAR MB flip-flops are cleared, they are set equal to the data word being transferred to the central processor. The output lines from the PAR MB flip-flops are routed to the parity option circuit along with the output lines from the parity flip-flop. In the parity option circuit, the parity of the data word, including the parity bit, is checked. Since the combination of the data word and the parity bit was stored in the memory with odd parity, the data word being transferred should cause the PAR ODD levels from the parity option circuit to be in the true condition. If an odd number of bits are lost from the data word during transfer, an error condition exists. When an error condition does exist, the generation of the RDRS signal to the central processor (zone B4), which signifies the end of the central processor read access time, is dependent upon the selected operating mode and is explained in Section A3.3.

The RDRS signal (zone B1) from the memory is used to start a 200 ns delay. The resultant signal from the delay is ANDed with the PAR EVEN level. If an error condition does exist, the resultant signal is used to generate the PAR ERROR signal which is routed to the central processor for further action and is used to set the PAR ERR flip-flop (zone A3).

A3.3 Read/Pause/Write Cycle

In this cycle, the read portion of the cycle is performed in an identical fashion as the read cycle explained above. The write portion of this cycle is performed in a slightly different fashion. The main difference between the write portion of the read/pause/write cycle and the write cycle is the way in which the PAR MB flip-flops are cleared. It is recalled that the PROC RQ signal was used at the start of the write cycle to generate the CLR PAR MEM BUF signal. In this cycle, the PROC RQ signal is generated at the start of the read portion and is not present at the start of the write portion. To generate the CLR PAR MEM BUF signal before the write portion of this cycle, the RD RS (read restart) signal starts a 200 ns delay and the resultant signal from the delay causes the CLR PAR MEM BUF signal to be generated, just prior to the write portion of the cycle. The CLR PAR MEM BUF signal clears the PAR MB flip-flops and the parity flip-flop.

A3.4 Operating Modes

The parity option can be operated in a normal mode or a stop-on-error mode. Either mode can be selected with S1 (zone D4 and C5) on the indicator panel. Differences between the two modes are explained below. In this section, circuit operation, when a fast memory control is selected and when the parity option is placed in by-pass, is also explained.

A3.4.1 Stop-on-Error Mode - This mode has no effect on the write cycle. During a read cycle, the RD RS signal (zone C2) from the memory is gated with the output from the parity option circuit and the STOP MODE signal. If an even parity exists, an error condition, the RD RS signal (zone C4) to the central processor is inhibited. At the same time, the PAR ERR, and the ERROR STOP flip-flops are set and the PAR ERROR (zone B3) signal is routed to the central processor. When this occurs, the transfer operation is stopped. The address being referenced and the data word are displayed on the indicator panel. To resume operation, the RESTART pushbutton (S3-zone C1) on the indicator panel is depressed. If the ERROR STOP (1) signal is true, the RESTART pushbutton sends a RD RS pulse to the central processor which restarts the processor. The RESTART pushbutton also starts a 100 ns delay which clears the ERROR STOP flip-flop.

A3.4.2 Normal Mode - In this mode, the write cycle is not affected. During a read cycle, the RD RS signal is returned to the central processor without a delay except for the logic delay. If an error condition does exist, the PAR ERR flip-flop is set and the PAR ERROR signal is returned to the central processor for further action. Normal operation continues without an interruption.

A3.4.3 Fast Memory Control Operation - When a fast memory control is selected, parity of the transfer is not checked and the parity option circuit operation is not affected.

A3.4.4 BYPASS Operation - When the parity plane in the memory is used for storing information other than parity information or when the parity bit is not in the stack the parity option must be bypassed. A jumper is inserted in the memory, when parity is not used, to generate the PAR BYPASS signal that sets the PAR BYPASS flip-flop. The information is then transferred without checking parity.

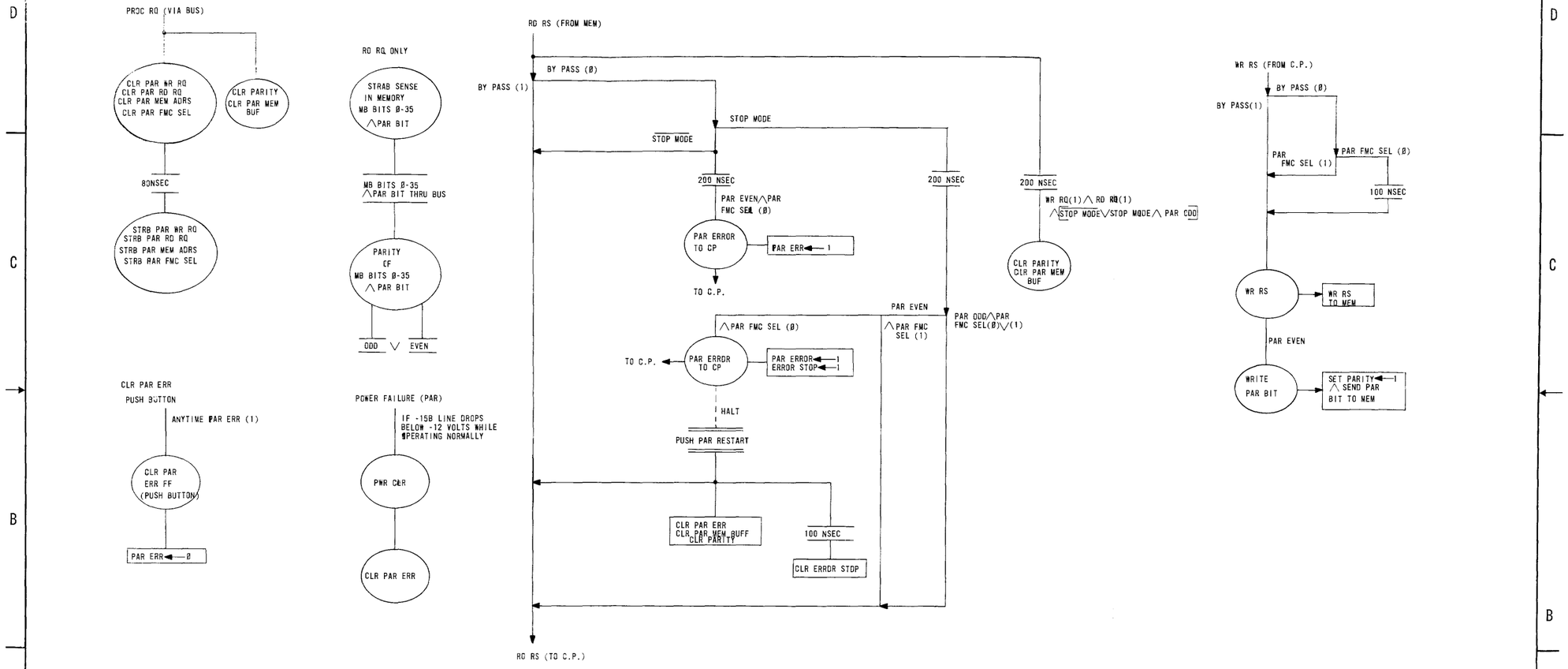
A4 ENGINEERING DRAWINGS

This section contains reduced copies of the engineering drawings required for understanding and maintaining the Type 169 Parity Option. The drawings are in addition to a complete set of full-size drawings supplied with each system. Should any discrepance exist between the drawings in this manual and those supplied with the equipment, assume the full-size drawings are correct.

Standard DEC drawing terminology is used in the drawing contained in this appendix. For an explanation of DEC logic symbology and information on the drawings, the reader can refer to Appendix 1 of the Arithmetic Processor 166 Instruction Manual, F-67 (166).

Type 169 Parity Option Engineering Drawings

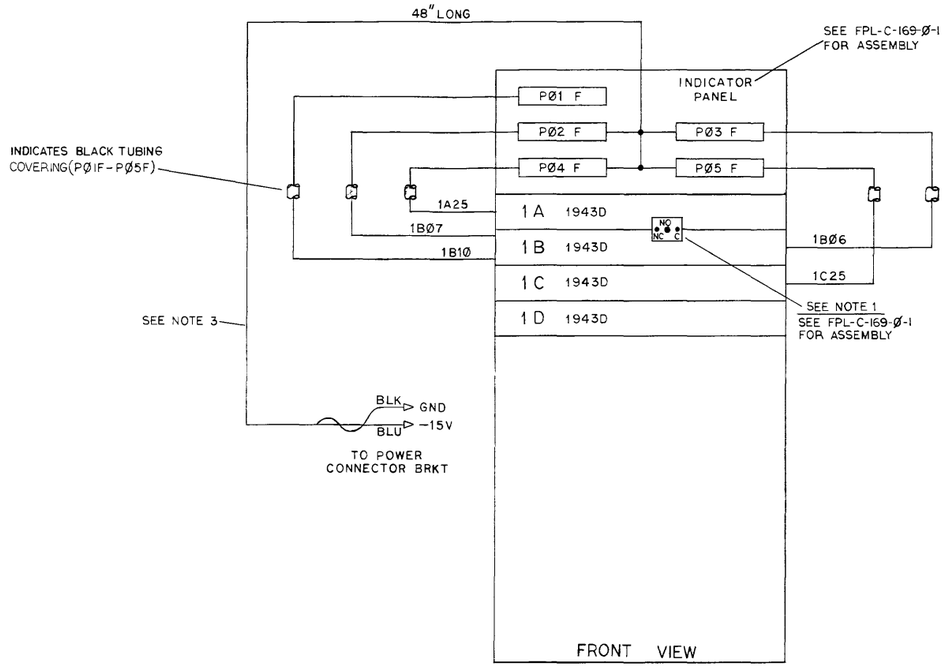
BS-E-169-0-PO
BS-D-169-0-POMA
BS-D-169-0-POMB (2 sheets)
BS-D-169-0-POC
PW-D-169-0-PW
SD-D-169-0-SBD
CD-D-169-0-WLCD
UML-D-169-0-UML
FD-D-169-0-5
MAD-D-169-0-8



D-FD-169-0-5 Memory Parity Option
Type 169. Rev. A

Slot	A	1B	1C	1D
1	W028 ADRS ACK RD RS WR RS PAR BIT RQ CYC MA 18$^{(Ø)}$ MA 18$^{(1)}$ MA 19$^{(Ø)}$ *NOTE: (BUS CABLE IN FROM C.P.)	W028 MA 19$^{(1)}$ MA 20$^{(Ø)}$ MA 20$^{(1)}$ MA 21$^{(Ø)}$ MA 21$^{(1)}$ MA 35$^{(Ø)}$ MA 35$^{(1)}$ MA FMC SELECT B (BUS CABLE IN)	W028 MA 22$^{(1)}$ MA 23$^{(1)}$ MA 24$^{(1)}$ MA 25$^{(1)}$ MA 26$^{(1)}$ MA 27$^{(1)}$ MA 28$^{(1)}$ MA 29$^{(1)}$ MA 30$^{(1)}$ BUS CABLE IN	W028 MA 31$^{(1)}$ MA 32$^{(1)}$ MA 33$^{(1)}$ MA 34$^{(1)}$ MA 35$^{(1)}$ RD RQ WR RQ PAR ERROR MA 21$^{(1)}$ BUS CABLE IN
2	W028 ADRS ACK RD RS WR RS PAR BIT RQ CYC MA 18$^{(Ø)}$ MA 18$^{(1)}$ MA 19$^{(Ø)}$ *NOTE: (BUS CABLE OUT TO MEMORIES)	W028 MA 19$^{(1)}$ MA 20$^{(Ø)}$ MA 20$^{(1)}$ MA 21$^{(Ø)}$ MA 21$^{(1)}$ MA 35$^{(Ø)}$ MA 35$^{(1)}$ MA FMC SELECT B (BUS CABLE OUT)	W028 MA 22$^{(1)}$ MA 23$^{(1)}$ MA 24$^{(1)}$ MA 25$^{(1)}$ MA 26$^{(1)}$ MA 27$^{(1)}$ MA 28$^{(1)}$ MA 29$^{(1)}$ MA 30$^{(1)}$ BUS CABLE OUT	W028 MA 31$^{(1)}$ MA 32$^{(1)}$ MA 33$^{(1)}$ MA 34$^{(1)}$ MA 35$^{(1)}$ RD RQ WR RQ PAR BYPASS MA 21$^{(1)}$ BUS CABLE OUT
3				
4	B113 MA 18$^{(1)}$ MA 19$^{(1)}$ MA 20$^{(1)}$ MA 21$^{(1)}$	W607 RD RS (TO CP) WR RS PAR ERROR	B113 MA 22$^{(1)}$ MA 23$^{(1)}$ MA 24$^{(1)}$ MA 25$^{(1)}$	B113 MA 30$^{(1)}$ MA 31$^{(1)}$ MA 32$^{(1)}$ MA 33$^{(1)}$
5	B204 PMA 18 PMA 19 PMA 20 PMA 21		B113 MA 26$^{(1)}$ MA 27$^{(1)}$ MA 28$^{(1)}$ MA 29$^{(1)}$	B113 MA 34$^{(1)}$ MA 35$^{(1)}$ RD RQ WR RQ
6		W020 MA INDICATORS 18-35	B204 PMA 22 PMA 23 PMA 24 PMA 25	B204 PMA 30 PMA 31 PMA 32 PMA 33
7		W020 PARITY ODD PARITY PAR ERROR PAR FMC SEL INDICATORS	B204 PMA 26 PMA 27 PMA 28 PMA 29	B204 PMA 34 PMA 35 PAR RD RQ PAR WR RQ
8		B105 CLR &STRB MEM ADRS ERROR STOP RD RS PAR ERROR →(1)	B602 CLR PAR MEM ADRS. RD RS ∧ CLR PAR ERROR ∧ CLR PAR MEM BUFFER	B204 PAR FMC SEL PAR BYPASS
9		B360 RD RS (DLY) & (PA)	W501 (RESTART) RD RS ∧ CLR PAR ERROR ∧ CLR PAR MEM BUFFER	B360 STRB PAR MEM ADRS (DLY) & (PA)
10		W028 STOP SWS. CLR PAR ERROR RESTART (CONTROL SWITCHES)	B113 PAR ERROR RD RS RD RS FMC SEL PAR ERROR RD RS	W6Ø7 STRB PAR MEM ADRS
11		B360 SET ERROR STOP FF	B104 RD RS DELAYED RD RS	
12		B204 PAR ERROR FF ERROR STOP FF	B105 CLR PAR MEM BUF PAR ERROR CLR PAR MEM BUFF CLR PAR MEM BUFF PWR CLR	
13		W501 CLR PAR ERROR FF (PB)	B115 CLR PAR MEM BUF CLR PAR MEM BUF	R401 PWR CLR (CLOCK)
14			B104 PAR ERROR WR RS CLR PAR MEM BUF PWR CLR PWR CLR PWR CLR	W505 LOW VOLTAGE DETECTOR
15		B113 WR RS ERROR STOP(1) RD RS CLR PAR MEM BUF STOP MODE WR∧RD RQ	B105 WR RS STOP MODE	
16		B360 WR RS (PA) & (DLY)	B602 PAR BIT CLR PAR MEM BUFFER	
17		W102 PAR BIT PA	W102 WR RS TO MEM RD RS TO CP PAR ERROR TO CP	
18			W102 WR RS TO MEM RD RS TO CP	
19			B130 PAR ODD PAR EVEN	
20			B130 ODD Ø-26 EVEN Ø-26	
21	B130 ODD Ø-8 EVEN Ø-8	B130 ODD 9-17 EVEN 9-17	B130 ODD 18-26 EVEN 18-26	B130 ODD 27-34 & PAR BIT EVEN 27-34 & PAR BIT
22	B130 PAR MB 6 PAR MB 7 PAR MB 8	B130 PAR MB 15 PAR MB 16 PAR MB 17	B130 PAR MB 24 PAR MB 25 PAR MB 26	B130 PAR MS 33 PAR MB 34 PAR BIT
23	B130 PAR MB 3 PAR MB 4 PAR MB 5	B130 PAR MB 12 PAR MB 13 PAR MB 14	B130 PAR MB 21 PAR MB 22 PAR MB 23	B130 PAR MB 3Ø PAR MB 31 PAR MB 32
24	B130 PAR MB Ø PAR MB 1 PAR MB 2	B130 PAR MB 9 PAR MB 1Ø PAR MB 11	B130 PAR MB 18 PAR MB 19 PAR MB 2Ø	B130 PAR MB 27 PAR MB 28 PAR MB 29
25	W020 MB INDICATORS Ø-17	B204 PAR MB 16 PAR MB 17 PAR MB 18 PAR MB 19	W020 MB INDICATORS 18-35	B204 PAR BIT
26	B204 PAR MB 4 PAR MB 5 PAR MB 6 PAR MB 7	B204 PAR MB 12 PAR MB 13 PAR MB 14 PAR MB 15	B204 PAR MB 24 PAR MB 25 PAR MB 26 PAR MB 27	B204 PAR MB 32 PAR MB 33 PAR MB 34 PAR MB 35
27	B204 PAR MB Ø PAR MB 1 PAR MB 2 PAR MB 3	B204 PAR MB 8 PAR MB 9 PAR MB 10 PAR MB 11	B204 PAR MB 2Ø PAR MB 21 PAR MB 22 PAR MB 23	B204 PAR MB 28 PAR MB 29 PAR MB 3Ø PAR MB 31
28		B113 MB 8 MB 17		B113 MB 26 PAR BYPASS MB 35
29	B113 MB 1 MB 3 MB 5 MB 7	B113 MB 10 MB 12 MB 14 MB 16	B113 MB 19 MB 21 MB 23 MB 25	B113 MB 28 MB 30 MB 32 MB 34
30	B113 MB 0 MB 2 MB 4 MB 6	B113 MB 9 MB 11 MB 13 MB 15	B113 MB 18 MB 2Ø MB 22 MB 24	B113 MB 27 MB 29 MB 31 MB 33
31	W028 MB Ø-8 CABLE	W028 MB 9-17 CABLE	W028 MB 18-26 CABLE	W028 MB 27-35 CABLE
32	W028 MB Ø-8 CABLE	W028 MB 9-17 CABLE	W028 MB 18-26 CABLE	W028 MB 27-35 CABLE

UML-D-169-0-UML Memory Parity Option
UML. Rev. A



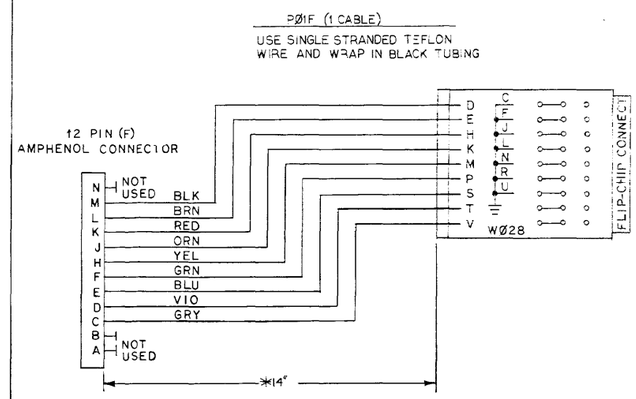
NOTES

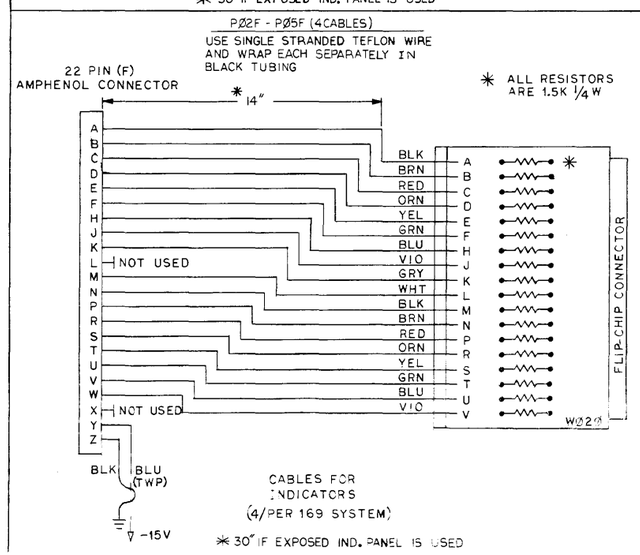
1. MICRO TOGGLE SWITCH (FOR MAINTENANCE PARITY ERROR CHECK) SPDT MOUNT ON BRACKET ABOVE 1B13 & 1B16

2. 4. PØ1F IS A 12 PIN AMPHENOL FEMALE PLUG

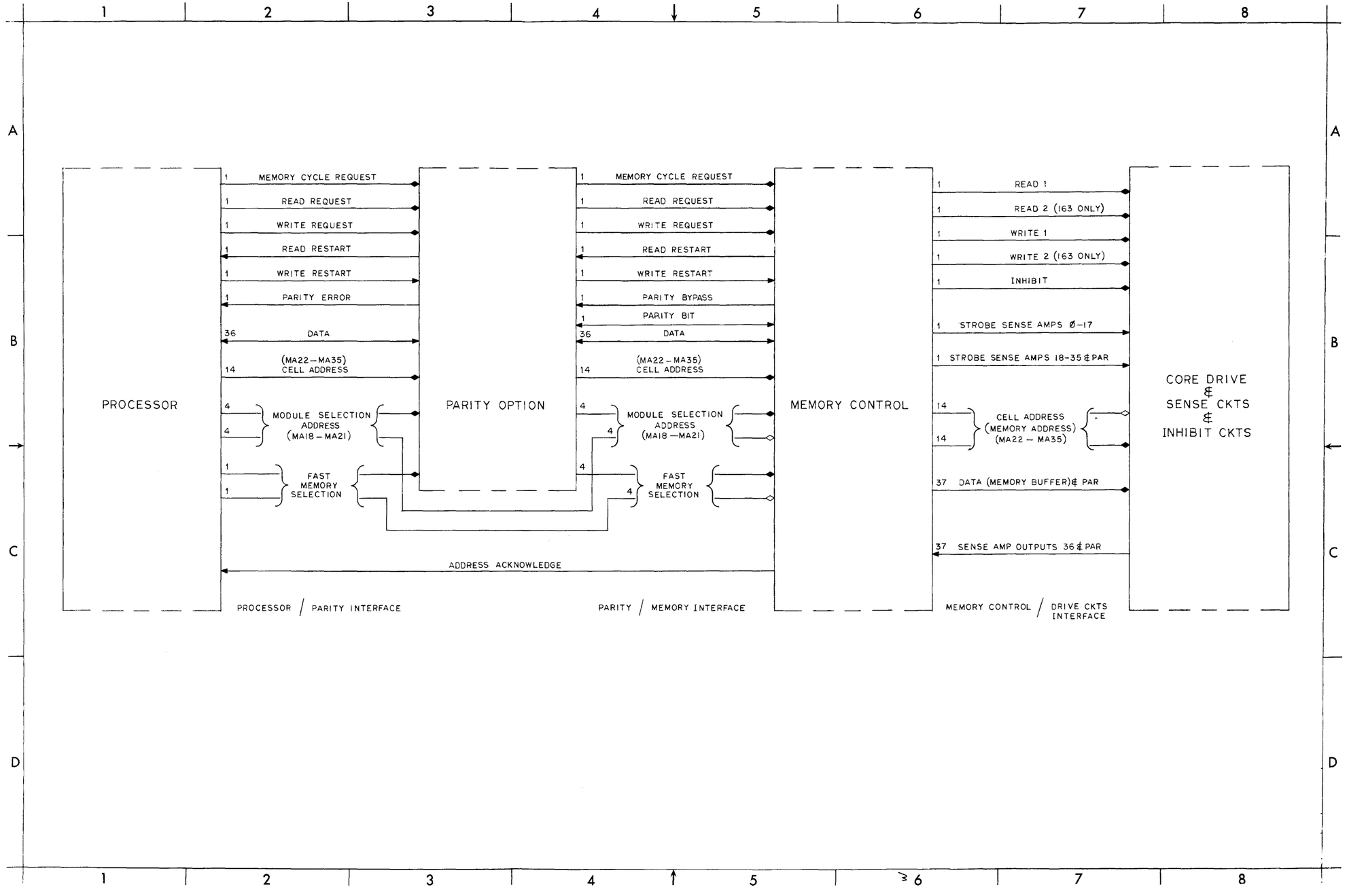
   B. PØ2F THRU PØ5F ARE 22 PIN AMPHENOL FEMALE PLUGS.

3. A. PIN Z OF PØ2F THRU PØ5F ARE BUSSED TOGETHER BY TWP BLK
   B. PIN Y OF PØ2F THRU PØ5F ARE BUSSED TOGETHER BY TWP BLU
   C. WIRE TWP BLU/BLK FROM PØ4F TO POWER CONNECTOR BRKT

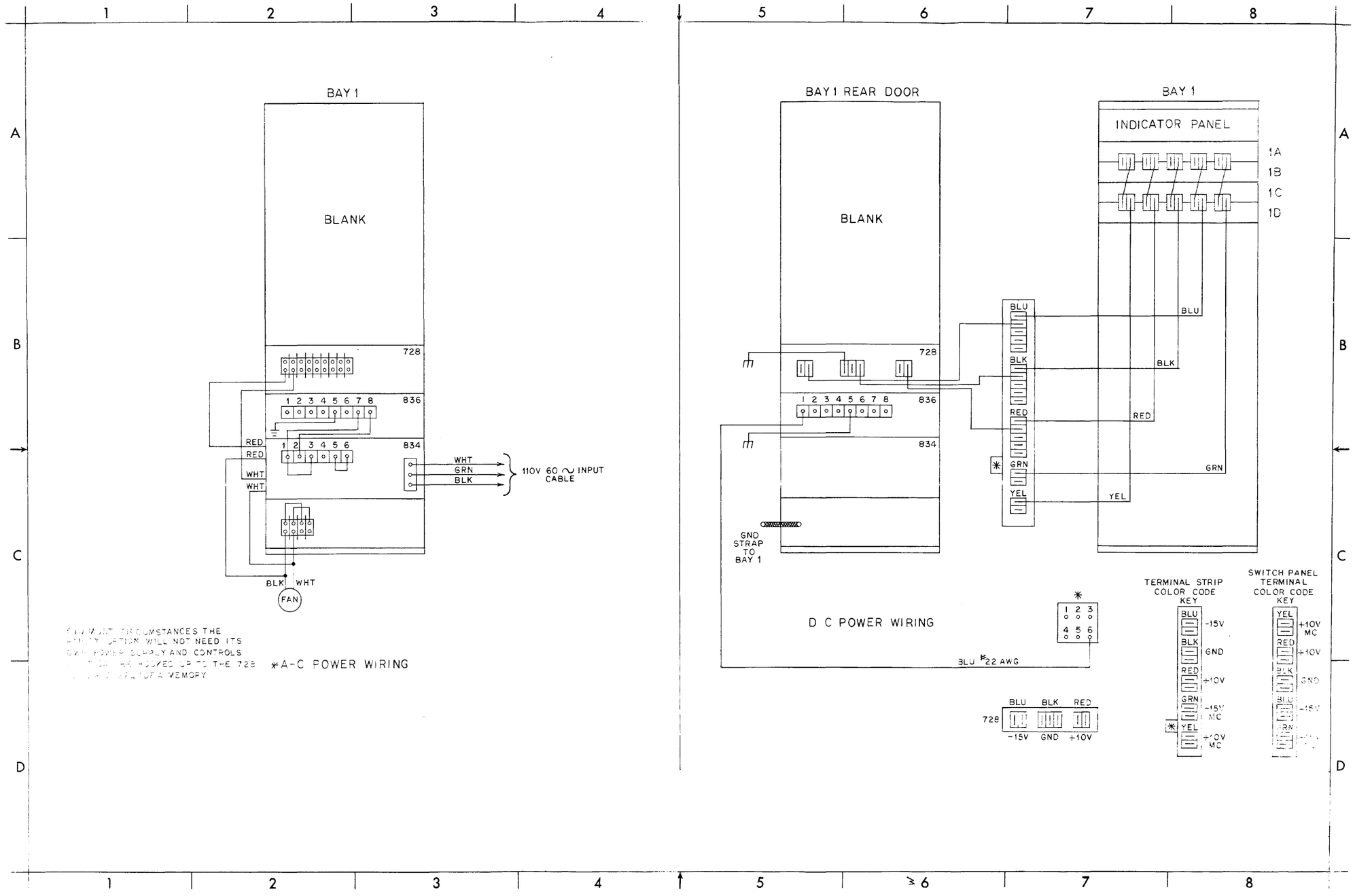




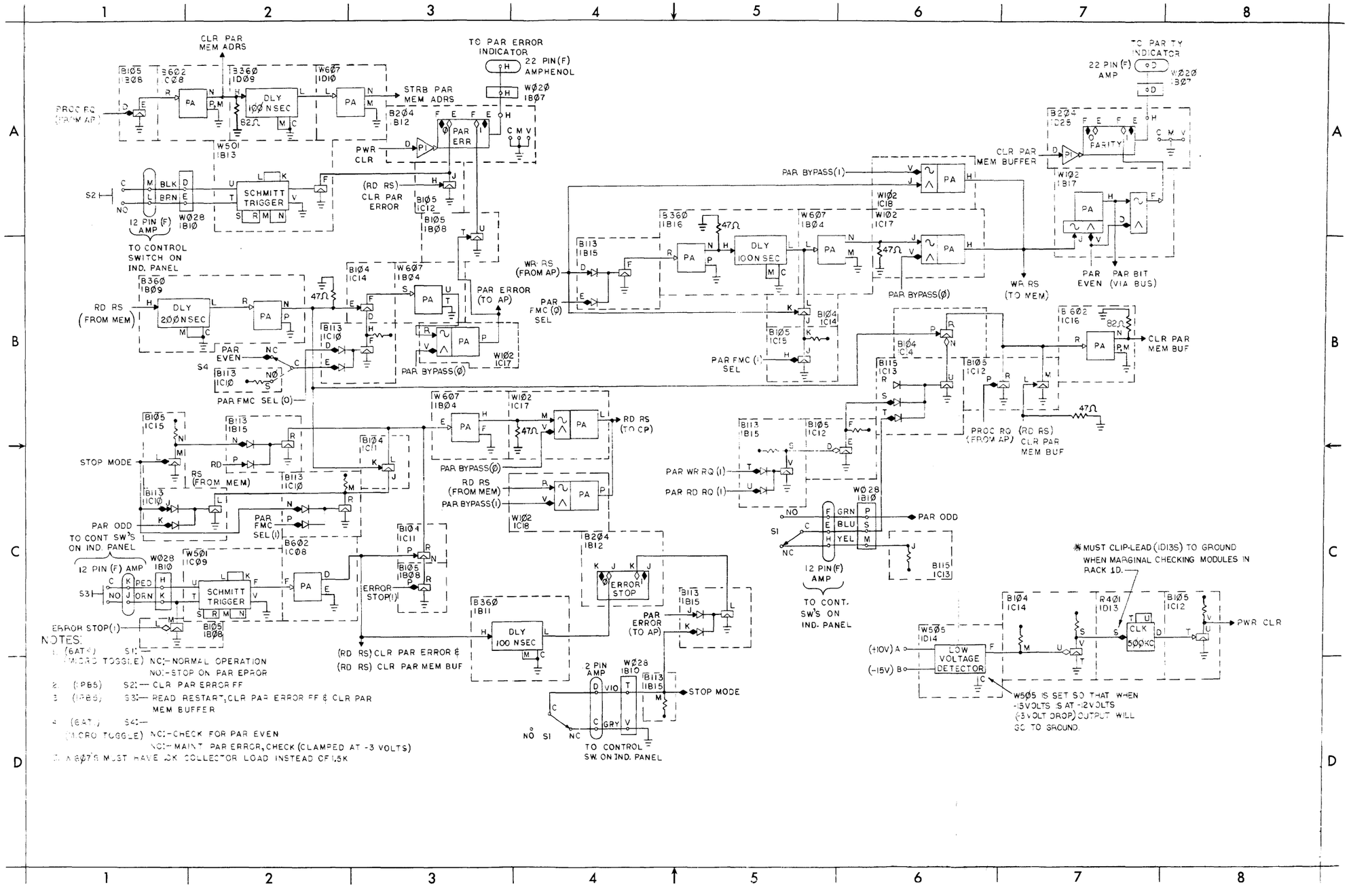
CD-D-169-0-WLCD Indicator & Switches
Cable Diagram Type 169 Parity Option
Rev. A

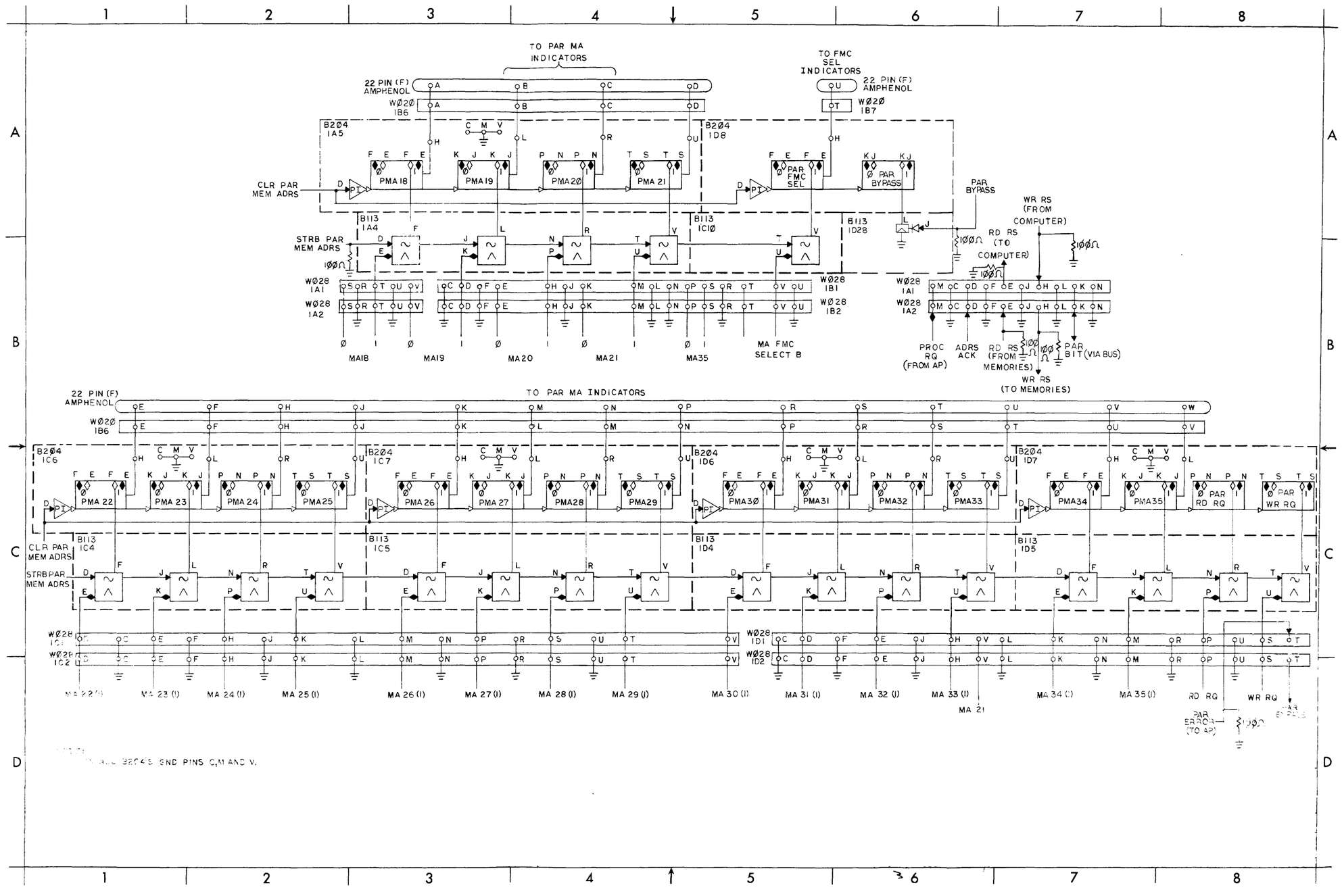


SD-D-169-0-SBD Type 169 Parity Option
System Block Diagram. Rev. A

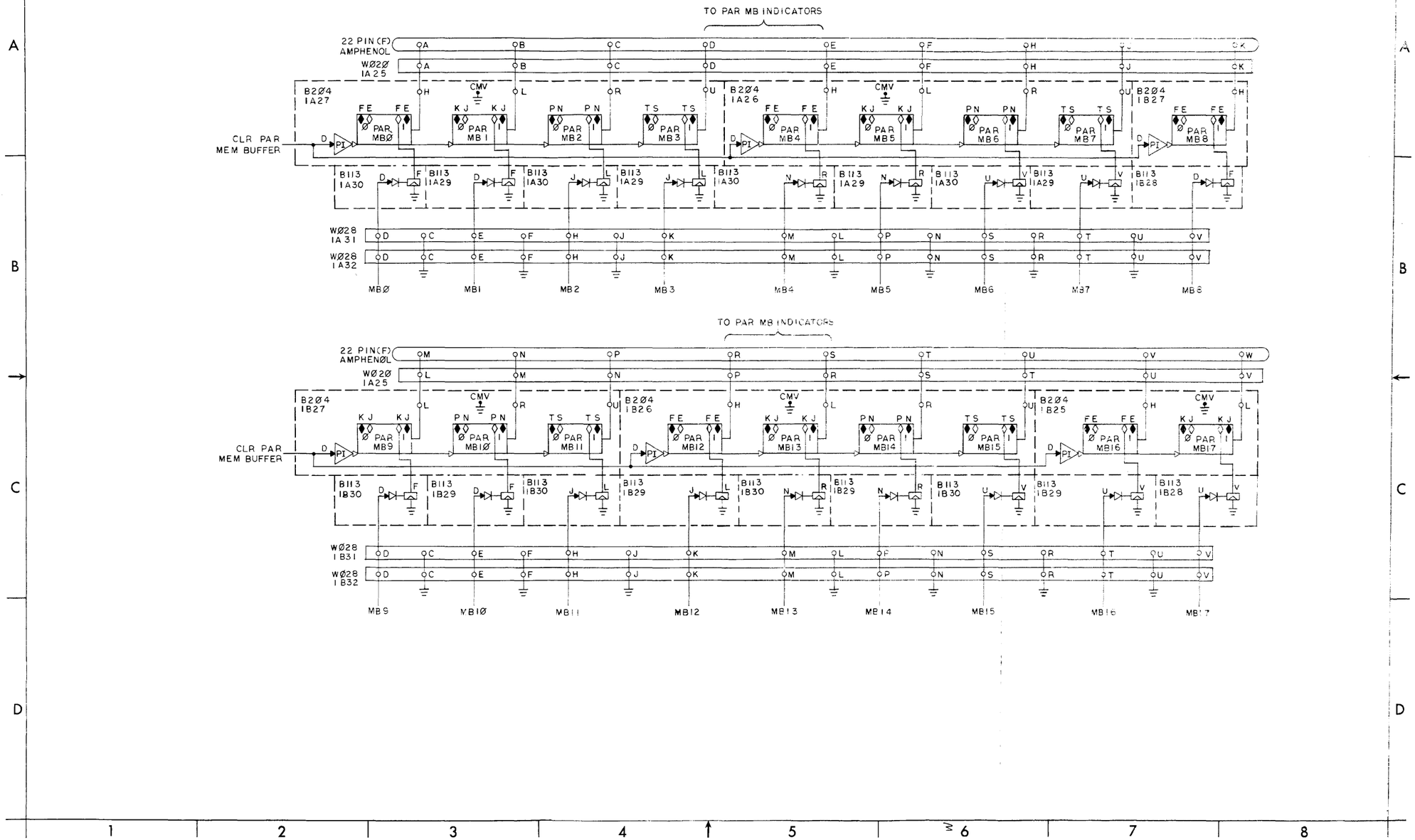


PW-D-169-0-PW AC-DC Wiring
Type 169 Parity Option

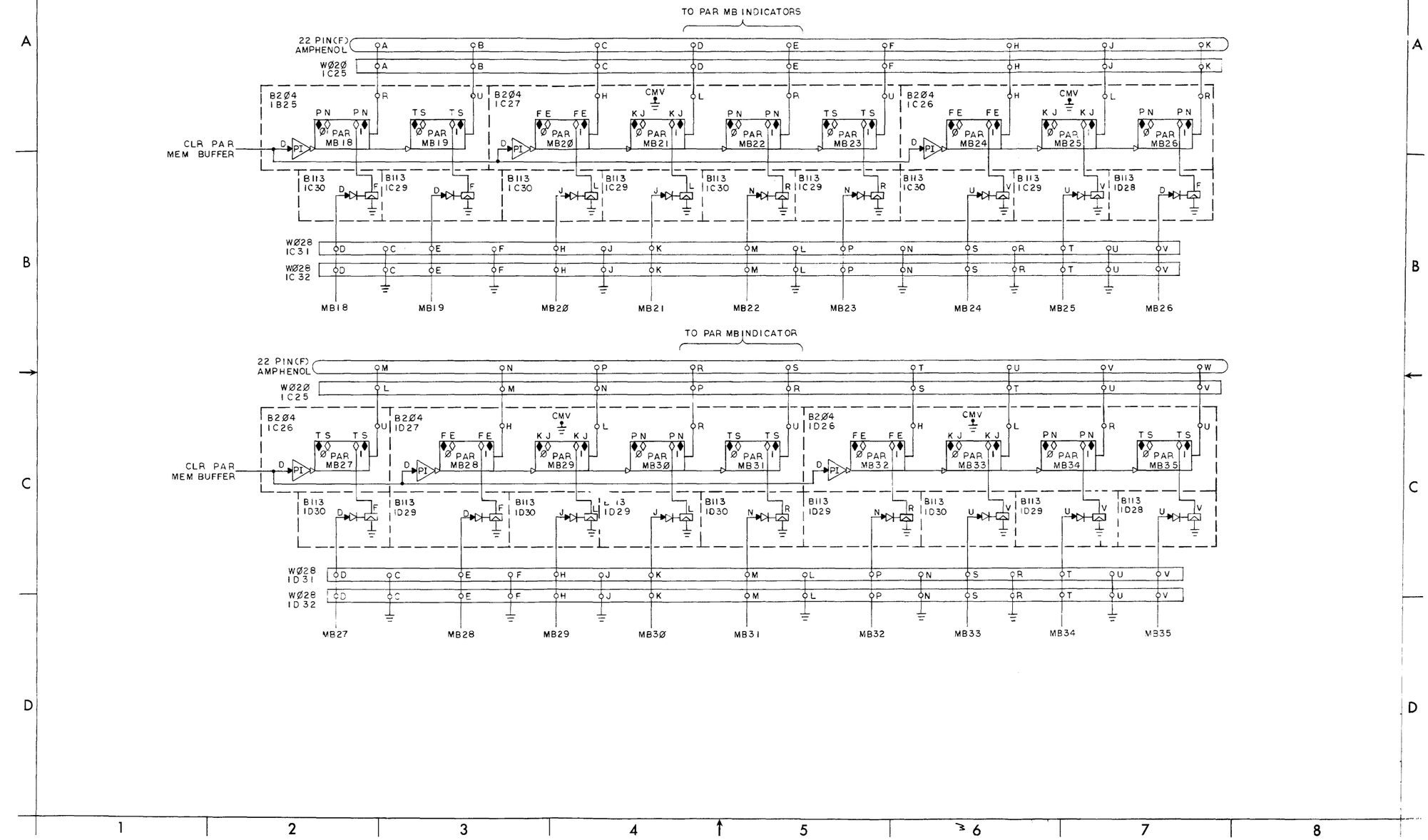

CLR PAR MEM ADRS
TO PAR ERROR INDICATOR
22 PIN(F) AMPHENOL
PROC RQ (FROM AP)
DLY 100 NSEC
STRB PAR MEM ADRS
PAR ERR
PWR CLR
SCHMITT TRIGGER
S2
12 PIN (F) AMP
TO CONTROL SWITCH ON IND. PANEL
(RD RS) CLR PAR ERROR
RD RS (FROM MEM)
DLY 200NSEC
PAR EVEN
S4
PAR FMC SEL (0)
PAR ERROR (TO AP)
PAR BYPASS(Ø)
WR RS (FROM AP)
PAR FMC (Ø) SEL
DLY 100NSEC
PAR BYPASS(1)
CLR PAR MEM BUFFER
PARITY
PAR BIT EVEN (VIA BUS)
WR RS (TO MEM)
PAR FMC (1) SEL
CLR PAR MEM BUF
PROC RQ (RD RS) (FROM AP) CLR PAR MEM BUF
STOP MODE
RS (FROM MEM)
PAR ODD TO CONT SW'S ON IND. PANEL
PAR FMC SEL (1)
RD RS (TO CP)
PAR BYPASS(Ø)
RD RS (FROM MEM)
PAR BYPASS(1)
PAR WR RQ (1)
PAR RD RQ (1)
PAR ODD
12 PIN(F) AMP
TO CONT. SW'S ON IND. PANEL
S3
ERROR STOP(I)
ERROR STOP
2 PIN AMP
TO CONTROL SW. ON IND. PANEL
(RD RS) CLR PAR ERROR &
(RD RS) CLR PAR MEM BUF
DLY 100 NSEC
PAR ERROR (TO AP)
STOP MODE
*MUST CLIP-LEAD (1D13S) TO GROUND WHEN MARGINAL CHECKING MODULES IN RACK 1D.
LOW VOLTAGE DETECTOR
(+10V) A
(-15V) B
PWR CLR
W5Ø5 IS SET SO THAT WHEN -15VOLTS IS AT -12VOLTS (-3VOLT DROP) OUTPUT WILL GO TO GROUND.
NOTES:
1. (6AT4) S1:— (MICRO TOGGLE) NO:—NORMAL OPERATION NC:—STOP ON PAR ERROR
2. (NPB5) S2:— CLR PAR ERROR FF
3. (NPB5) S3:— READ RESTART, CLR PAR ERROR FF & CLR PAR MEM BUFFER
4. (6AT4) S4:— (MICRO TOGGLE) NO:—CHECK FOR PAR EVEN NC:—MAINT. PAR ERROR, CHECK (CLAMPED AT -3 VOLTS)
5. A8Ø79 MUST HAVE 2K COLLECTOR LOAD INSTEAD OF 1.5K



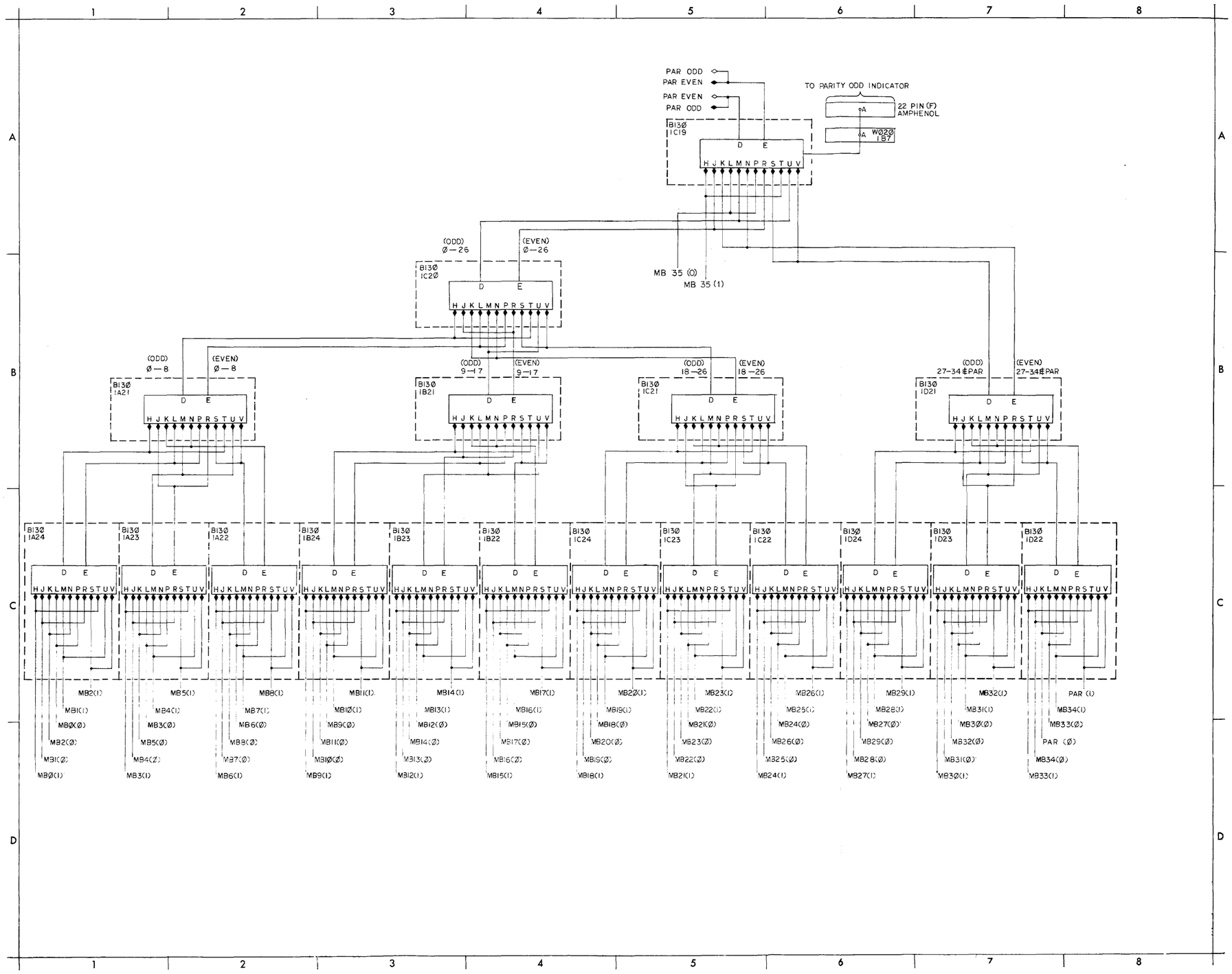
BS-D-169-0-POMA Memory Address Parity
Option Type 169. Rev. A

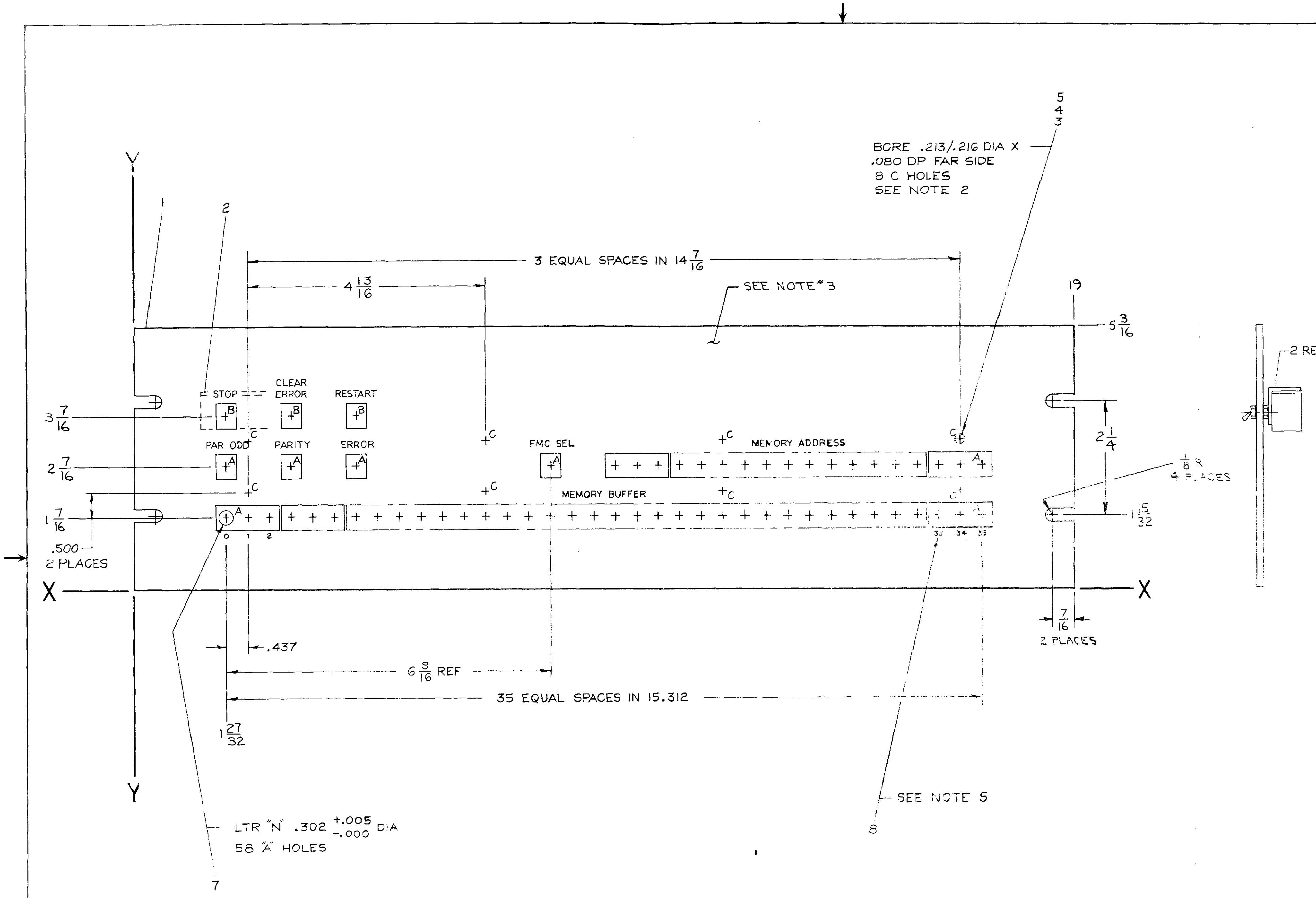


BS-D-169-0-POMB PAR MB Bits 0-17 Memory Parity Option Type 169. Rev. A



BS-D-169-0-POMB PAR MB Bits 18-35 Memory
Parity Option Type 169. Rev. A


PAR ODD
PAR EVEN
PAR EVEN
PAR ODD
TO PARITY ODD INDICATOR
22 PIN (F) AMPHENOL
BI3Ø IC19
W ØZØ IB7
D E
H J K L M N P R S T U V
MB 35 (Ø)
MB 35 (1)
(ODD) Ø—26
(EVEN) Ø—26
BI3Ø IC2Ø
(ODD) Ø—8
(EVEN) Ø—8
BI3Ø IA21
(ODD) 9—17
(EVEN) 9—17
BI3Ø IB21
(ODD) 18—26
(EVEN) 18—26
BI3Ø IC21
(ODD) 27-34 ‡PAR
(EVEN) 27-34 ‡PAR
BI3Ø ID21
BI3Ø IA24
BI3Ø IA23
BI3Ø IA22
BI3Ø IB24
BI3Ø IB23
BI3Ø IB22
BI3Ø IC24
BI3Ø IC23
BI3Ø IC22
BI3Ø ID24
BI3Ø ID23
BI3Ø ID22
MB2(1) MB1(1) MBØ(Ø) MB2(Ø) MB1(Ø) MBØ(1)
MB5(1) MB4(1) MB3(Ø) MB5(Ø) MB4(Ø) MB3(1)
MB8(1) MB7(1) MB6(Ø) MB8(Ø) MB7(Ø) MB6(1)
MB11(1) MB10(1) MB9(Ø) MB11(Ø) MB10(Ø) MB9(1)
MB14(1) MB13(1) MB12(Ø) MB14(Ø) MB13(Ø) MB12(1)
MB17(1) MB16(1) MB15(Ø) MB17(Ø) MB16(Ø) MB15(1)
MB20(1) MB19(1) MB18(Ø) MB20(Ø) MB19(Ø) MB18(1)
MB23(1) MB22(1) MB21(Ø) MB23(Ø) MB22(Ø) MB21(1)
MB26(1) MB25(1) MB24(Ø) MB26(Ø) MB25(Ø) MB24(1)
MB29(1) MB28(1) MB27(Ø) MB29(Ø) MB28(Ø) MB27(1)
MB32(1) MB31(1) MB30(Ø) MB32(Ø) MB31(Ø) MB30(1)
PAR (1) MB34(1) MB33(Ø) PAR (Ø) MB34(Ø) MB33(1)



NOTES:

1. ALL DECIMAL DIM'S TO BE ±.005 & NOT TO BE ACCUMULATIVE
2. INSTALL CONCEALED HD PEM STUDS #6-32 X 3/8 LG 8 "C" PLACES, FRONT OF PANEL TO BE FLUSH AFTER THIS OPERATION
3. FRONT OF PANEL TO BE SANDED 100 GRIT IN LONGEST DIRECTION BEFORE SPEC #20

5. COLOR OF ALL NOMENCLATURE & BLOCKS TO BE SPEC. #3-73 (BLACK)

QTY.	DESCRIPTION	DRWG. NO.	ITEM NO.
1	SILK SCREEN	SS-D-169-0-8	8
58	INDICATOR LIGHTS (WHITE) #11-50N DRAKE #39 28-375 DIALIGHT, #CF2-WT-1762 ELDEMA		7
8	KEPS HEX NUT #6-32 THD. SCT.		5
8	CONCEALED HEAD PEM STUD #6-32 x 3/8 LG. #CFHS-632-6 PENN ENG & MFG. CORP.		4
2	4905 LIGHT BRKT (36 LIGHTS)	PL-A-4905-0-1	3
1	POWER SWITCH BRACKET	MA-C-169-0-[illegible]	2
1	INDICATOR PANEL (PARITY OPTION) TYPE 169	[illegible]	1

MAD-D-169-0-8 Indicator Panel (Parity Option) Type 169

digital

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

Printed in U.S.A.

DEC-06-I5AA-D

INSTRUCTION MANUAL

DRUM PROCESSOR
TYPE 167

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

PROPRIETARY DATA STATEMENT

This manual contains proprietary information. It is provided to the customers of Digital Equipment Corporation to help them properly use and maintain DEC equipment. Revealing the contents to any person or organization for any other purpose is prohibited.

CONTENTS

CONTENTS (continued)

TABLES

FIGURES

ENGINEERING DRAWINGS

ENGINEERING DRAWINGS (continued)

ENGINEERING DRAWINGS (continued)

CHAPTER 1
INTRODUCTION AND DESCRIPTION

1.1 PURPOSE OF EQUIPMENT

The 167 Drum Processor is designed to transfer data directly between an input/output device and the random access memory of a modular PDP-6 system. The drum processor can transfer characters of various sizes at a 4-mhz rate.

1.2 GENERAL DESCRIPTION

In systems using magnetic drums, the drum processor is normally housed in a single-bay cabinet, which also provides space for the Type 236 Magnetic Drum Control. The front panel of the cabinet displays register and status indicators, as shown in figure 1-1, and the cabinet contains an integral power supply and a cooling system. The logic circuits are packaged on standard DEC FLIP CHIP [TM] modules.

1.3 PERTINENT DOCUMENTS

The following documents are pertinent to the study of the material in this manual:

PDP-6 Handbook, F-65

Type 166 Central Processor Maintenance Manual, F-67

Digital System Modules Handbook

FLIP CHIP Modules Handbook

1.4 SPECIFICATIONS

Physical and electrical specifications for the drum processor are listed in section 3-1.

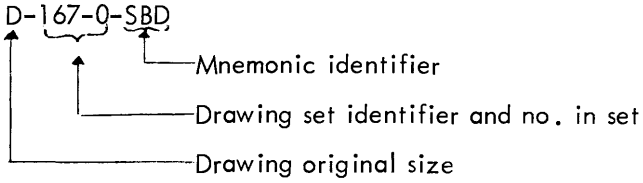
1.5 REFERENCE CONVENTIONS

The Digital Equipment Corporation engineering drawing conventions and instruction manual referencing should be understood at this point. A study of the reference conventions in this paragraph and chapter 6 will save considerable time and preserve thought continuity when reading the text that follows. Any reference to figure numbers or table numbers indicates that the illustration or table is located in text.

[TM] Registered trademark of the Digital Equipment Corporation.

All engineering drawings have a full drawing number. These drawings are included in chapter 6. In text, references to engineering drawings are abbreviated. For example:

Full drawing no. D-167-0-SBD



For this number, the first reference in text is 167-0-SBD; all subsequent references are -SBD.

CHAPTER 2
THEORY OF OPERATION

Discussion of the operation of the 167 Drum Processor (DP), for the purpose of this manual, begins with general system operations. These are illustrated by a DP system block diagram and a comprehensive flow diagram. Detailed treatment of functional units, with particular reference to the engineering drawings in the appendix, follows.

2.1 FUNCTIONAL DESCRIPTION

The primary functional units of the 167 DP in furnishing a direct memory path for high-speed input/output devices are represented in the following diagram (figure 2-1):

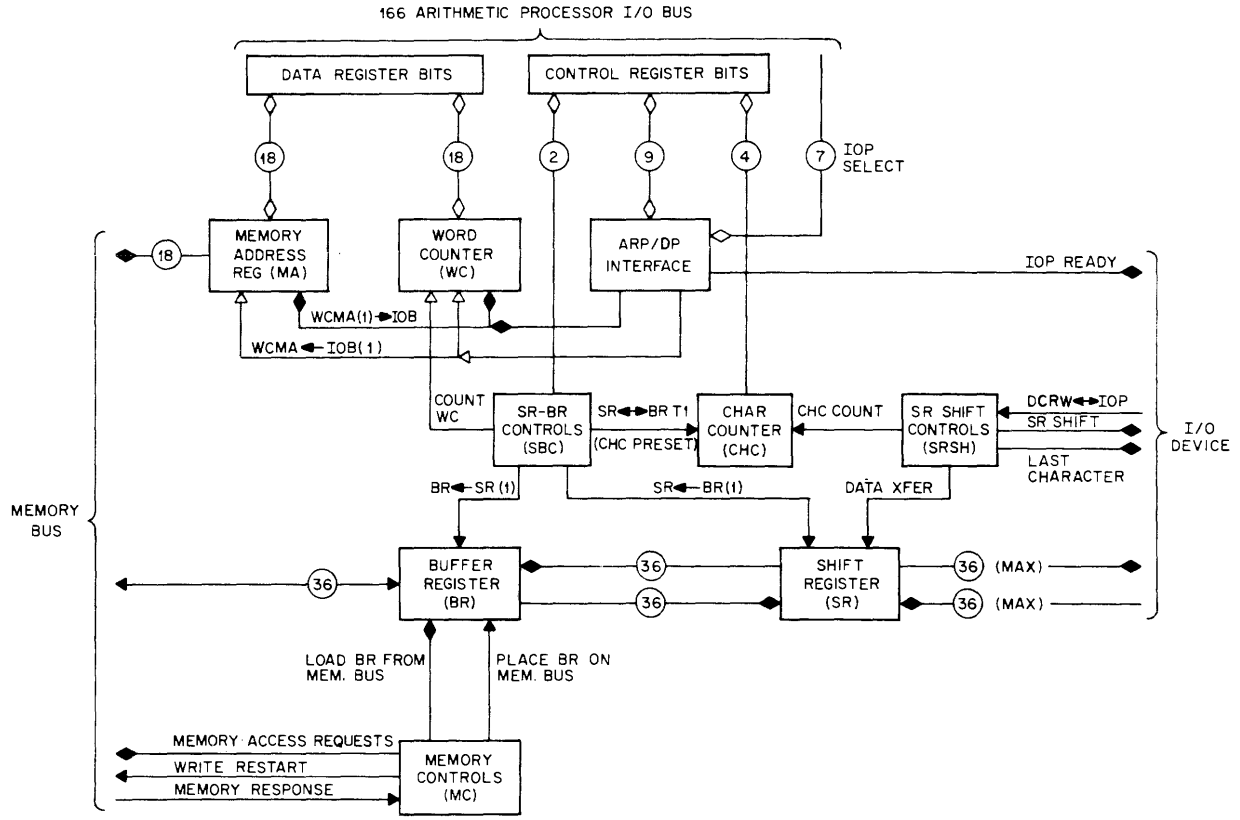
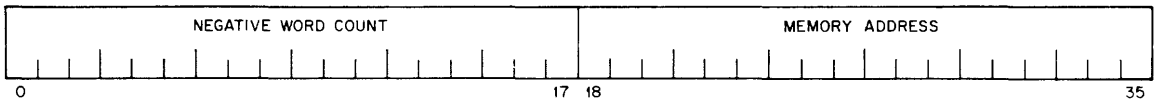


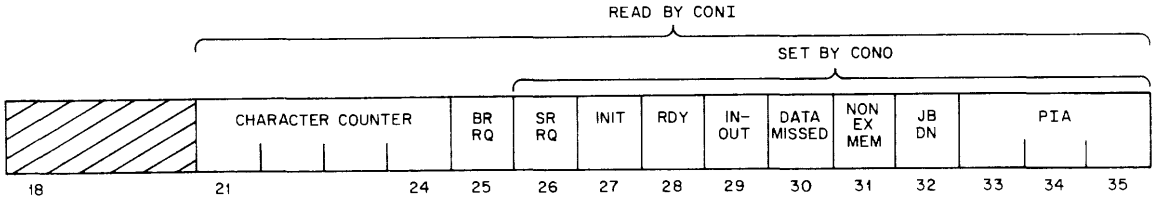
Figure 2-1 Type 167 Drum Processor System Block Diagram

In operation, the volume of data for transfer and the memory addresses for use in the transfer process are established in a 166 Arithmetic Processor (AP) program, noted below in chapter 4. (The data transfer continues asynchronously without regard for further attention from the program which may operate simultaneously in other memory areas.) When the transfer is complete, the 167 DP requests an interrupt. Its units perform the following functions:

a. Interface - The AP/DP interface section sets up the data transfer between the I/O device and system memory under control of the AP program. The interface contains bits 26-35 of the DP control register (figure 2-2), which can be examined by a CONI instruction or set by a CONO instruction. The significance of control register bits is explained in table 2-1 at the end this chapter.



A. DRUM PROCESSOR DATA REGISTER



B. DRUM PROCESSOR CONTROL REGISTER

Figure 2-2 Drum Processor Data and Control Registers

b. Memory Address Register (MA) - This 18-bit register controls the addressing of memory locations required for the data transfer operation, and is incremented by one with each word transferred. The memory address register comprises bits 18-35 of the DP data register (figure 2-2), and is loaded with the starting memory address by a DATAO or BLKO instruction.

c. Word Counter (WC) - This is an 18-bit register loaded with the negative of the number of words to be transferred by a DATAO or BLKO instruction, and is incremented by one with each word transferred. The word counter comprises bits 0-17 of the DP data register. The contents of the word counter and memory address registers may be read at any time by the AP program with a DATAI or BLKI instruction, provided that the RDY flag is reset.

d. Buffer Register (BR) - The buffer register buffers the flow of 36-bit words between the system memory and the shift register, in response to signals from the memory control and the SR-BR controls.

e. Shift Register (SR) - Characters or bytes from the I/O device are assembled into 36-bit words in the shift register for transfer to memory through the buffer register during a memory write operation. During a memory read operation, 36-bit memory words are disassembled in the shift register for character-sequential transfer to the I/O device. Bytes transferred between the DP and the I/O device may assume one of several sizes, as shown in table 3-1.

f. SR Shift Controls (SRSH) - The SR shift controls control word assembly and disassembly in the shift register, and synchronizes character transfer between the DP and the I/O device.

g. Character Counter (CHC) - The character counter counts the characters or bytes transferred between the DP and the external I/O device, to determine when one complete 36-bit word has been transferred. The contents of the character counter prior to each word transfer are determined by a plug-in card, wired to accommodate the number of I/O device characters that make up one memory word. The character counter forms bits 21-24 of the DP control register (figure 2-2), and may be examined by means of a CONI instruction.

h. SR-BR Controls (SBC) - The SBC controls the parallel transfer of 36-bit words between the shift register and the buffer register. Data word transfers are synchronized by the SR RQ and BR RQ flip-flops located within the SBC. These flip-flops form bits 25 and 26 of the DP control register (figure 2-2), and may be examined by a CONI instruction.

i. Memory Controls (MC) - The memory controls synchronize the flow of data between the buffer register and the system memory, and step the memory address register with each word transferred.

2.1.1 System Flow Diagram

In executing a programmed data transfer through the DP 167, as shown in the system flow diagram, drawing D-167-0-FD, the following principle phases are significant (the numbers in parentheses refer to the index numbers on the flow diagram):

a. Initializing the DP (1, 2, 3, 4)

b. Requesting memory access to transfer data between the DP and the system memory (5)

c. Transferring characters between the DP and the I/O device (7)

d. Internal transfer of the data word between the shift register and the buffer register (6)

In the initialization phase, the AP program may elect to read the control and data registers with CONI and DATAI instructions, respectively. Then, when the control and data words returned to the computer indicate that the DP is ready for a programmed data transfer, a CONO instruction may be issued. The CONO instruction causes the AP to send a CONO CLR pulse, followed by a CONO SET

pulse to the interface. The CONO CLR pulse clears the DP control register. The CONO SET pulse enables loading of the control register with the direction of the requested data transfer and the number of the priority interrupt line assigned to the DP by the program.

A DATAO instruction (index no. 4 on the flow chart) may now be issued to clear and reload the word counter and memory address registers, clear the JB DN flag, and enable presetting of the character counter. The system sets certain control flip-flops (explained in table 2-1), and branches to one of two control cycles, depending upon the requested direction of data transfer:

a. IN: Data from the I/O device is to be written into system memory. The first step is to accept sequential characters from the I/O device, and assemble them into a 36-bit word in the shift register (data transfer cycle).

b. OUT: Data is to be read from memory for disassembly and transfer to the I/O device. The first step is to request memory access to read one word (memory request cycle).

After one of the two cycles listed above is executed, a 36-bit data word is present in either the buffer register or the shift register, and the system moves to point 7 on the flow chart to transfer the word to the alternate internal register (BR or SR). The three word-transfer cycles thus generated operate as follows:

2.1.1.1 Memory Request Cycle (Index No. 5) - The memory request cycle is initiated by a transition of the BR RQ flip-flop to the set state. Logic for controlling the memory request cycle is shown in drawing D-167-0-MC. For a memory write operation, the DP sends a RQ CYC signal to memory, accompanied by a WR RQ signal. The core memory responds to the request by decoding the lines from the DP memory address register and initiating a write cycle at the resultant address. The memory then returns on address acknowledge (ADR ACK) pulse to the DP. The DP clears the RQ CYC flip-flop, steps the memory address, and sends a write restart (WR RS) pulse to memory, accompanied by data pulses on 36 lines.

For a memory read operation, the DP sends simultaneous RQ CYC and RD RQ signals to memory, and enables loading of the buffer register from the memory bus. If the memory responds with an ADR ACK within 100 μsec, the MA is stepped and the RQ CYC flag is cleared. A read restart pulse is sent from memory at MC T0 time, indicating that memory has transferred the word to the buffer register.

Data transfer in either direction is completed at MC T1 time, established by a delay line in the memory control section of the DP. The BR RQ and read-or-write request flags are cleared at this time.

2.1.1.2 Data Transfer Cycle (Index No. 5) - The presence of the IOP ~ BUSY level signals the I/O device to start sending or receiving characters. The I/O device responds with a DCRW ⟷ IOP pulse that causes the IOP to

a. assemble the sequential characters from the I/O device if a memory write operation is in progress; or

b. disassemble the 36-bit word in the SR for character-sequential transmission to the I/O device if a memory read operation is in progress.

The character counter is incremented with each character transferred, and the word assembly/disassembly process is stopped when the character counter reaches zero. The SR RQ flag is set at this time.

The word assembly/disassembly process is discussed in detail in the subsections on the buffer and shift registers, the shift register shift controls, and the character counter.

2.1.1.3 Internal Transfer Cycle (Index No. 7) - Word transfer between the buffer register and the shift register is initiated by either

a. a transition of the BR RQ flip-flop from 1 to 0 when the SR RQ flip-flop is in the set state, shown in drawing D-167-0-SBC, which indicates that the BR has been loaded from memory, and the DP responds by transferring the contents of the BR to the SR; or

b. a transition of the SR RQ flip-flop from 0 to 1 while the BR RQ flip-flop is in the set state, which indicates that the SR has been loaded from or unloaded by the I/O device, and the DP responds by transferring the contents of the SR to the BR.

Note that the set states of these request flip-flops (SR RQ and BR RQ) indicate that the corresponding register is ready to accept a word from the alternate internal register (SR or BR). Consequently, transitions of these flip-flops indicate that the corresponding register is ready to transfer a word to the alternate internal register.

The character counter is preset at the start of the internal word transfer cycle. The transfer is completed at time T1, and the INIT flag is checked. If the INIT flag is set, indicating that the first word has just been read from memory, the system transfers the contents of the BR to the SR, and branches to the data transfer cycle. If the INIT flag is clear, the word counter is incremented as the internal transfer is executed. If the word counter is equal to -1, the system sets the JB DN flag, and completes the word transfer operation in progress.

Note that in the output operation, the DP reads the next word from memory while the current word is being disassembled and transferred to the I/O device. Similarly, in the input operation, the DP begins assembly of the next word from the drum while the current word is being written into memory.

2.1.2 System Timing Diagram

A read/write timing diagram for the DP, operating in conjunction with a 163 Core Memory and a 236 Drum Control is shown on drawing D-167-0-RWT. The 236 Drum Control generates and accepts 4-bit characters. Note that the relatively slow word assembly/disassembly process governs the data rate.

The system timing diagram must be studied in connection with the previous description of the system flow chart, and referenced as necessary in the discussions of functional units that follow.

2.2 DETAILS OF FUNCTIONAL UNITS

2.2.1 AP/DP Interface

The communication link between the AP and the DP is set up in the interface section, shown in drawings D-167-0-IOPI 1, 2. Each control signal to the interface (CONO CLR, CONO SET, DATAO CLR, DATAO SET, DATAI, CONI) is accompanied by the DP selection code, formed by the program in bits 3-9 of the I/O instruction word. The selection code is interpreted in the interface, which generates an enabling level, and permits an exchange of data between the control and data registers and the AP.

The interface section contains nine flip-flops, which comprise bits 27-35 of the DP control register (refer to figure 2-2 and table 2-1). The three PIA flip-flops store the number of the priority interrupt line assigned to the DP by the AP program. This number is decoded by a binary-to-octal decoder (A2 and B2), to enable a transition on one of the seven PIA lines whenever one of the three interrupt flip-flops (DATA MISSED, NON EX MEM, JB DN) is set.

2.2.2 Word Counter (WC)

The word counter (drawings D-167-0-WC 1, 2) is an 18-bit register that comprises bits 0-17 of the DP data register. The word counter is loaded from the I/O bus by a DATAO SET pulse to the interface with the number of words to be transferred in negative, 2's complement form. (Thus, three would be represented as octal 777775, and decimal 310 would be represented as octal 777324.) The counter is stepped with each word transferred by a count pulse from the SBC. The count pulse toggles all flip-flops that have been enabled by their associated interstage NAND gates (A18, A24, and A30). Each NAND gate monitors the states of all less significant flip-flops to inhibit toggling of its associated flip-flop until the previous stages have reached their maximum count.

When the word counter reaches -1, a logic level transition is generated at terminal T of A18 to signal the SBC that the last data word is in the DP.

Signals from the interface provide for clearing the word counter (WCMA ← 0), loading the word counter from the I/O bus (WCMA ← IOB(1)), and placing the contents of the word counter on the I/O bus (WCMA (1)→ IOB).

2.2.3 Memory Address Register (MA)

The 18-bit memory address register and its associated memory bus drivers are shown in drawings D-167-0-MA 1, 2, and D-167-0-MAI. The memory address register comprises bits 18-35 of the DP data register. The memory address register is a binary counter similar to the word counter, and is stepped with each data word transferred by count pulses from the memory controls. Signals from the interface clear the word counter, load the word counter from the I/O bus, and place the contents of the word counter on the I/O bus. The contents of the memory address register are statically available to memory via the memory bus.

2.2.4 Buffer (BR) and Shift (SR) Registers

The buffer register and shift register (drawings BRSR 1- 4) buffer the flow of data through the DP under control of signals from the SR-BR controls (drawing SBC), SR shift controls (drawing SRSH), and the memory controls (drawing MC).

The SRSH DATA XFER pulse transfers characters of up to 36 bits from the I/O device to the shift register. To implement an output operation, characters of less than 36 bits must be transferred to the low-order stages of the shift register from the I/O device. Conversely, to implement an output operation, characters of less than 36 bits must be accepted by the I/O device from the high-order stages of the shift register. If, for example, the 236 Drum Control is the I/O device, and data is to be written into memory, the first 4-bit character will be loaded into stages 32-35 of the shift register. The first shift pulse from the SR shift controls (SR SH 4) will transfer this character to SR stages 28-31, to permit loading the next character into stages 32-35, and so on.

The shift register is also capable of shifting on increments of three bits (SR SH 3). Shift register control is discussed below. Signals controlling parallel data word transfers are as follows:

Signal Designation	Origin	Effect
MEM BUS ← BR LT(1)	MC	Contents of BR to memory
MEM BUS EN LT	MC	Memory data word allowed to enter BR
BR ← SR(1)	SBC	Contents of SR to BR
SR ← BR(1)	SBC	Contents of BR to SR

2.2.5 Shift Register Shift Controls (SRSH)

The SRSH shift controls (drawing D-167-0-SRSH) send shift pulses to the SR to control assembly/disassembly of 36-bit data words for transfer between the I/O device and the DP. The SR is shifted in increments of either three or four bits, and at intervals of 100 nsec. Five shift pulses (SRSH T0 - SRSH T4) are available to position each chracter or byte within the shift register, either for character transfer to the I/O device or for word assembly and transfer to the buffer register.

The number of shift pulses used for positioning each byte and the shift line selected for each pulse (SRSH 3 or SRSH 4) are determined by a module. The wiring of the module for the several possible modes of operation is shown on the referenced drawing. The drawing also includes a table showing the shift pulses generated by the SRSH controls for each mode. All five shift pulses, for example, are used to position an 18-bit character within the shift register. The first three pulses are applied on the SH4 line, and the next three pulses are applied on the SH3 line, for a total shift of 18 positions.

A pulse designated DCRW⟷IOP is sent from the I/O device to initiate each shift cycle. During an input operation, this pulse gives rise to a DATA XFER pulse, which loads the shift register with a character from the I/O device. During an output operation, the DCRW⟷IOP pulse indicates that the I/O device has accepted the character placed on the data lines by the IOP.

A pulse is generated to step the character counter after each character is positioned in the shift register. The SHIFT flip-flop is reset at this time, causing the DP to generate a transition on the IOP~BUSY line. This informs the I/O device that the DP is ready for another character, if an operation is in progress.

2.2.6 Character Counter (CHC)

The character counter (drawing D-167-0-CHC) counts down with each character transferred between the DP and the I/O device. When the character counter reaches zero, a CHC = 0 signal is generated, to signal the SBC that the word assembly/disassembly and transfer between the SR and the I/O device is complete. The character counter is preset to the number of characters per word, minus one, by a pulse from the SBC prior to each word transfer operation between the DP and the external device. The number loaded into the character counter is determined by a plug-in module. Wiring of the plug-in module to implement the several possible modes of operation is shown in a table appearing on the referenced drawing.

The character counter forms bits 21-24 of the DP control register, and may be examined by a CONI instruction.

2.2.7 SR-BR Controls

The shift register and buffer register (SBC) controls monitor the status of the buffer register and the shift register to determine

a. If the BR is ready to accept a word from memory

b. If the BR is ready to send a word to memory

c. If the SR is ready to accept a word from the BR

d. If the SR is ready to send a word to the BR

The SR-BR controls contain the SR RQ and BR RQ flip-flops, which define the ready states of the buffer register and the shift register. The sequencing of the SR RQ and BR RQ flip-flops as the DP executes word transfer operations is covered previously in this section in the discussion of the system flow diagram.

The SR-BR controls contain a timing generator, consisting of one 100 nHz delay. The generator times the internal transfer cycle, described earlier in this section and illustrated in the system flow diagram. The SBC also contains logic for clearing the BR and the SR, enabling BR/SR transfer, enabling character counter preset, and stepping the word counter.

2.2.8 Memory Controls (MC)

The memory controls (drawing D-167-0-MC) synchronize the exchange of data words between the buffer register and system memory, and generate a count pulse to step the memory address register with each word transferred. The memory controls execute the memory request cycle, discussed earlier in this section and illustrated in the system flow diagram.

TABLE 2-1 EXPLANATION OF CONTROL REGISTER BITS

Bits	Name	Function
21-24	CHC	Character counter. Contains one less than the number of bytes necessary to complete the current word transfer. Preset to the number of bytes/word minus one for each new word, and is counted down for each byte transferred.
25	BR RQ	Buffer register request. Set on input to memory each time a word has been transferred from the shift register to the buffer register, and reset when the word is accepted by memory. Set on output when initializing, and each time a word is sent to the shift register, unless the word transferred was the last one. Reset when a new word arrives from memory.
26	SR RQ	Shift register request. Set on output from memory when initializing, and when shift register is emptied. Reset when the shift register is reloaded from the buffer register. Set on input when shift register is full, and reset when shift register contents have been transferred to the buffer register. This bit may be set by the AP program to force a jam transfer between the BR and the SR.
27	INIT	Initialize. Set by a DATAO on output. Reset by RDY turning on. Causes first word to be read from memory. Always reset on input to memory.
28	RDY	Ready. Indicates DP ready to transmit data to/from the I/O device. Set on input by a DATAO, and on output by first word arriving in shift register. Reset by occurrence of an interrupt condition (JB DN, NON EX MEM, or DATA MISSED).
29	IN/OUT	A 1 indicates output to the device; 0 indicates input to the memory.
30	DATA MISSED	Set if I/O device attempts to transfer a byte when SR RQ is on. Requests interrupt and turn off RDY.
31	NON EX MEM	Set if non-existent memory location is addressed, or if a memory cycle cannot be obtained within 100 μsec after requested. Requests interrupt and turns off RDY.
32	JB DN	Set when last word has been transferred to the DP on input or output. If an input operation is in progress, the last word will be written into memory before interrupt is acknowledged, provided that the DP has at least as high a memory access priority as the AP. Otherwise, it may be necessary to wait for the BR RQ to clear before issuing another CONO or DATAO to the DP. The JB DN flag requests an interrupt, and may be reset by a DATAO.
33-35	PIA	Priority interrupt assignment. Specifies which of seven program interrupt lines is to be used to inform the AP of an interrupt condition.

CHAPTER 3
INSTALLATION AND INTERFACE

3.1 SPECIFICATIONS FOR 167 DRUM PROCESSOR

The physical and electrical specifications for the processor are listed below.

Dimensions	23-1/2 inches wide 27-1/16 inches deep 69-1/8 inches high
Weight	
Cabinet	160 lbs.
Logic	84
728 Power Supply	20
836 Power Control	18
Indicator panels and blank panels	30
Total	312
Service clearances	8-3/4 inches in front, 14-3/4 inches in back
Color	Raffi and Swansen Tweed, DEC Blue #5150-565
Power	115 vac 60 cps 3-wire single phase; 6-amp starting current, 4-amp operating current
Ambient temperature	50-105° F
Signal cables	8 Type W028 I/O Bus Cables 8 Type W028 Memory Bus Cables 7 Type W020 Indicator Cables 9 Type W028 DP Interface Cables (Lengths of all types above dependent upon system configuration.)

3.2 INSTALLATION

The drum processor requires 115v, single-phase, 60-cps power, and must be operated within an environment defined in section 3.1. Locations of cable connectors are shown in drawing D-167-0-IOCD.

Jumpers on modules on the DP determine the number of characters or bytes from the I/O device that make up one memory word and the number of shifts required to position each byte for transfer between the DP and the I/O device. The modules are normally wired before shipment of the unit. Module wiring is shown in drawing D-167-0-SRSH. The data rates for the several possible byte sizes are shown in table 3-1.

TABLE 3-1 DATA RATES FOR VARIOUS BYTE SIZES

Byte Size	Bytes per Word	Min. Time Between Bytes	Max. Byte Rate
3	12	250 nsec	4.0 hz
4	9	250 nsec	4.0 hz
6	6	350 nsec	2.86 hz
7	5	350 nsec	2.86 hz
8	4	350 nsec	2.86 hz
9	4	400 nsec	2.5 hz
10-12	3	400 nsec	2.5 hz
13-16	2	500 nsec	2.0 hz
17-18	2	600 nsec	1.67 hz
36	1	250 nsec*	4.0 hz

3.3 INTERFACE

Drawing D-167-0-IOSD gives a functional picture of all signals passing between the DP and the AP, the core memory, and the connected I/O device. Drawing D-167-0-IODI shows the pin assignments for each of the interface signals between the DP and the I/O device. In table 3-2, functions of all interface signals are explained.

* Limited also by memory speed.

TABLE 3-2 167 DP INTERFACE SIGNALS

Type	Name	Function
	POWER ON -15v	Turn on DP power. Present whenever AP power is on.
	POWER RESET	Clears control register. Present (a) Whenever the I/O reset key on the computer console is pressed. (b) Whenever the program gives a CONO instruction to the priority interrupt system to clear the system. (c) Whenenver the power is turned on.
Signals Between 166 AP and 167 DP	IOS 3-9 (0/1)	Carry 167 selection code. Set up 1 μsec before any other command signal occurs.
	DATAO CLR,	Sequential pulses generated by DATAO command.
	DATAO SET	Clears and loads data register.
	CONO CLR,	Sequential pulses generated by CONO command.
	CONO SET	Clears and loads control register.
	IOB ← DATA	Places data register bits on I/O bus.
	IOB ← STATUS	Places control register on I/O bus.
	PIA 1-7	A signal is present on one of these lines when an interrupt request occurs.
	IOB 0-35	AP input/output bus lines.
	DCRW ⟷ IOP	Indicates I/O device is ready to send/receive one character (byte).
Signals Between DP and I/O Device Note All data levels are -3v for a logic 1.	IOP READY	Data is available at the DP, or the DP is ready to accept data.
	SR LAST CHAR	Sent by DP to indicate DP has only two characters to send or receive to complete operation in progress.
	IOP ~BUSY	Signals I/O device whether an I/O operation is in progress.
	DATA IN 0-35	Input data lines available to I/O device.
	DATA OUT 0-35	Output data lines available to I/O device.

TABLE 3-2 167 DP INTERFACE SIGNALS (continued)

Type	Name	Function
Signals Between DP and Memory	RQ CYC	Read cycle-DP requests memory access for a read or write operation.
	RD RQ	Read request-DP requests memory read operation.
	WR RQ	Write request-DP requests memory write operation.
	ADR ACK	Address acknowledge-returned by memory within 0.2 μsec in response to RQ CYC signal if memory module is free.
	RD RS	Read restart-indicates memory has transferred one word to DP (sent by memory).
	WR RS	Write restart-sent by DP to indicate that next word is available.
	MA (18-35)	Memory address lines from DP.
	MB (0-35)	Memory bus lines for bidirectional transfer of data.

CHAPTER 4
OPERATION AND PROGRAMMING

4.1 PROGRAMMING

The AP program must condition the DP for a data transfer before initializing the connected I/O device. The DP is set up by a CONO instruction to specify the direction of data transfer, followed by a DATAO instruction specifying the number of words to be transferred and the initial memory address selected for the transfer. (See figure 2-2 for the configurations of the DP control and data registers.)

As an example, the following symbolic instructions may be used to set up the DP for output:

```
CONO    DP,    OUT
DATAO   DP,    WCMA
```

The location OUT might contain the octal configuration 101, to set the DP output bit and allocate priority interrupt channel number 1 for the DP. Similarly, location WCMA contains the desired word count in negative 2's complement form, plus the first memory address to be used. The mnemonic DP causes the PDP-6 assembler (MACRO-6) to generate the DP selection code in bits 3-9 of the CONO and DATAO instruction words.

The AP program to service DP interrupts should include a CONI instruction to enable examination of bits 30, 31, and 32 of the DP control register. The program can then respond appropriately to the condition that caused the interrupt (DATA MISSED, NON EX MEM, or JB DN).

CHAPTER 5
MAINTENANCE

Since the Type 167 Drum Processor must operate with a Data Processor Type 166, a full knowledge and understanding of the descriptions of chapter 9 of Type 166 Arithmetic Processor Maintenance Manual form the basis of maintenance instructions. Many of the drum processor maintenance procedures can be conducted using Maindec programs associated with the 166.

Suggested tools and test equipment suitable for use while performing maintenance procedures on the 167 Drum Processor are listed in chapter 9 of the Type 166 Maintenance Manual. The principal maintenance tool is a detailed understanding of the principles of operation of the drum processor, as outlined in chapter 2 of this manual. Diagnostic programs (Maindecs) are provided to check on-line operation of the DP. For more specific checks, the maintenance technician may use the PDP-6 console controls and indicators to exercise the communication links between the DP and the AP, the DP and the system memory, and the DP and the connected I/O device. The DP front panel indicators allow visual monitoring of the DP registers as instructions are executed in the single step mode.

CHAPTER 6
ENGINEERING DRAWINGS

This section contains reduced copies of the engineering drawings and replacement schematics required for understanding and maintaining the 167 Drum Processor. These reduced copies are in addition to the complete set of full-size drawings forwarded with each DP. Maintenance personnel should use only the full-size drawings for work on units. The replacement schematics are furnished only for test and maintenance purposes. These circuits are proprietary in nature and should be treated accordingly.

6.1 CIRCUIT SYMBOLS

The block schematics of DEC equipment are multipurpose drawings that combine signal flow, logical function, circuit type and physical location, wiring, and other pertinent information. Individual circuits are shown in block or semiblock form, using special symbols that define the circuit operation. These symbols are similar to those appearing in "The Digital Logic Handbook" but are often simplified. Figure 6-1 illustrates some of the symbols used in the DEC engineering drawings for the 167 Drum Processor.

6.1.1 Logic Levels

The standard DEC logic level is either ground (0 to −0.3v) or −3v (−2.5 to 3.5v). DEC logic levels are indicated by an open diamond (——◇) for a ground level and by a closed diamond (——◆) for a negative (−3v) level. The DEC logic symbols shown at the inputs of most circuits indicate the enabling conditions required to produce a desired output; e.g., an open diamond at a circuit input indicates that a ground level is required to produce the desired output.

6.1.2 FLIP CHIP Pulse

Two types of pulses, R series and B series, are used in FLIP CHIP circuit operation. The pulse produced by R-series modules (see figure 6-2) starts at −3v, goes to ground (−0.2v) for 100 nsec, then returns to −3v.

The B-series negative pulse is 2.5v in amplitude and 40 nsec in width as shown in figure 6-3. If this pulse is applied to the base of an inverter, the inverter output will be a narrow pulse, similar in shape to the R-series standard pulse. The B-series positive pulse, which goes from ground to +2.5v, is the inverse of the B-series pulse.

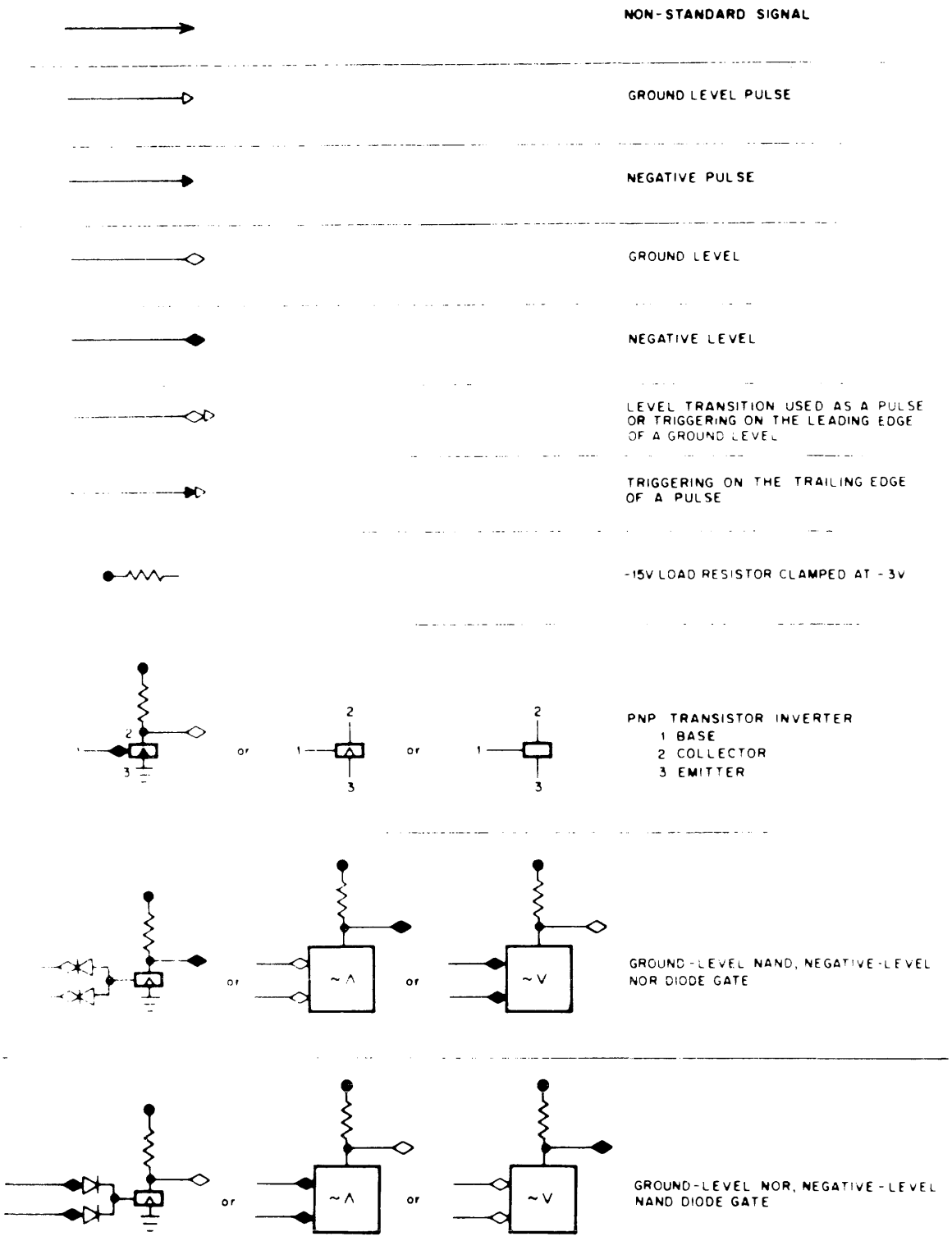
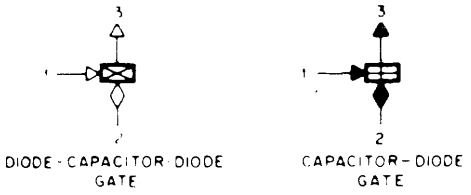
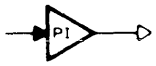


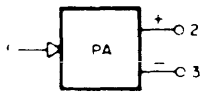
Figure 6-1 DEC Symbols



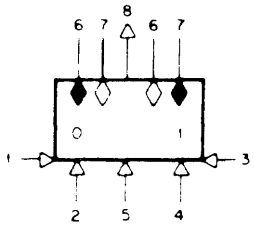
1. PULSE INPUT
2 CONDITIONING LEVEL INPUT
3 PULSE OUTPUT



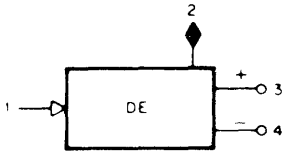
PULSE INVERTER



PULSE AMPLIFIER
1 PULSE INPUT, POLARITY INDICATED BY INPUT SIGNAL
2,3 TRANSFORMER-COUPLED PULSE OUTPUT EITHER TERMINAL MAY BE GROUNDED



FLIP-FLOP (MOST FLIP-FLOPS HAVE ONLY SOME OF THE FOLLOWING):
1 DIRECT CLEAR INPUT
2 GATED-CLEAR INPUT
3 DIRECT-SET INPUT
4 GATED SET INPUT
5 COMPLEMENT INPUT
6 OUTPUT LEVEL, -3 V IF 0, 0 V IF 1
7 OUTPUT LEVEL, 0 V IF 0, -3 V IF 1
8 CARRY PULSE OUTPUT, UPON BEING CLEARED



DELAY (ONE-SHOT MULTIVIBRATOR)
1 INPUT PULSE
2 OUTPUT LEVEL, -3 V DURING DELAY
3,4 TRANSFORMER-COUPLED PULSE OUTPUT EITHER TERMINAL MAY BE GROUNDED

Figure 6-1 DEC Symbols (continued)

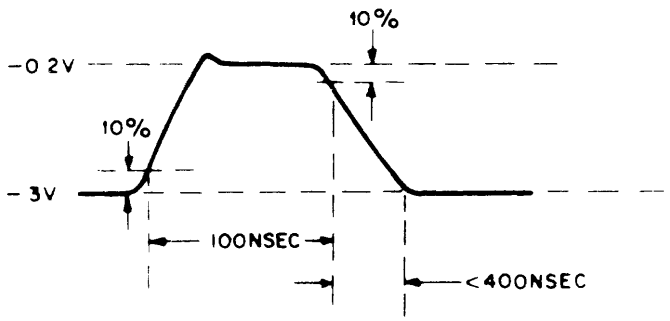


Figure 6-2 FLIP CHIP R-Series Standard Pulse

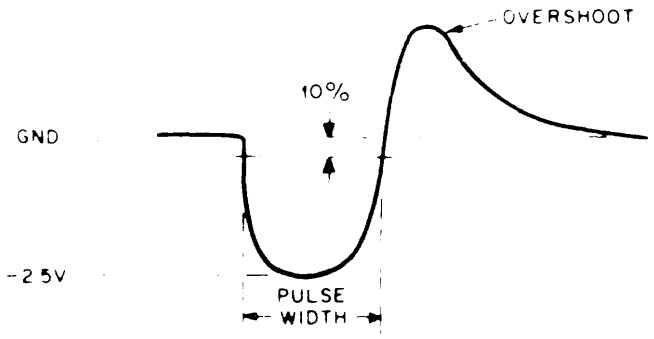


Figure 6-3 FLIP CHIP B-Series Standard Pulse

6.2 DEC ENGINEERING DRAWING CONVENTIONS

6.2.1 Coordinate System

DEC engineering drawings are subdivided into zones bounded by horizontal sections A through D and vertical sections 1 through 8. Circuit locations are referenced by a numeral(s)- letter-numeral combination (e.g., 3A3). The first numeral is the last number of the drawing number; the letter-numeral combination (e.g., A3) designates the module location in drawing coordinates.

6.2.2 Module Identification and Symbology

Circuits enclosed by dotted lines on the engineering drawings are FLIP CHIP modules and are labeled on the drawings by two 4-character designations. The upper designation (i.e., a letter followed by three numerals) specifies the module type as listed in DEC catalog C-105, "The Digital Logic Handbook." The lower 4-character designation indicates the location of the module. In this designation,

the first character (sometimes omitted) is a numeral specifying the number of the bay that contains the equipment; the second character is a letter indicating the mounting panel; the third and fourth characters are numerals indicating the module location from left to right within a specified panel. Individual module pins are labeled from top to bottom by the letters A through Z (excluding letters G, I, O, and Q). Pin letters on logic diagrams are written near their respective terminals. On flow charts and timing diagrams, pin designations are formed by adding the pin letter to the module location code; e.g., 2M10H indicates pin H of module 10 in panel M of bay 2.

Figure 6-4 is an example illustrating module identification, DEC symbols and nomenclature on engineering drawings. The module shown is a Type R202 Flip-flop and is located in the twelfth slot from the left (when viewed from the wiring side) of row B (the second row of modules from the top). The symbol marked FF is a flip-flop with two complementary outputs, terminals S and T. The output of terminal S is connected to terminals D18F and B6M, while the output at terminal T is connected to terminal A3C.

In the 0 state, terminal S is at −3v and terminal T is at ground (as shown by the diamonds inside the symbol at the left). The −3v output is the enabling condition for D18F (SAFE signal) and is also the disabling level for B6M. The ground level at terminal T is a disabling condition for A3C. In the 1 state, terminal T is at −3v and terminal S is at ground (as shown by the diamonds at the right). The ground level is enabling condition for B6M ($\overline{\text{SAFE}}$) and is also a disabling level for D18F. The −3v at terminal T is the enabling for A3C (GO).

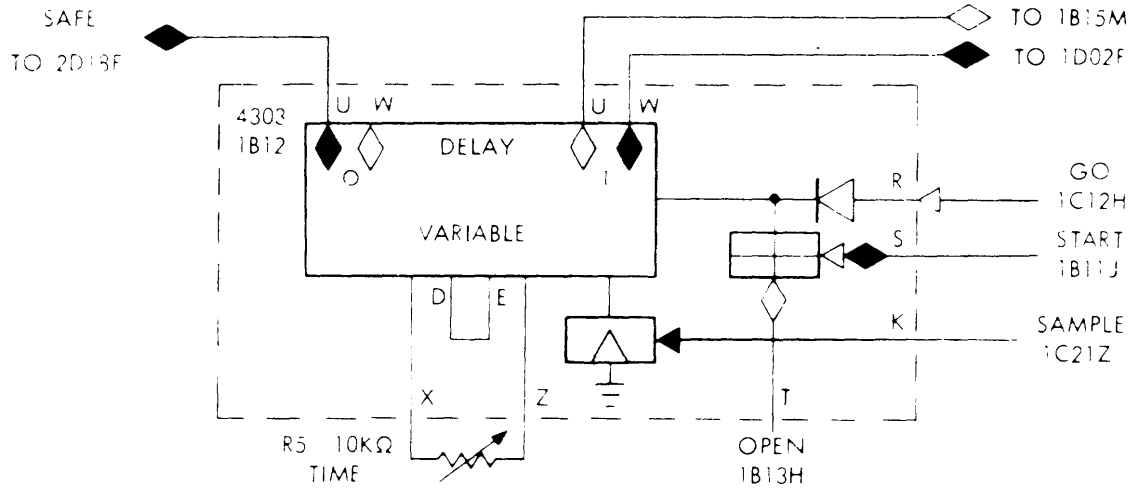


Figure 6-4 Typical DEC Logic Symbology

Tl flip-flop is triggered to the 1 state through its DCD (diode-capacitor-diode) set gate. The two ground-level enabling inputs to this gate are the START pulse (applied to pulse-input terminal B12U) and the OPEN signal (applied to level-input terminal B12V). The OPEN signal must be present for at least 400 nsec before the START pulse can enable the DCD gate to set the flip-flop. The START pulse comes from terminal B11J, and the OPEN signal from terminal C21C.

The flip-flop is cleared to the 0 state either through its DCD reset gate or through its direct-clear input terminal. Since the level-input terminal (B12P) of the DCD gate is permanently grounded, the flip-flop is cleared whenever the RST pulse arrives at terminal B12N. This pulse is received from terminal F4A. The flip-flop is also cleared by the CLR pulse which arrives at direct-clear B12R from D6B.

6.3 REPLACEMENT SCHEMATICS

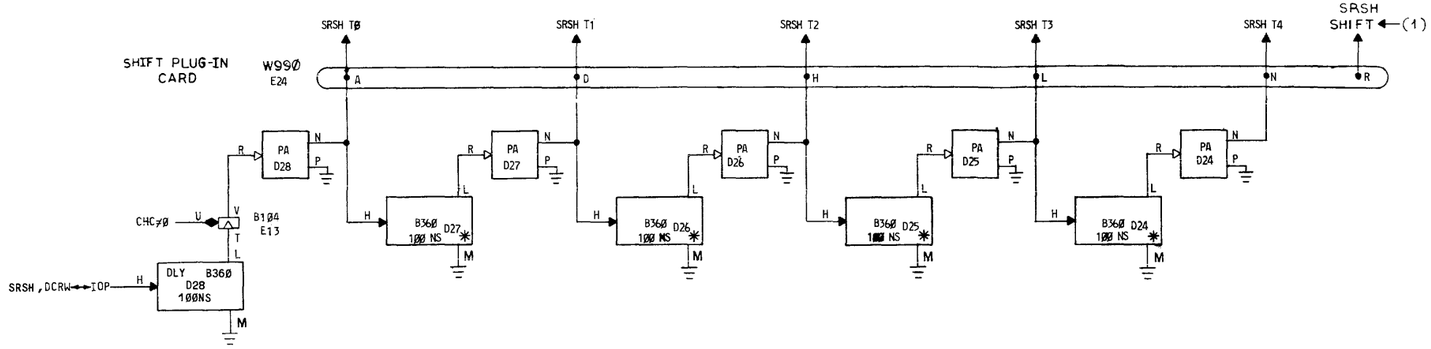
Drawing numbers for replacement schematics are formed by a letter-numeral combination as follows: RS-(letter)-(3 or 4 numerals). The letter indicates the drawing size, and the numerals indicate the circuit type. Another numeral to the right of these numerals indicates the number of the revision, while a letter to the left indicates the circuit board revision. This revision letter is etched on the circuit board and embossed after the type numbers on the aluminum frame of the module.

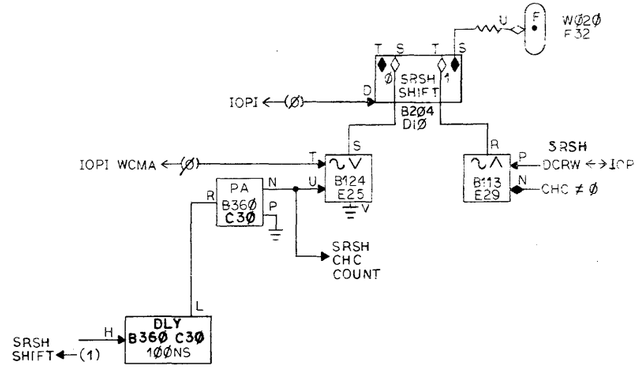
6.3.1 Semiconductor Substitution

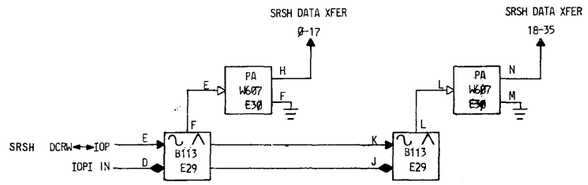
Standard EIA components specified in table 6-1 can replace the majority of DEC semiconductors used in modules of the drum processor. Exact replacement is recommended for semiconductors not listed.

TABLE 6-1 SEMICONDUCTOR SUBSTITUTION

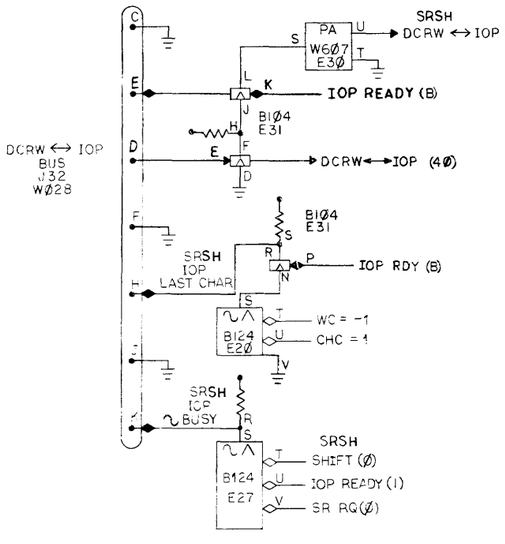
DEC	EIA	DEC	EIA
D662	1N645	DEC2894-3B	DEC2894-3A
D664	1N3606	D3009	2N3009
D668	Two 1N3606 in series	DEC3639,-0	2N3639
DEC2219	2N2219	DEC3639B	2N3639
DEC2894,-1,-2,-3	DEC2894		

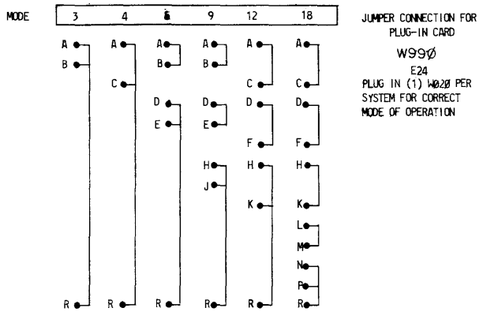






MODE	TØ	T1	T2	T3	T4
3	3				
4	4				
6	3	3			
9	3	3	3		
12	4	4	4		
18	4	4	4	3	3

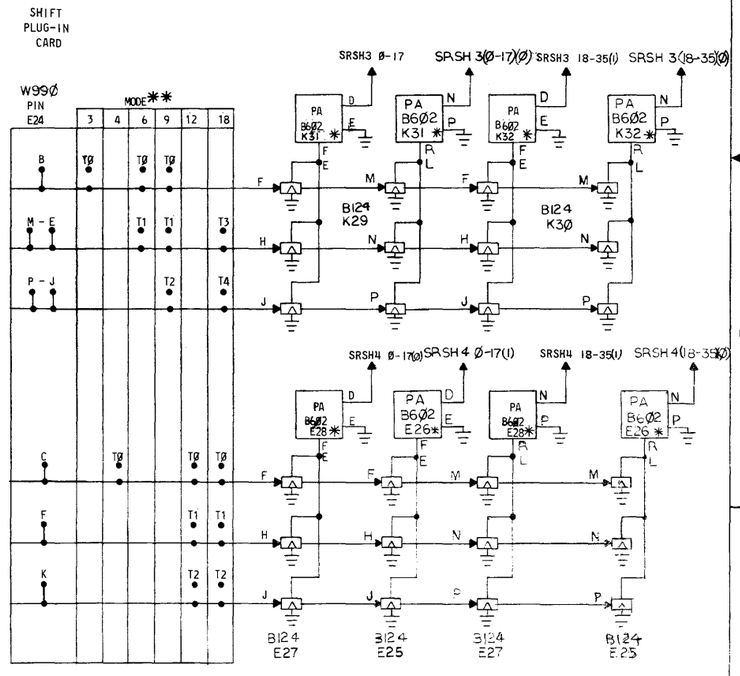




JUMPER CONNECTION FOR PLUG-IN CARD W99Ø E24 PLUG IN (1) WØ2Ø PER SYSTEM FOR CORRECT MODE OF OPERATION

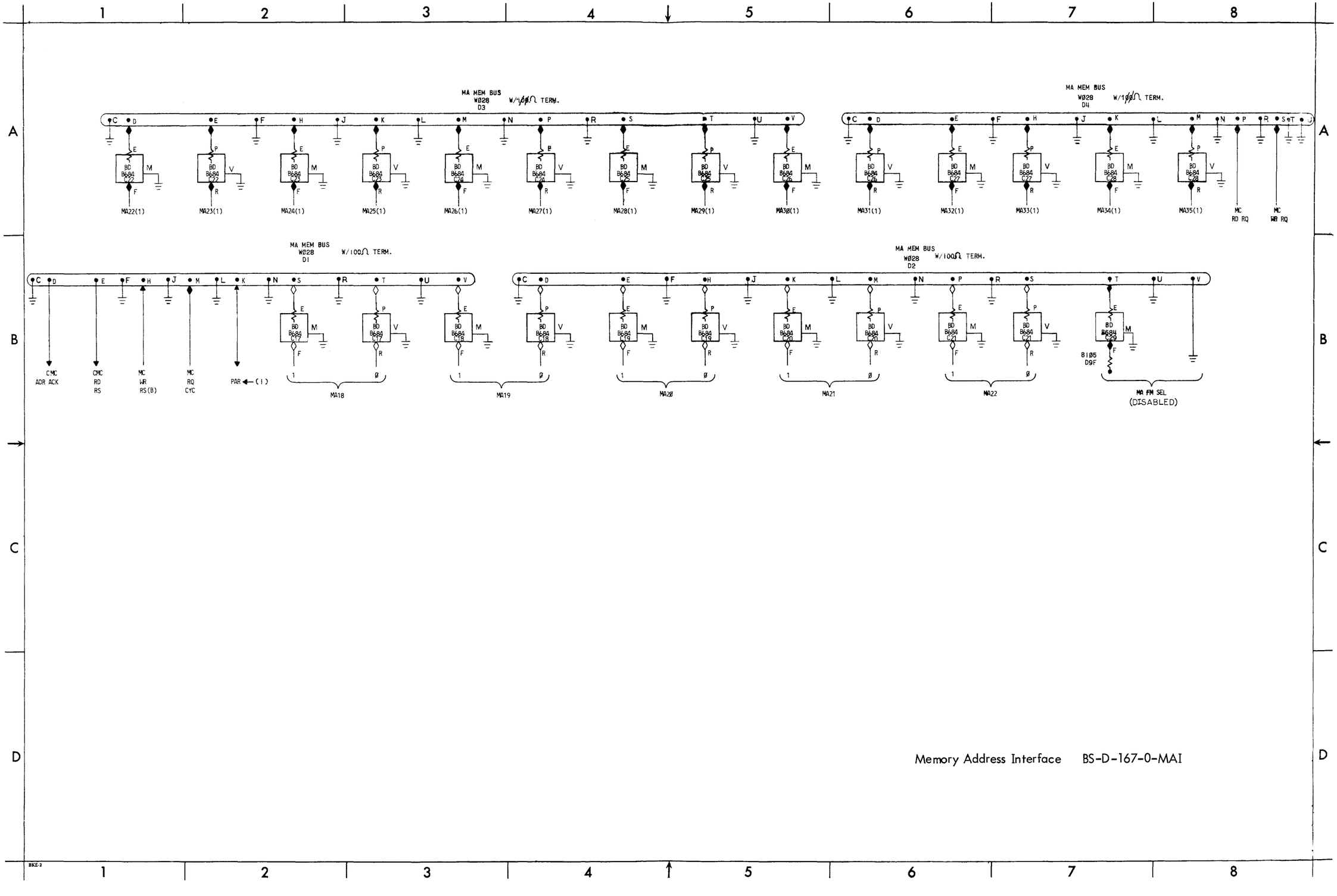
*REMOVE PKG IF NOT USED.

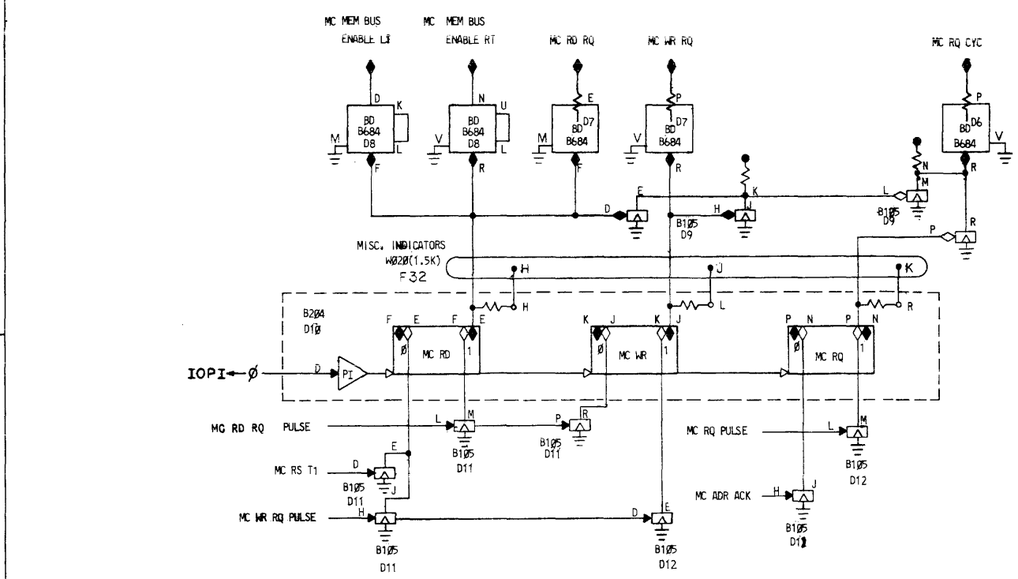
**MAKE CONNECTIONS AS SHOWN IN VERTICAL COLUMN UNDER MODE OF OPERATION.

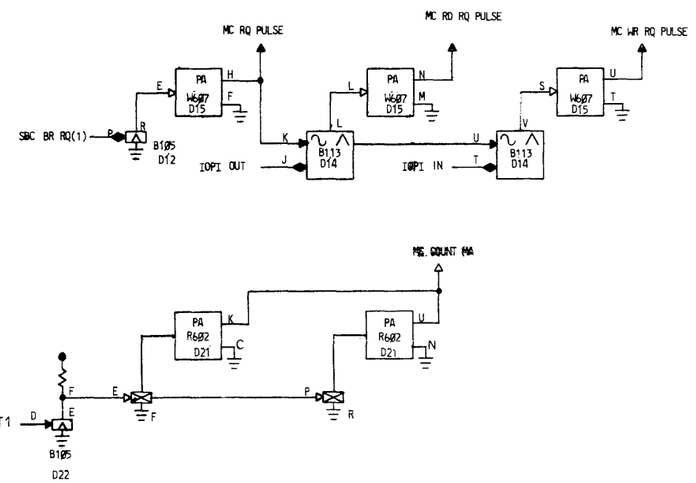


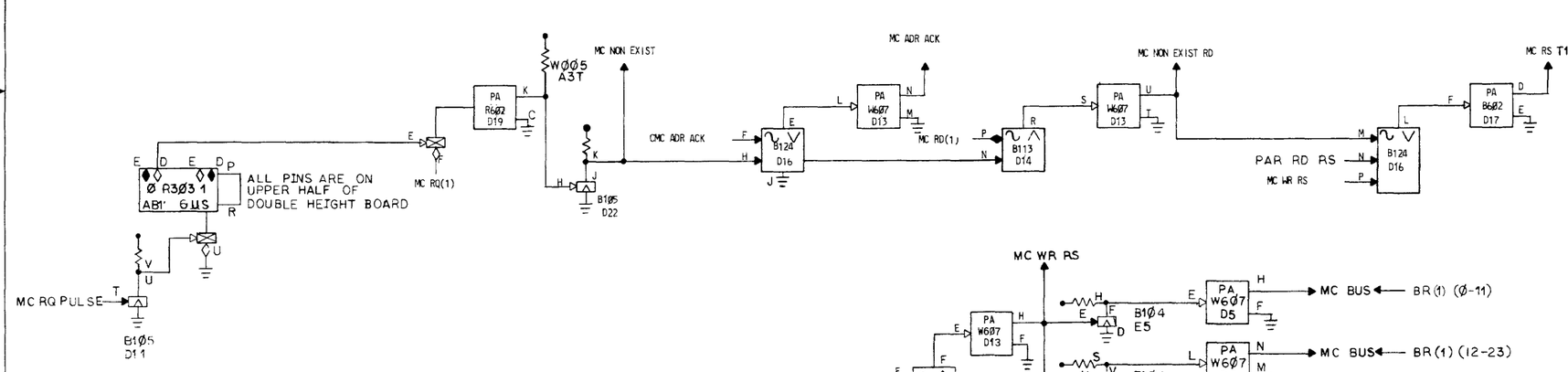
SR Shift BS-D-167-0-SRSH

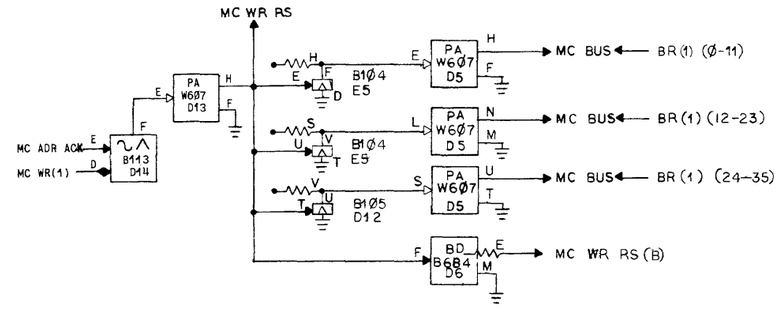
BKE-2


MA MEM BUS
W028
D3
W/100Ω TERM.
MA22(1)
MA23(1)
MA24(1)
MA25(1)
MA26(1)
MA27(1)
MA28(1)
MA29(1)
MA30(1)
MA MEM BUS
W028
D4
W/100Ω TERM.
MA31(1)
MA32(1)
MA33(1)
MA34(1)
MA35(1)
MC
RD RQ
MC
WR RQ
MA MEM BUS
W028
D1
W/100Ω TERM.
CMC
ADR ACK
CMC
RD
RS
MC
WR
RS(B)
MC
RQ
CYC
PAR ◄ (1)
MA18
MA19
MA MEM BUS
W028
D2
W/100Ω TERM.
MA20
MA21
MA22
B105
D9F
MA FM SEL
(DISABLED)
Memory Address Interface BS-D-167-0-MAI
BKE-2



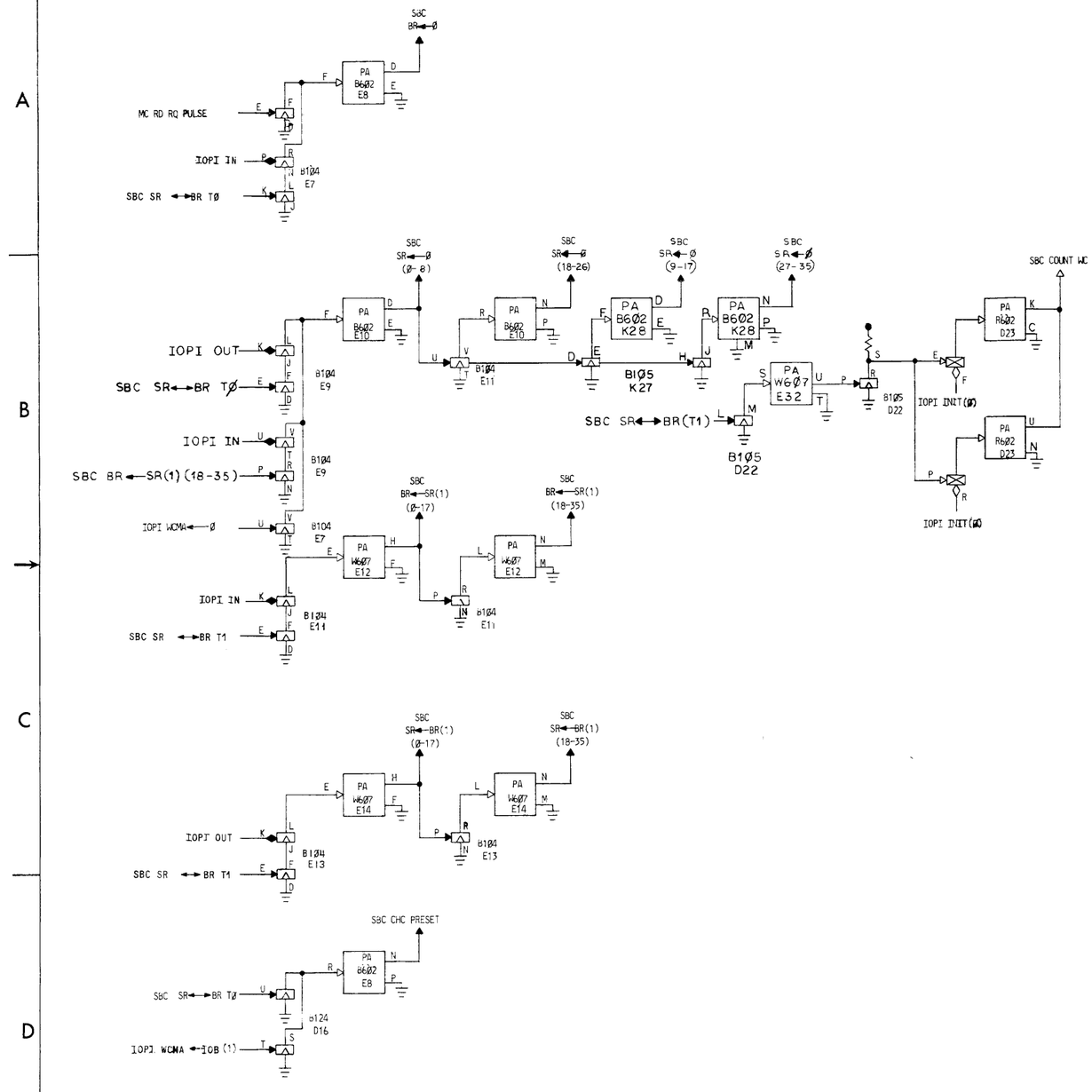


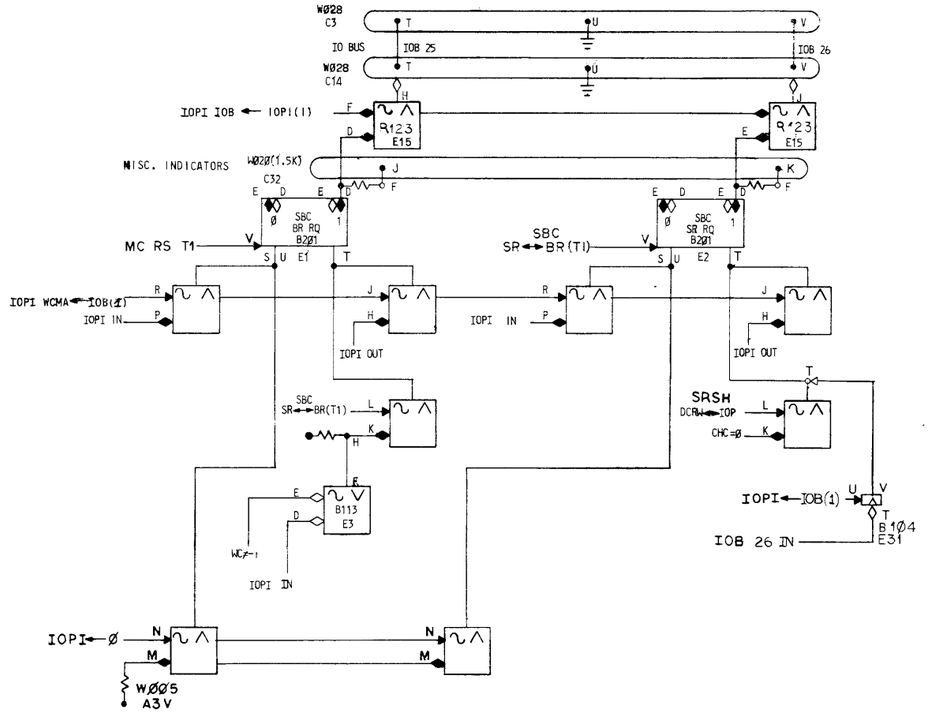


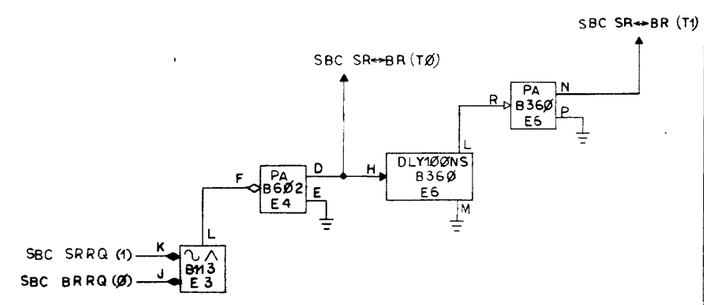


Memory Control BS-D-167-0-MC

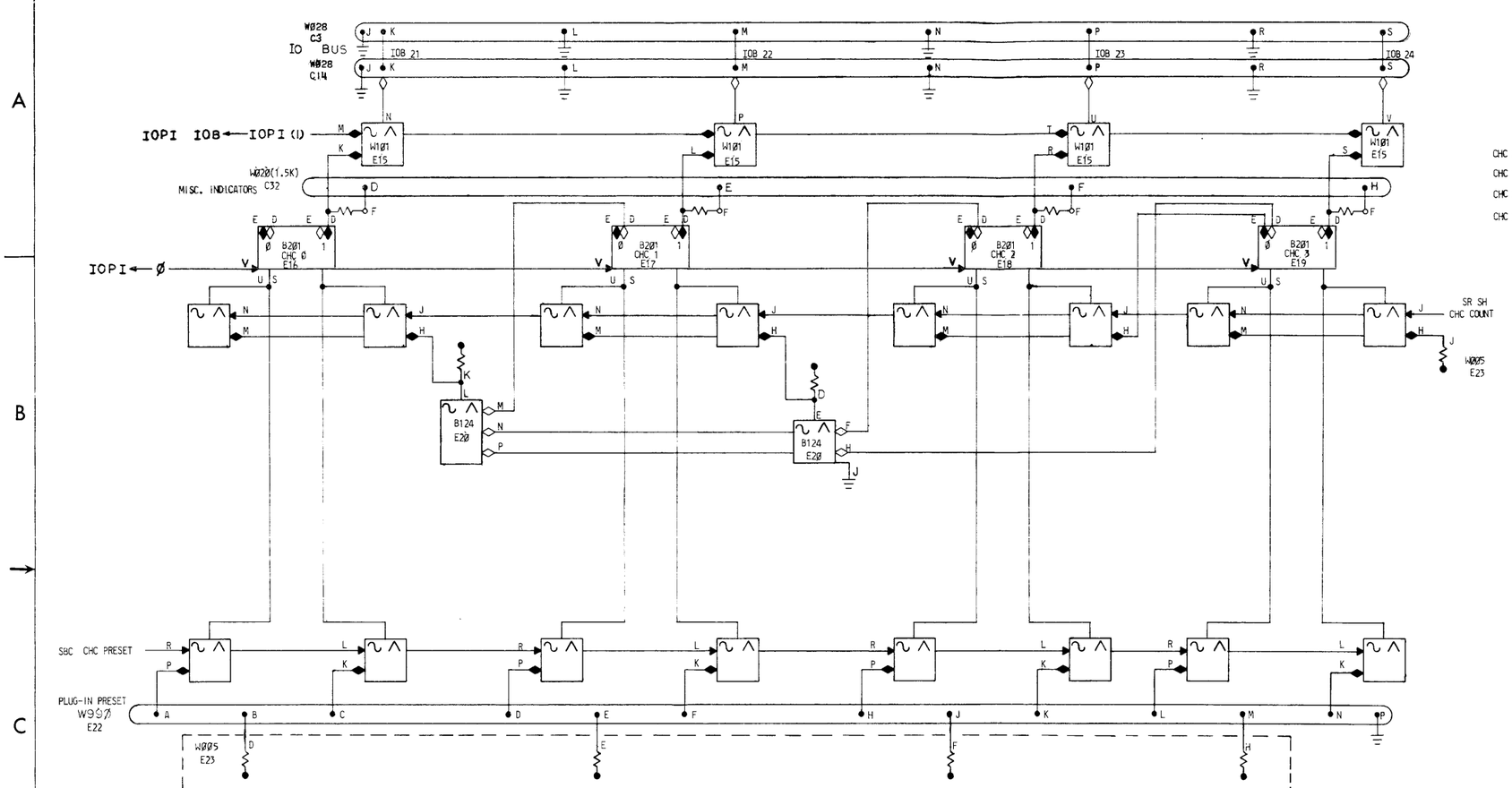
BKE-2

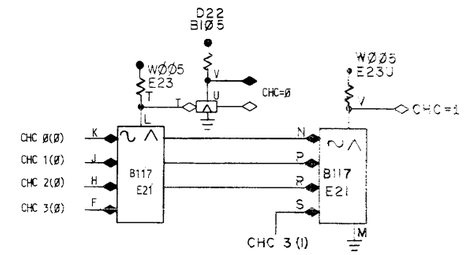






SR BR Control BS-D-167-0-SBC



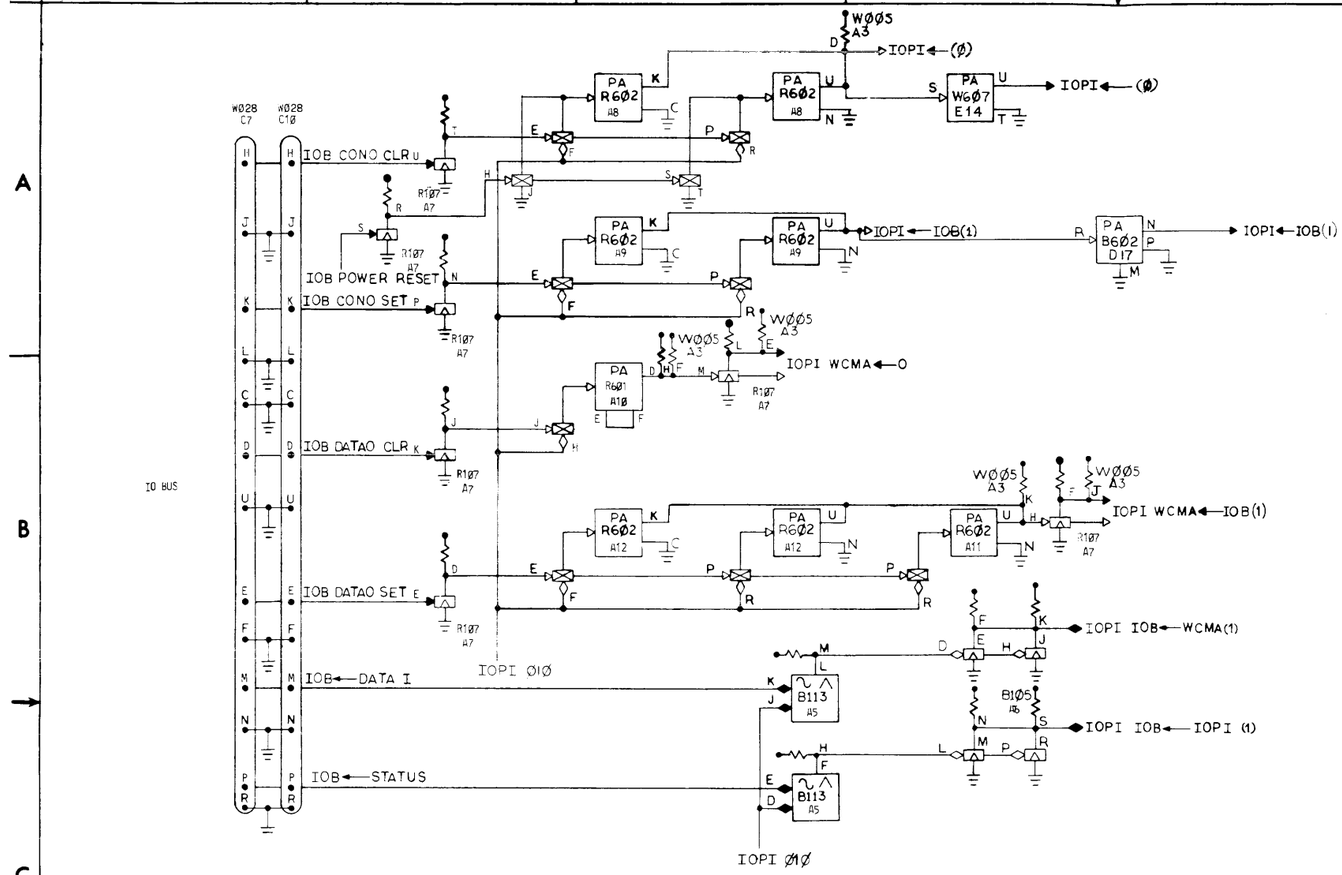


MODE	#CH/CARD	OCTAL	SH-1 (PRESET)	CARD CONNECTIONS
3	12	14	1Ø11	(B-C) (D-E) (J-K) (M-N) (A-F-H-L-P)
4	9	11	1ØØØ	(B-C) (D-E) (H-J) (M-L) (A-F-K-N-P)
6	6	6	Ø1Ø1	(A-B) (E-F) (H-J) (M-N) (C-D-K-L-P)
9	4	4	ØØ11	(A-B) (D-E) (J-K) (M-N) (C-F-H-L-P)
12	3	3	ØØ1Ø	(A-B) (D-E) (J-K) (M-L) (C-F-H-N-P)
18	2	2	ØØØ1	(A-B) (D-E) (H-J) (M-N) (C-F-K-L-P)
36	Ø	Ø	ØØØØ	(A-B) (D-E) (H-J) (M-L) (C-F-K-N-P)

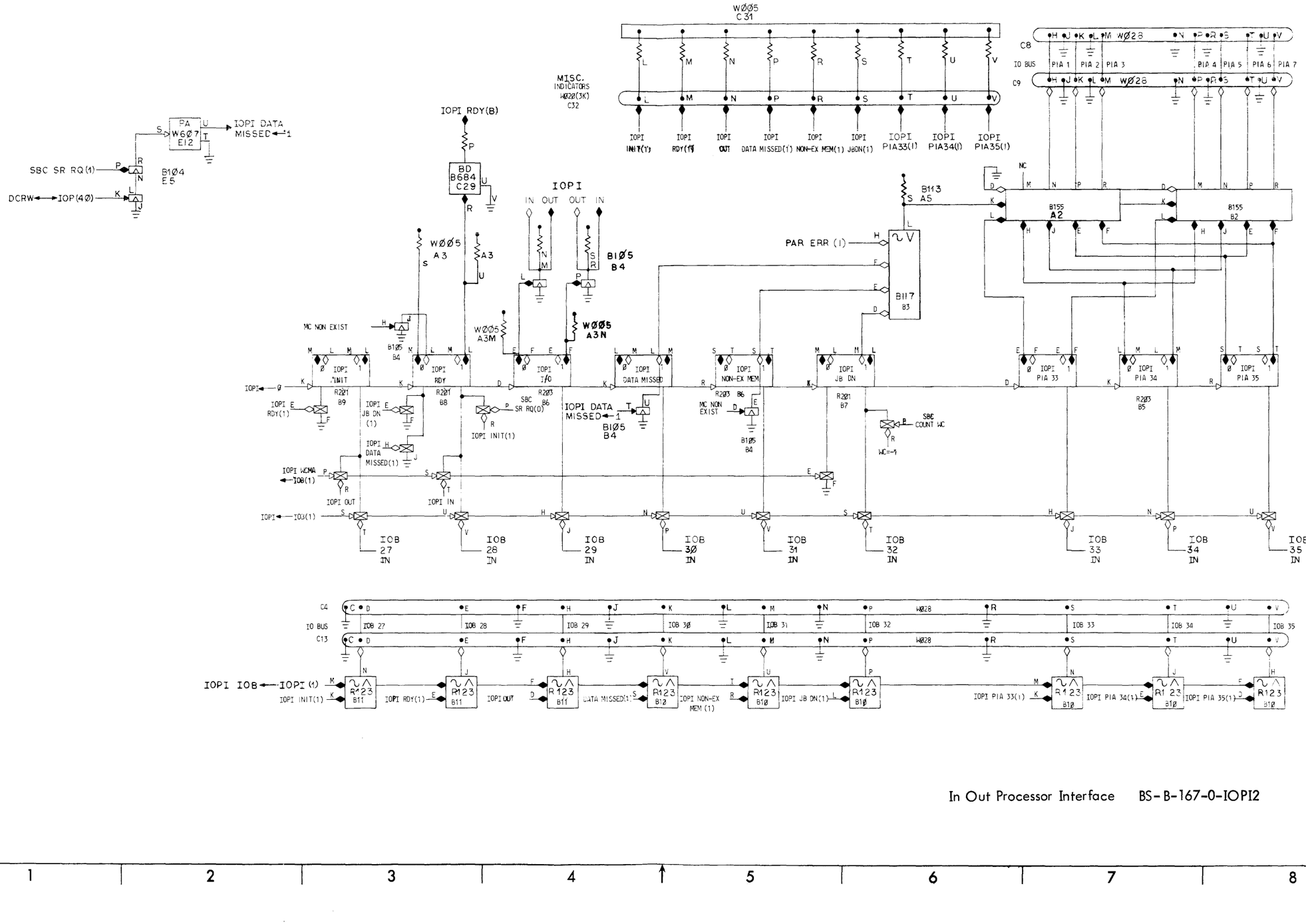
JUMPER CONFIGURATION
FOR
PLUG-IN PRESET
W99Ø
E22

Character Counter BS-D-167-0-CHC

BKE-2

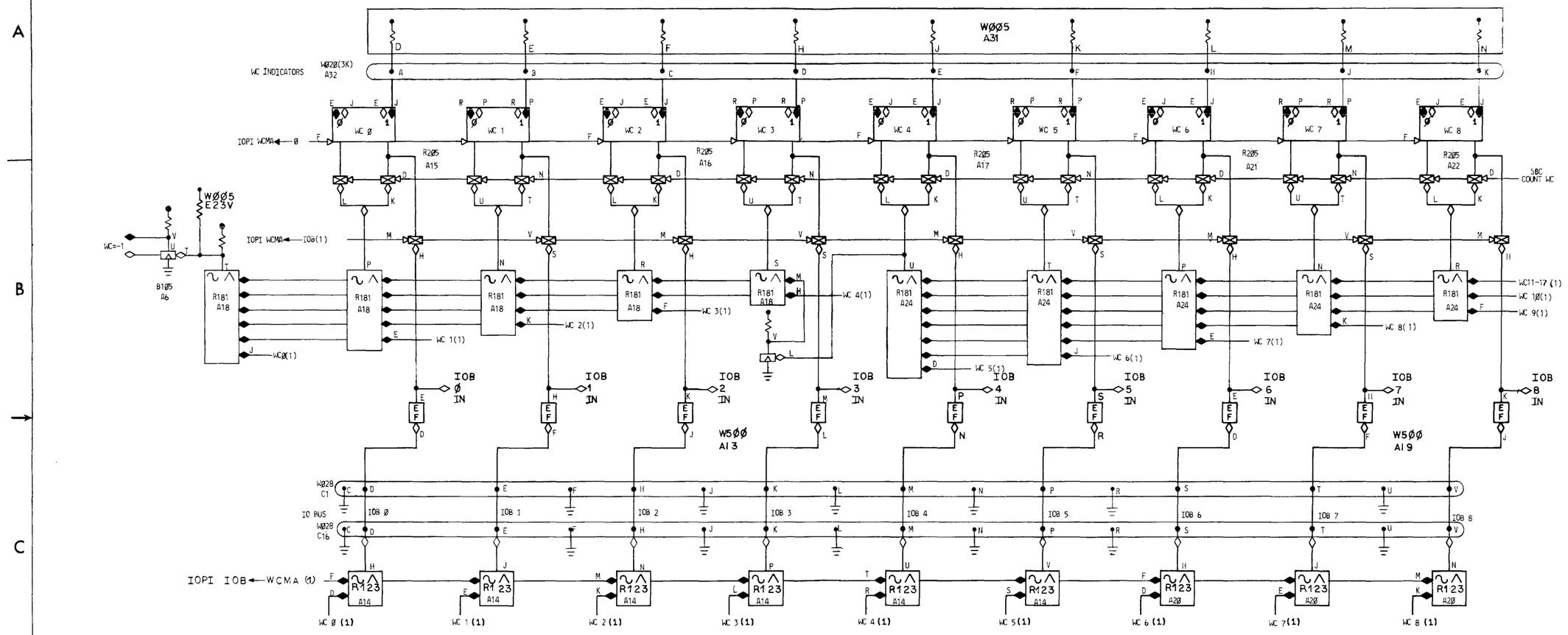


In Out Processor Interface BS-D-167-0-IOPI1

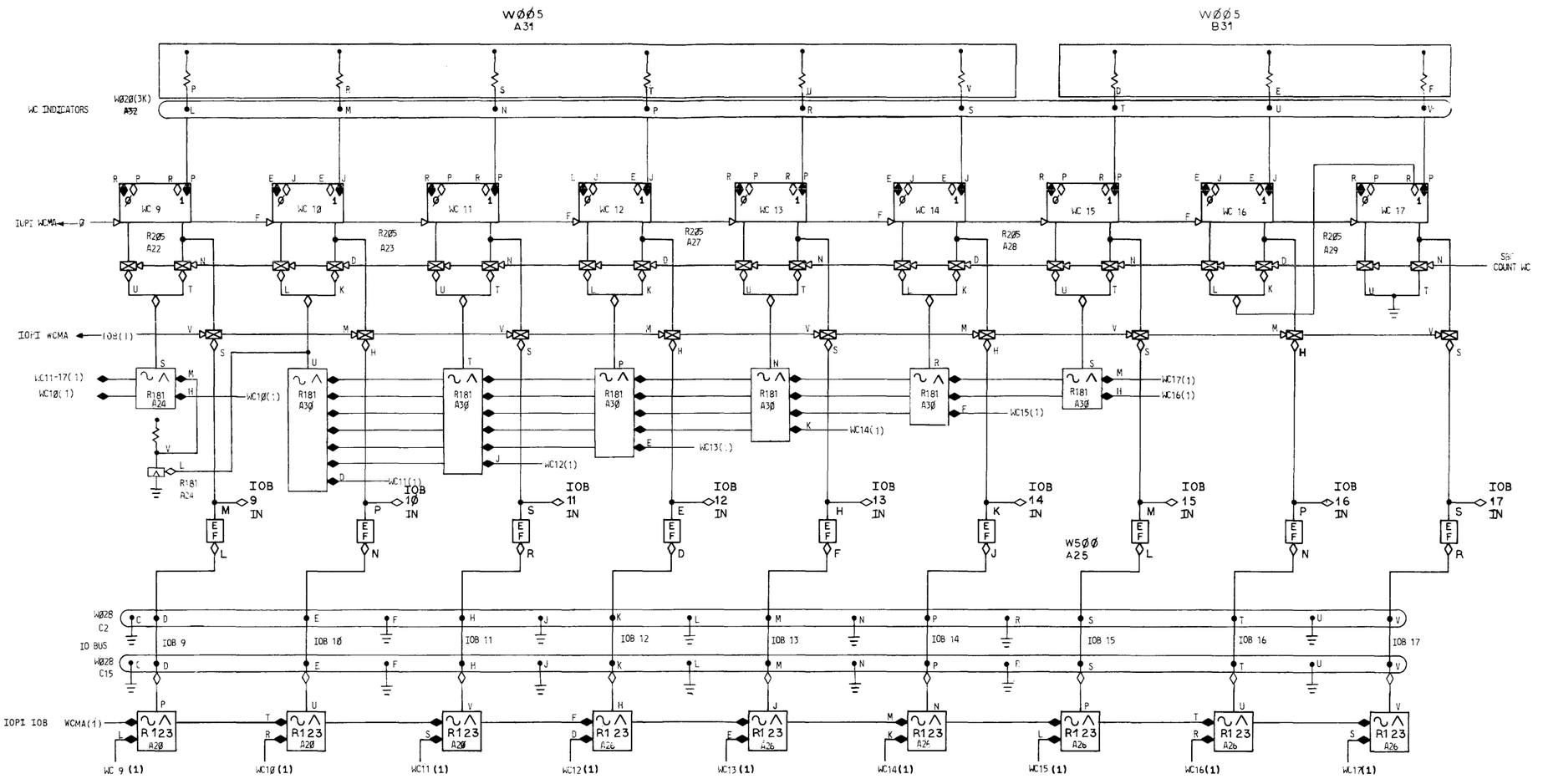


In Out Processor Interface BS-B-167-0-IOPI2

Word Counter 0-8 BS-D-167-0-WC-1

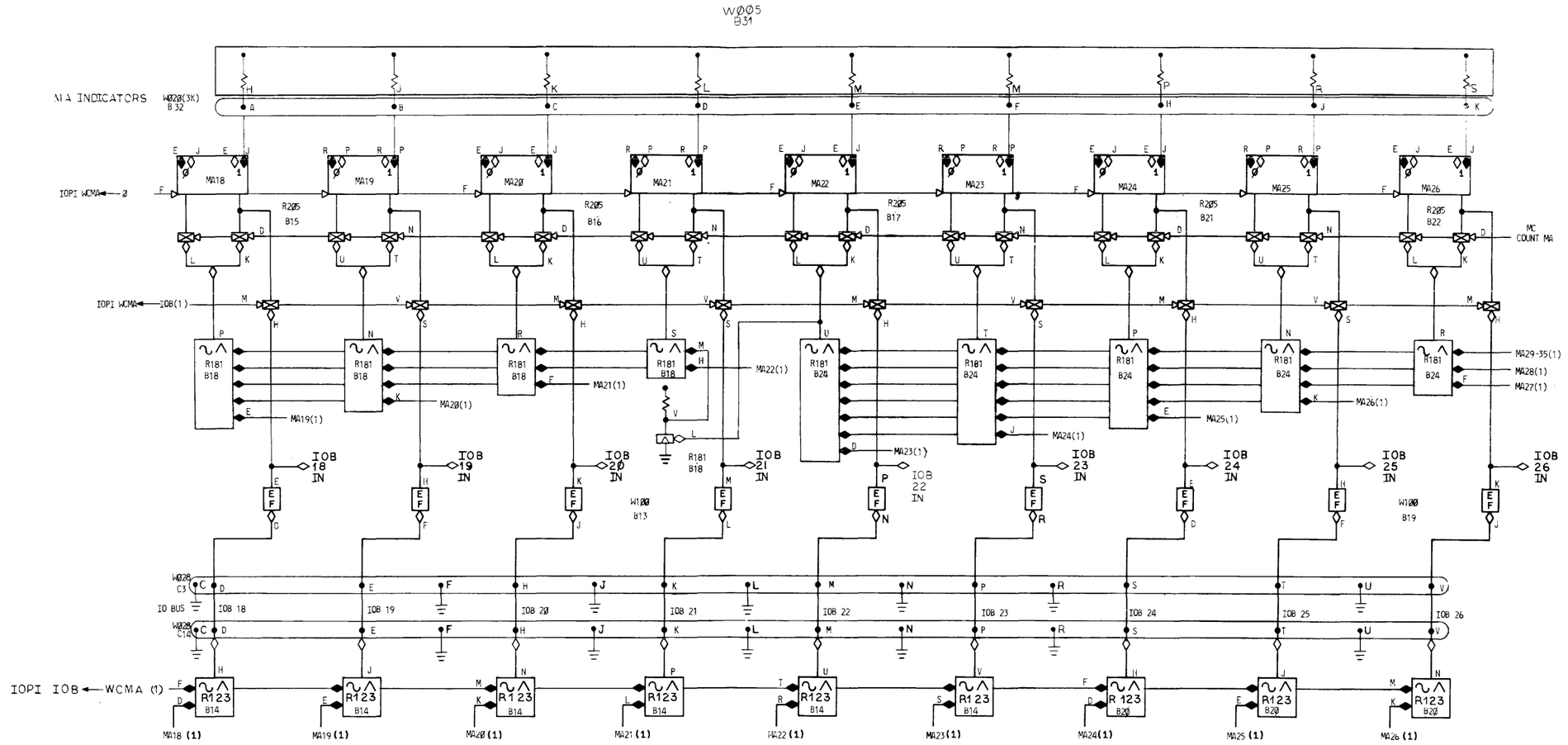


Word Counter 0-8 BS-D-167-0-WC-1

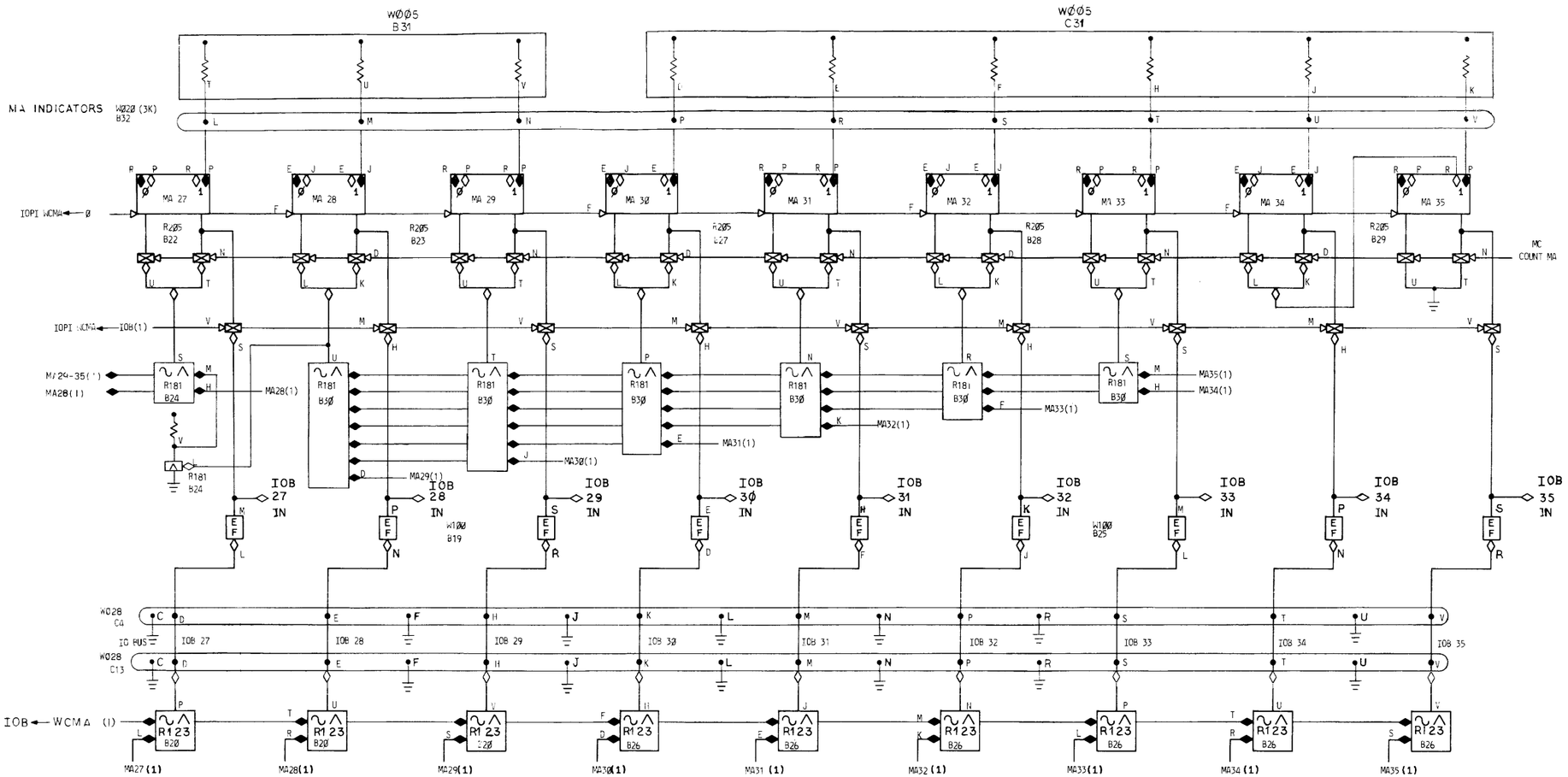


Word Counter 9-17 BS-D-167-0-WC-2

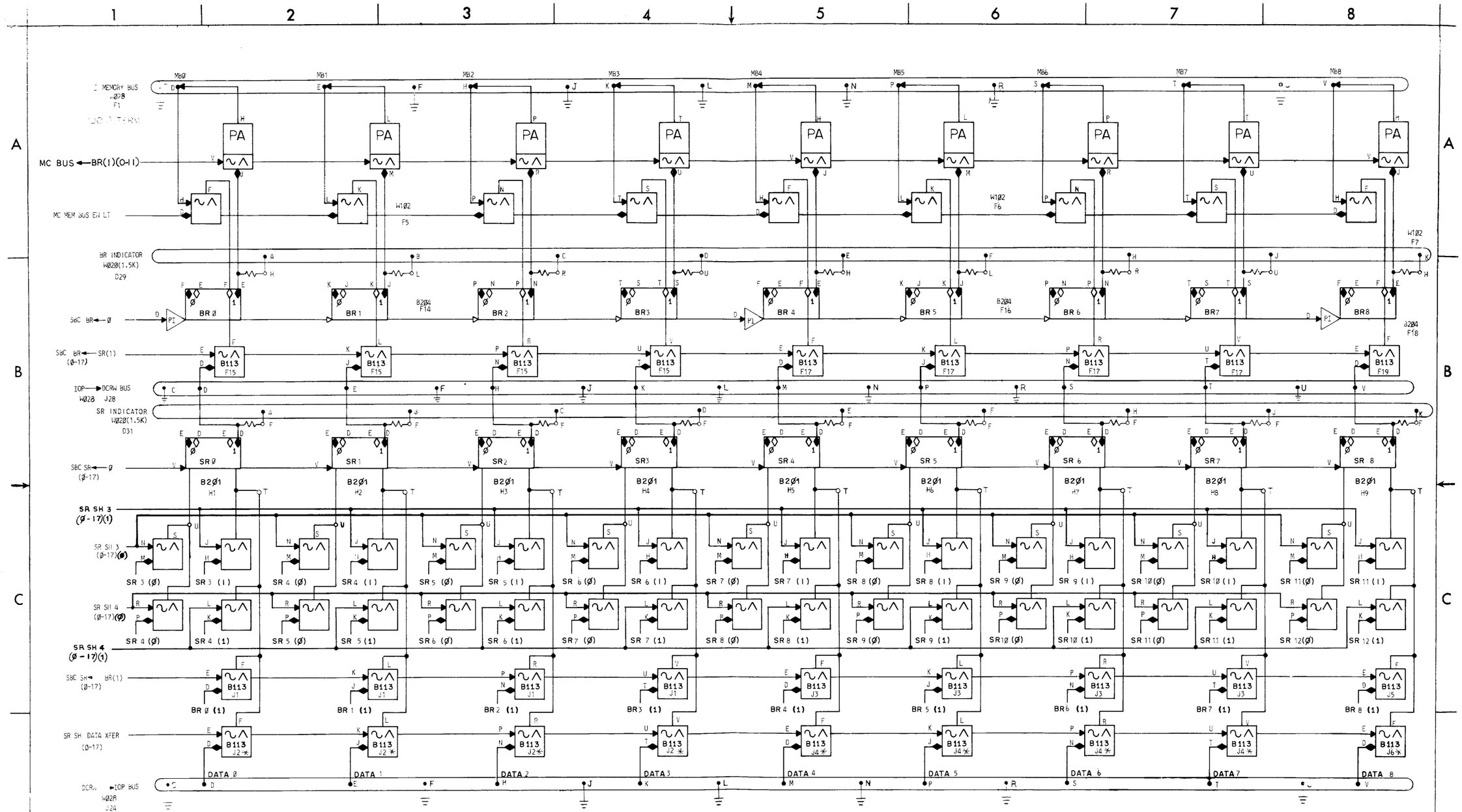
BKE 2



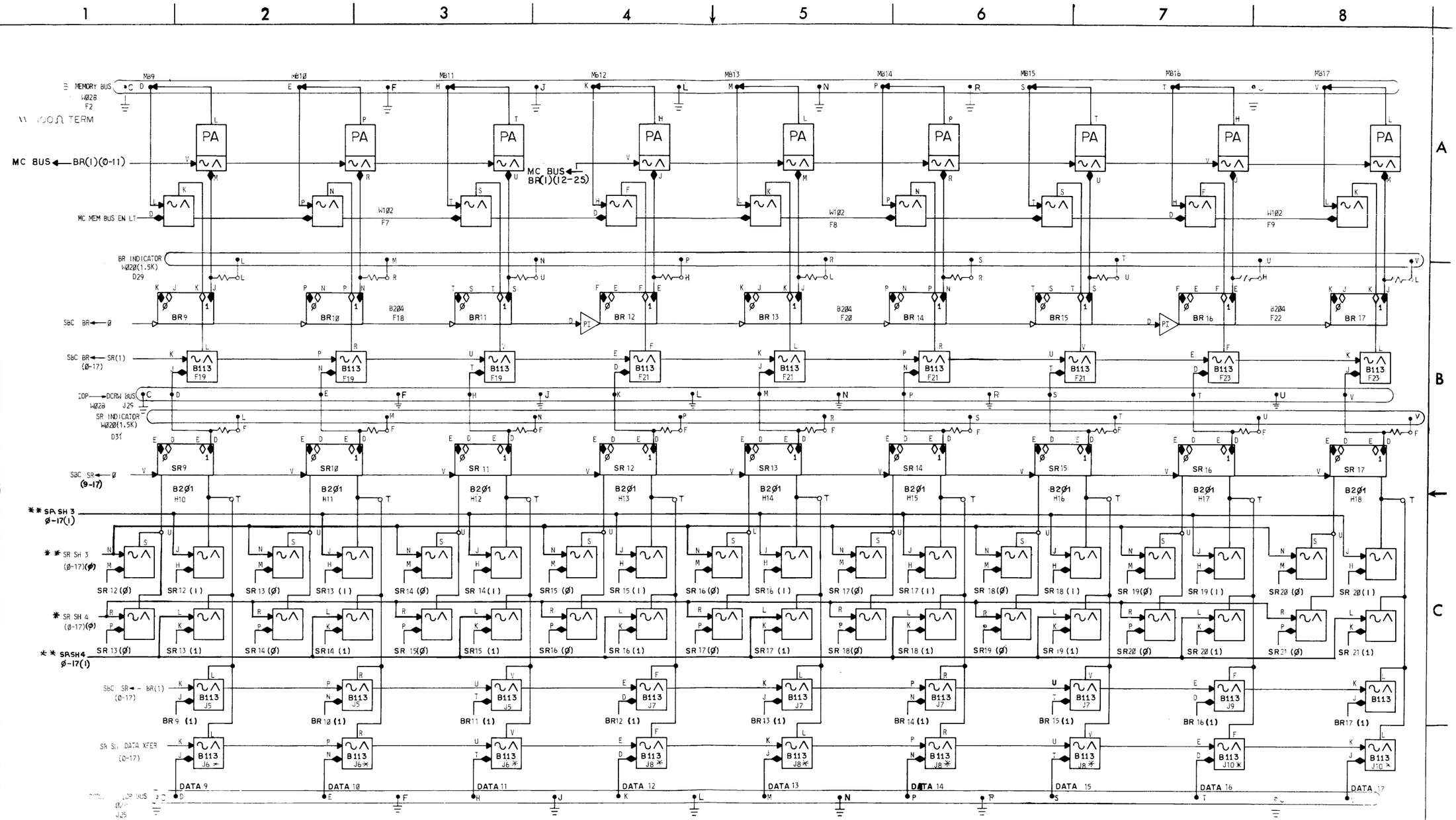
Memory Address 18-26 BS-D-167-0-MA-1



Memory Address 27-35 BS-D-167-0-MA-2

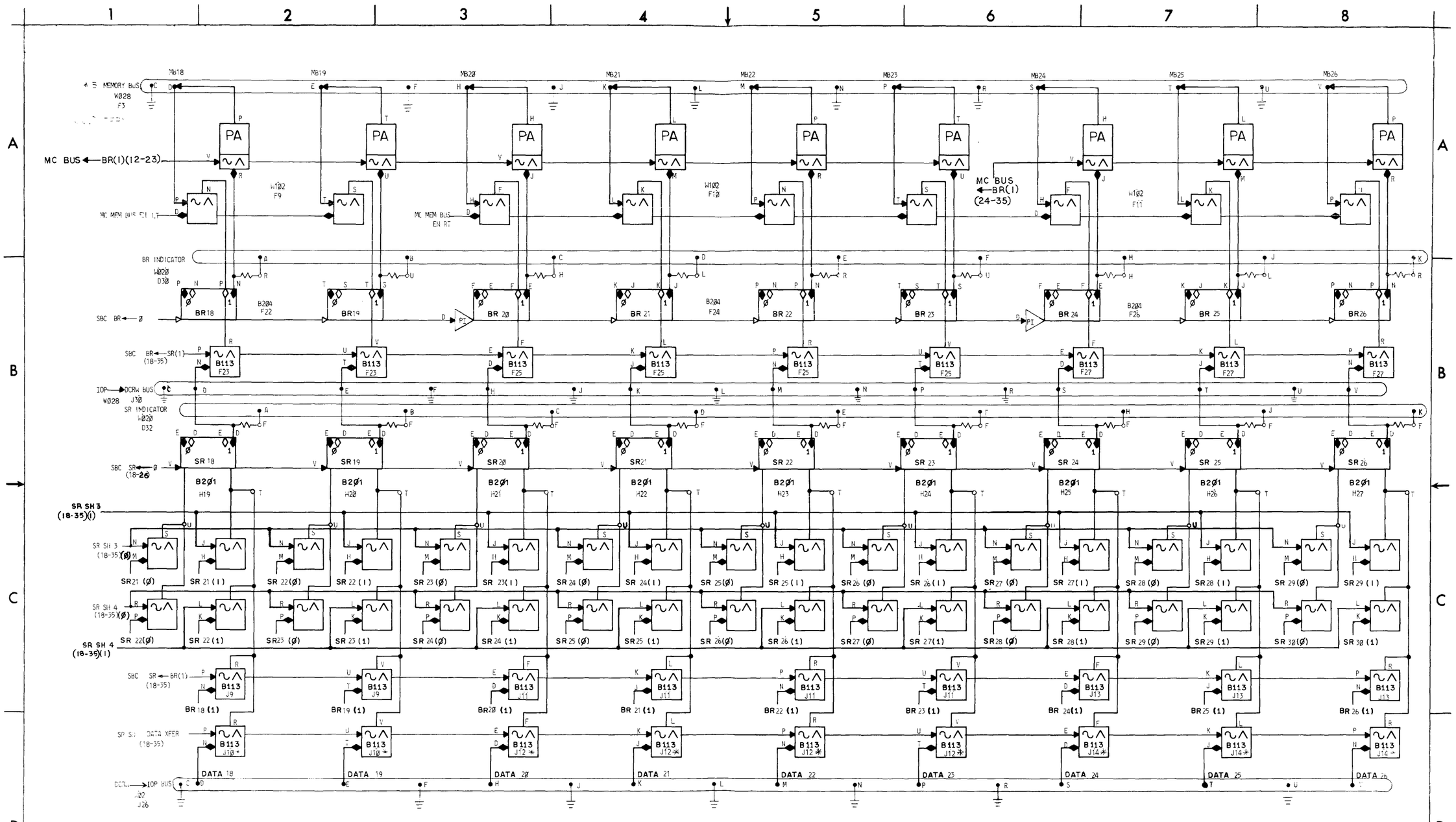


BR SR 0-8 BS-D-167-0-BRSR1



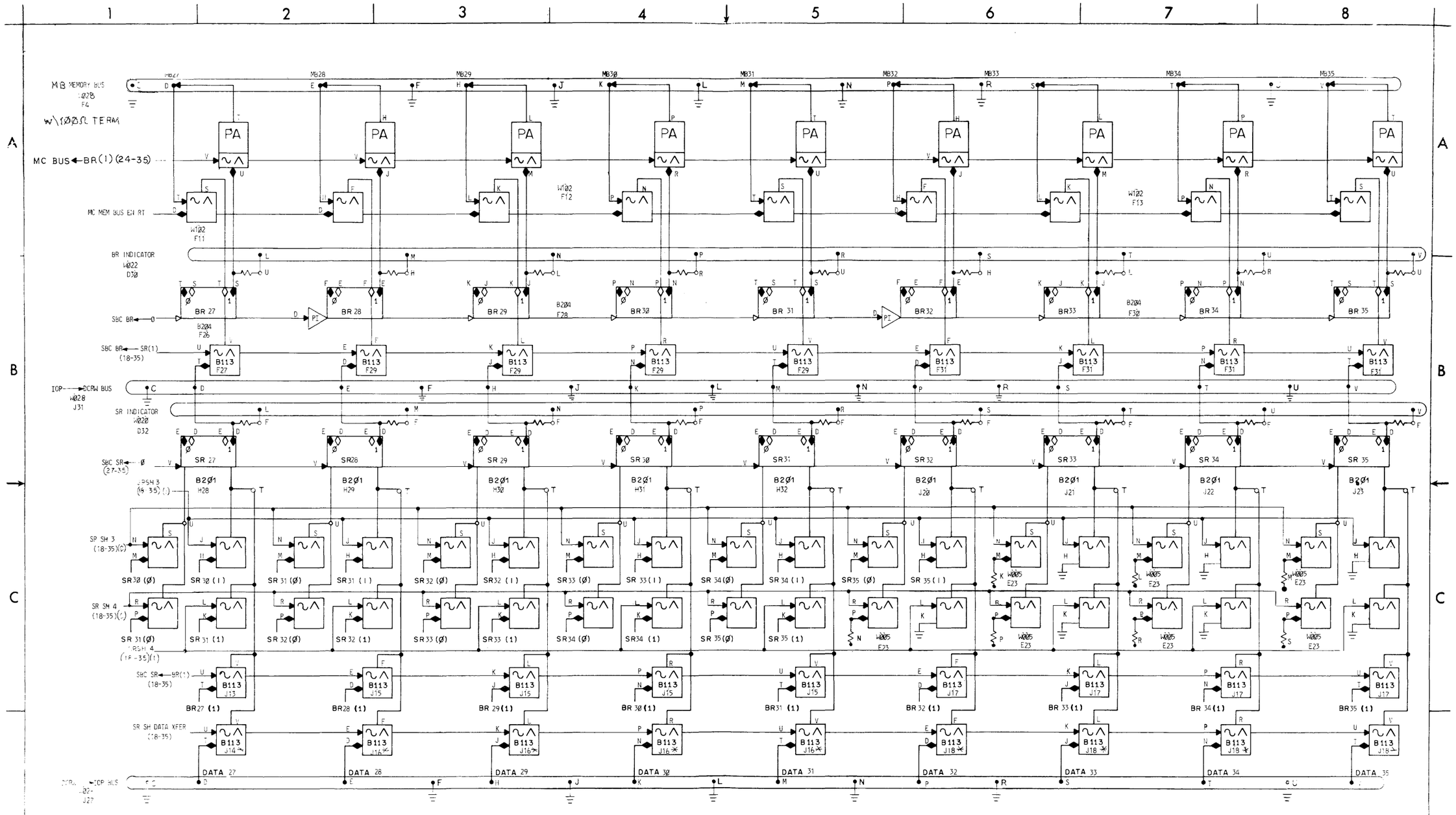
** IHIØ-N,J,R,L, ARE BUSSED SEPARATELY, THERE IS NO PHYSICAL WIRING BETWEEN N,J OR R,L.

BR SR 9-17 BS-D-167-0-BRSR2



* REMOVE PKG IF NOT USED

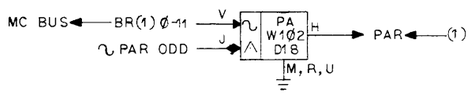
BR SR 18-26 BS-D-167-0-BRSR3

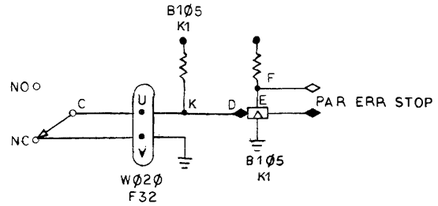


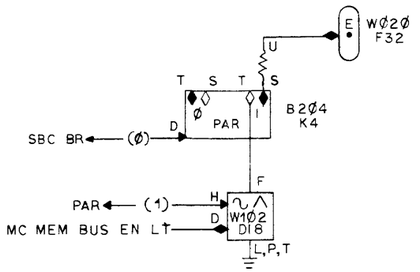
*REMOVE PKG IF NOT USED

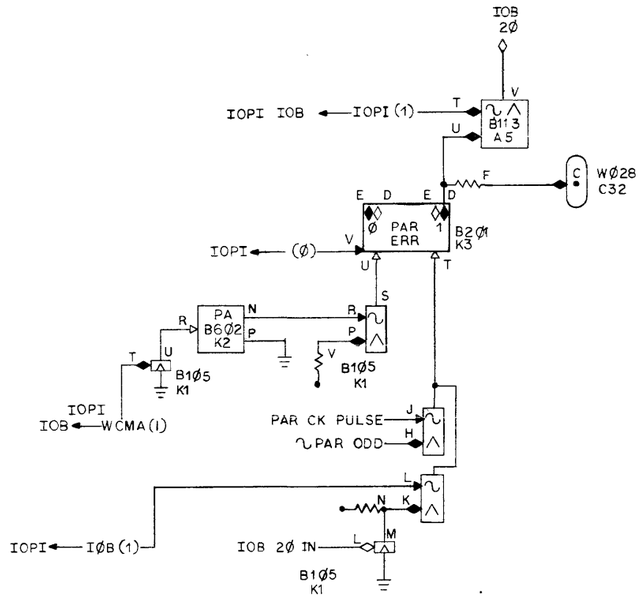
BR SR 27-35 BS-D-167-0-BRSR4

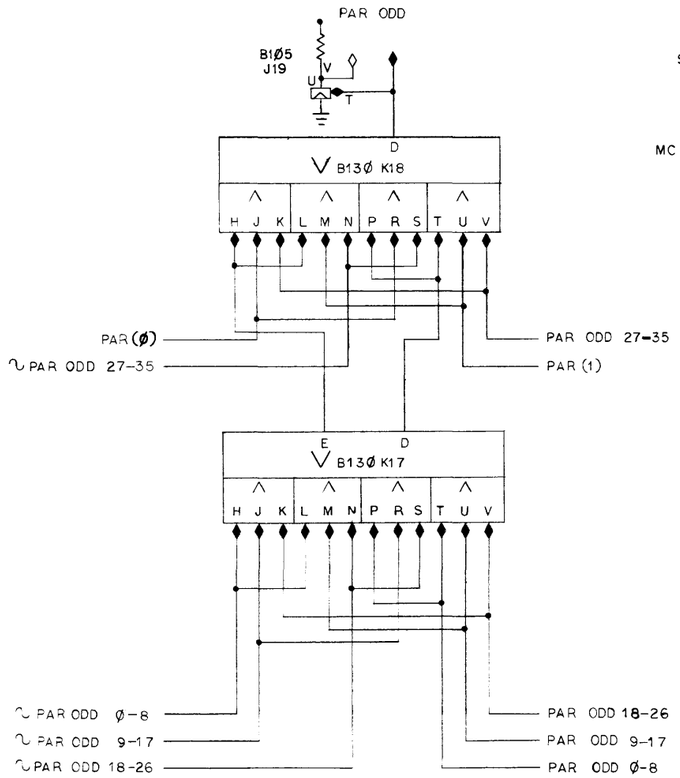
Parity Control and Net (Sheet 1) BS-D-167-0-PAR

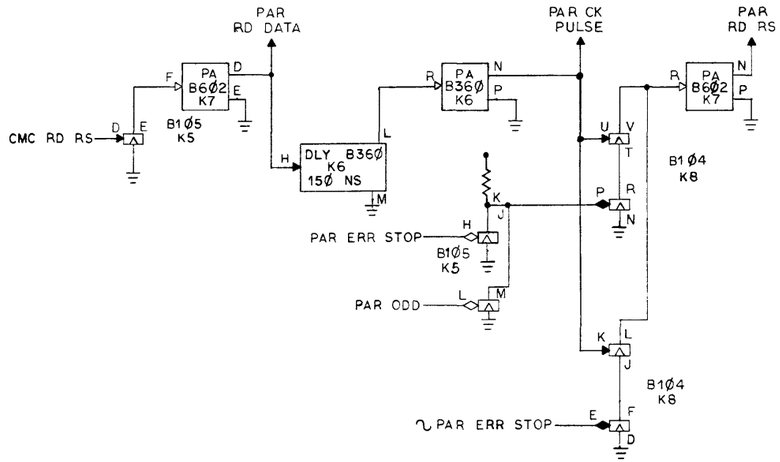






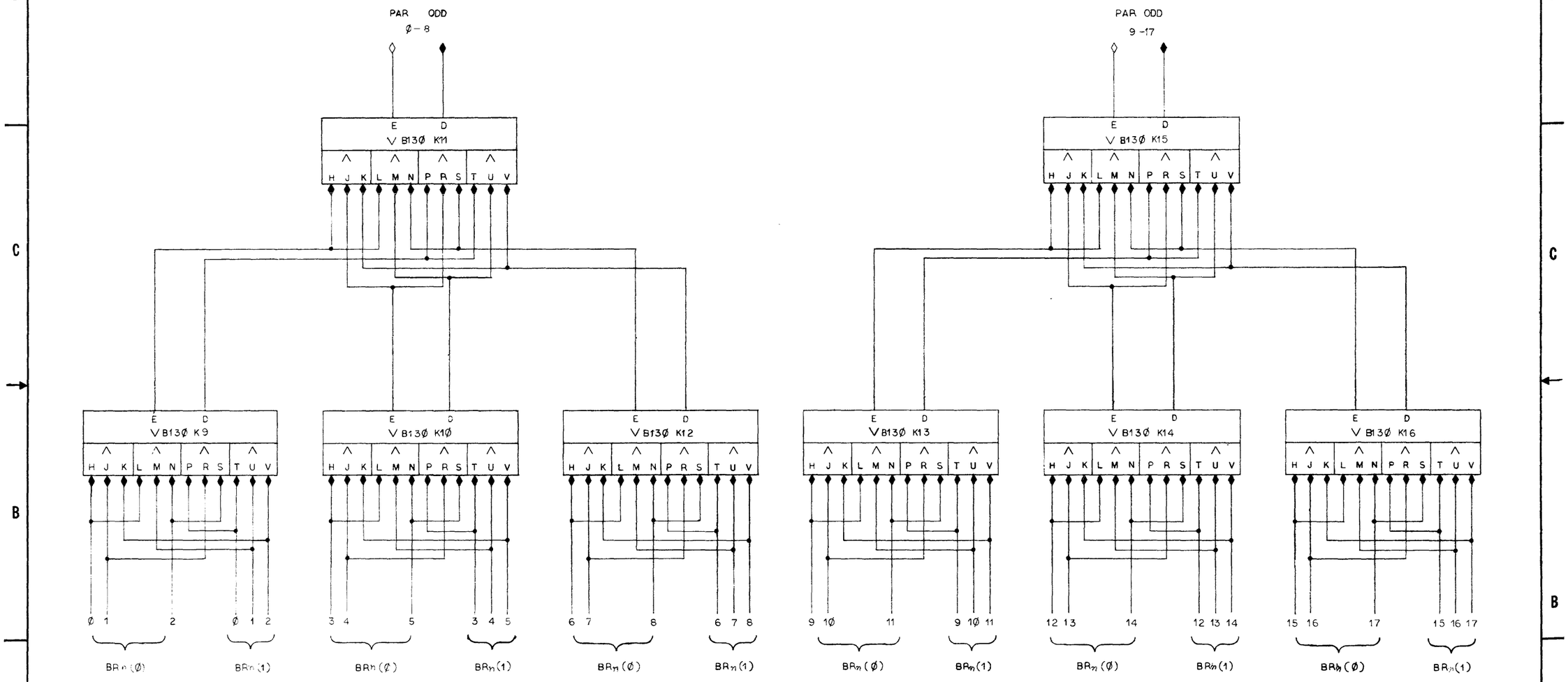






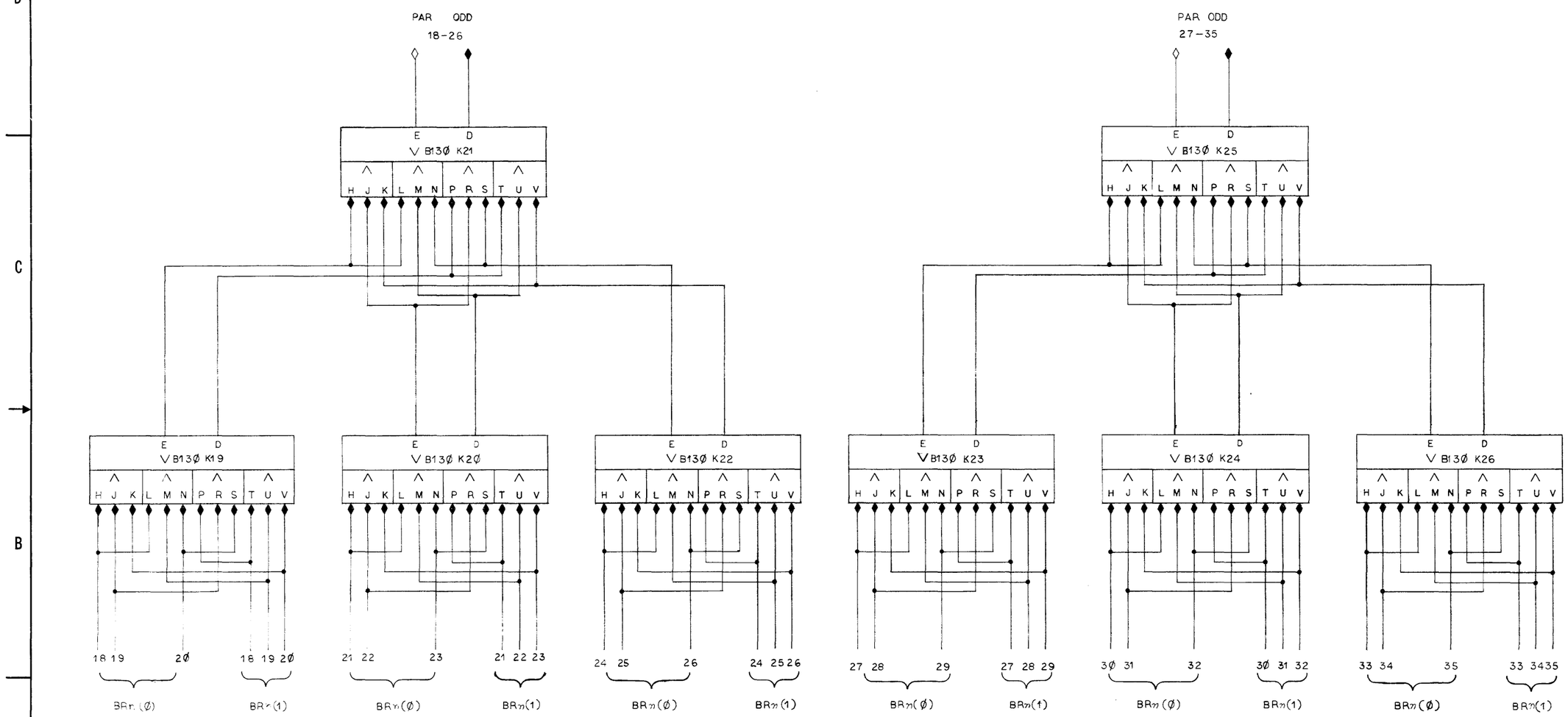
Parity Control and Net (Sheet 1) BS-D-167-0-PAR

Parity Control and Net (Sheet 2) BS-D-167-0-PAR

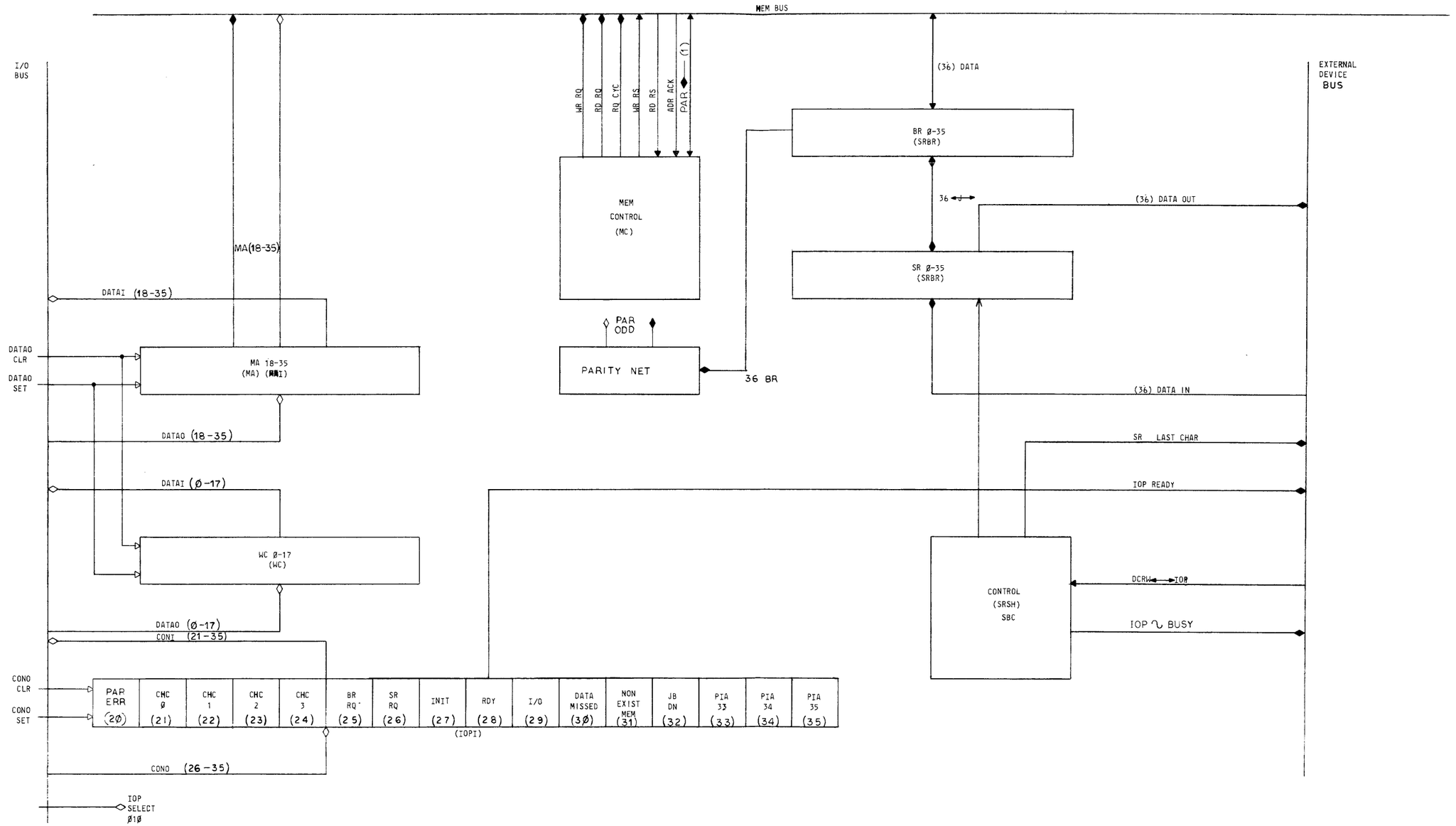


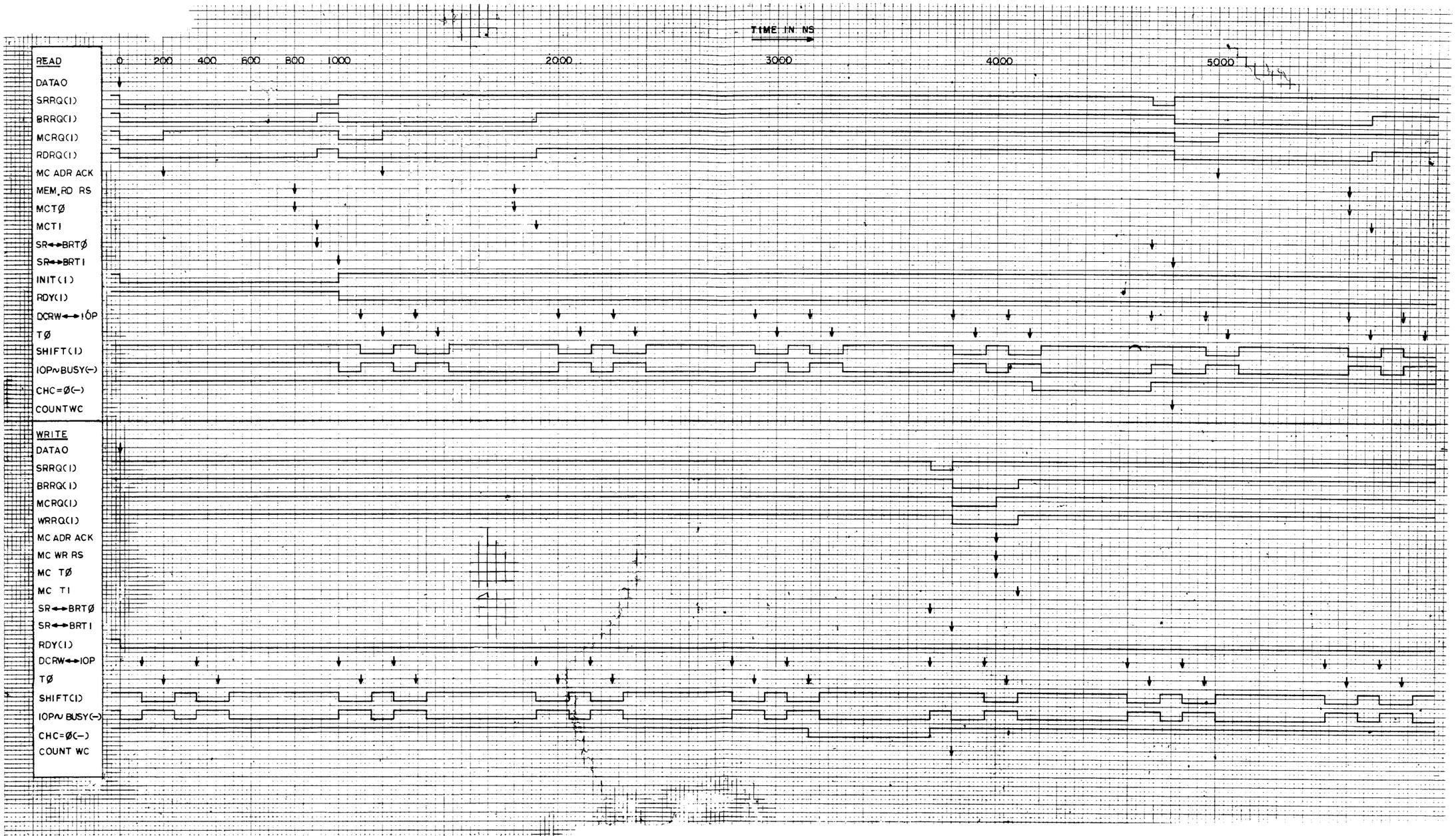
Parity Control and Net (Sheet 2) BS-D-167-0-PAR

Parity Control and Net (Sheet 3) BS-D-167-0-PAR



Parity Control and Net (Sheet 3) BS-D-167-0-PAR


MEM BUS
I/O BUS
WR RQ
RD RQ
RQ C/C
WR RS
RD RS
ADR ACK
PAR (1)
(36) DATA
EXTERNAL DEVICE BUS
MEM CONTROL (MC)
BR Ø-35 (SRBR)
36
(36) DATA OUT
SR Ø-35 (SRBR)
MA(18-35)
DATAI (18-35)
PAR ODD
PARITY NET
36 BR
DATAO CLR
DATAO SET
MA 18-35 (MA) (RRI)
(36) DATA IN
DATAO (18-35)
SR LAST CHAR
DATAI (Ø-17)
IOP READY
WC Ø-17 (WC)
CONTROL (SRSH) SBC
DCRW
IOP
IOP BUSY
DATAO (Ø-17)
CONI (21-35)
CONO CLR
CONO SET
PAR ERR (2Ø)
CHC Ø (21)
CHC 1 (22)
CHC 2 (23)
CHC 3 (24)
BR RQ (25)
SR RQ (26)
INIT (27)
RDY (28)
I/O (29)
DATA MISSED (3Ø)
NON EXIST MEM (31)
JB DN (32)
PIA 33 (33)
PIA 34 (34)
PIA 35 (35)
(IOPI)
CONO (26-35)
IOP SELECT Ø1Ø



Read-Write Timing TD-D-167-0-RWT

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
R3Ø3	B155	WØØ5	B171	B113	B105	RIØ7	R6Ø2	R5Ø2	R6Ø1	R6Ø2	R6Ø2	WSØØ	R123	R2Ø5	R2Ø5	R2Ø5	R181	WSØØ	R123	R2Ø5	R2Ø5	R2Ø5	R181	WSØØ	R123	R2Ø5	R2Ø5	R2Ø5	R181	WØØ5	WØ2Ø
MC NON EXIST	PIA 1,2,3	IOPI ← Ø IOPI WCMA IOPI I/O IOPI RDY MC NONEXST IOPI RDY(B) SBC BR RQ	IOPI ØIØ	IOPI IOB IOPI(I) IOPI IOB ← WCMA(I) IOPI IOB ← WCMA(I) IOB 2Ø	IOPI IOB ← WCMA(I) IOPI IOB ← IOPI(I) WC=I	IOPI WCMA ← IOB(I) IOPI WCMA ← Ø IOPI ← IOB(I) IOPI ← Ø	IOPI ← Ø	IOPI WCMA ← Ø	IOPI WCMA Ø	IOPI WCMA IOB(I)	IOPI WCMA ← IOB(I)	IOB Ø IN IOB 1 IN IOB 2 IN IOB 3 IN IOB 4 IN IOB 5 IN IOB 6 IN	IOB Ø-1 IOB 2-3 IOB 4-5	WC Ø WC 1	WC 2 WC 3	WC 4 WC 5	WC Ø-3 WC 3	IOB 6 IN IOB 7 IN IOB 8 IN IOB 9 IN IOB 1Ø IN IOB 11 IN	IOB 6-7 IOB 8-9 IOB 1Ø-11	WC 6 WC 7	WC 8 WC 9	WC 1Ø WC 11	WC 4-1Ø	IOB 12 IN IOB 13 IN IOB 14 IN IOB 15 IN IOB 16 IN IOB 17 IN	IOB 12-13 IOB 14-15 IOB 16-17	WC 12 WC 13	WC 14 WC 15	WC 16 WC 17	WC 1Ø-15	WC Ø WC 1 WC 2 WC 3 WC 4 WC 5 WC 6 WC 7 WC 8 WC 9 WC 1Ø WC 11 WC 12 WC 13 WC 14	WC Ø THRU WC 17
	B155	B117		R2Ø3	R2Ø3	R2Ø1	R2Ø1	R2Ø1	R123	R123		WIØØ	R123	R2Ø5	R2Ø5	R2Ø5	R181	WIØØ	R123	R2Ø5	R2Ø5	R2Ø5	R181	WIØØ	R123	R2Ø5	R2Ø5	R2Ø5	R181	WØØ5	WØ2Ø
	PIA 4,5,6,7	PIA 1,2,3	IOPI RDY IOPI IOPI DATA MISS	IOPI PIA 33 IOPI PIA 34 IOPI PIA 35	IOPI I/O IOPI DATA MISSED IOPI NON-EX MEM	IOPI JB DN	IOPI RDY	IOPI INIT	IOB 3Ø-31 IOB 32-33 IOB 34-35	IOB 27-28 IOB 29		IOB 18 IN IOB 19 IN IOB 2Ø IN IOB 21 IN IOB 22 IN IOB 23 IN	IOB 18-19 IOB 2Ø-21 IOB 22-23	MA 18 MA 19	MA 2Ø MA 21	MA 22 MA 23	MA 18-21 MA 21	IOB 24 IN IOB 25 IN IOB 26 IN IOB 27 IN IOB 28 IN IOB 29 IN	IOB 24-25 IOB 26-27 IOB 28-29	MA 24 MA 25	MA 26 MA 27	MA 28 MA 29	MA 22-27 MA 27	IOB 3Ø IN IOB 31 IN IOB 32 IN IOB 33 IN IOB 34 IN IOB 35 IN	IOB 3Ø-31 IOB 32-33 IOB 34-35	MA 3Ø MA 31	MA 32 MA 33	MA 34 MA 35	MA 28-33	WC 15 WC 16 WC 17 MA 18 MA 19 MA 2Ø MA 21 MA 22 MA 23 MA 24 MA 25 MA 26 MA 27 MA 28 MA 29	MA 18 THRU MA 26 THRU MA 35
WØ28	WØ28	WØ28	WØ28	WØ28	WØ28	WØ28	WØ28	WØ28	WØ28	WØ28	WØ28	WØ28	WØ28	WØ28	WØ28	B684	B684	B684	B684	B684	B684	B684	B684	B684	B684	B684	B684	B684	B36Ø	WØØ5	WØ2Ø
IOB Ø THRU IOB 8	IOB 9 THRU IOB 17	IOB 18 THRU IOB 26	IOB 27 THRU IOB 35	IO BUS	IO BUS	IO BUS	IO BUS	IO BUS	IO BUS	IO BUS	IO BUS	IOB 27 THRU IOB 35	IOB 18 THRU IOB 26	IOB 9 THRU IOB 17	IOB Ø THRU IOB 8	MA MEM BUS	MA MEM BUS	MA MEM BUS	MA MEM BUS	MA MEM BUS	MA MEM BUS	MA MEM BUS	MA MEM BUS	MA MEM BUS	MA MEM BUS	MA MEM BUS	MA MEM BUS	MA MEM BUS IOPI RDY(B)	SRSH CHC COUNT	MA 3Ø MA 31 MA 32 MA 33 MA 34 MA 35 INIT(I) RDY(I) OUT DATA MIS(I) NON-EXMEM(I) JB DN(I) PIA33(I) PIA34(I) PIA35(I)	PAR ERR MISC INDICATORS
W028	W028	W028	W028	W607	B684	B684	B684	B105	R204	B105	B105	W607	B113	W607	B124	B602	W102	R602		R602	B105	R602	B36Ø	B36Ø	B36Ø	B36Ø	B36Ø	WØ2Ø	WØ2Ø	WØ2Ø	WØ28
MA MEM BUS	MA MEM BUS	MA MEM BUS	MA MEM BUS	MC BUS ← BR(I) (Ø-11) MC BUS ← BR(I) (12-23) MC BUS ← BR(I) (24-35)	MC WR RS (B) MC RQ CYC	MC RD RQ MC WR RQ	MC MEM BUS ENABLE LT & RT	MC RQ CYC MC RQ CYC	MC RD MC WR MC RQ SRSH SHIFT	MC RD MC RD MC WR MC WR MC NON EXIST	MC WR MC RQ MC RQ MC RQ PULSE MC BUS ← BR(I) (24-35)	MC WR RS MC ADR ACK MC NON EXIST RD	MC WR RS MC RD RQ PULSE MC NON EXIST RD MC WR RQ PULSE	MC RQ PULSE MC RD RQ PULSE MC WR RQ PULSE	MC ADR ACK MC RS TI SBC CHC PRESET	MC RS TI IOPI ← IOB(I)	PAR PAR	MC NON EXIST	MC COUNT MA	MC COUNT MA MC NON EXIST SBC COUNT WC CHC = Ø	SBC COUNT WC	SRSH T4	SRSH T3	SRSH T2	SRSH T1	SRSH TØ	BR Ø THRU BR 8 THRU BR 17	BR 18 THRU BR 35	SR Ø THRU SR 8 THRU SR 17	SR 18 THRU SR 35

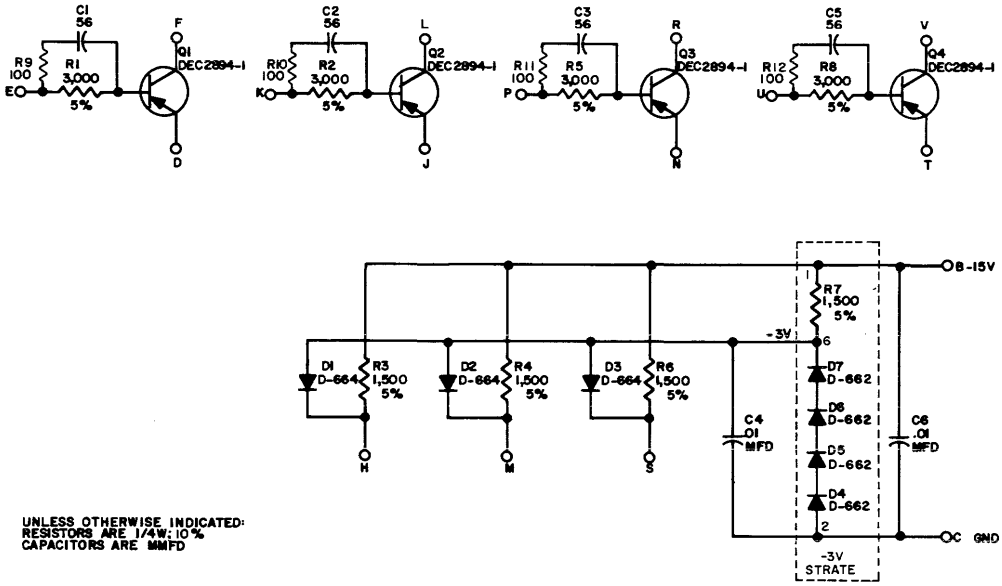
Utilization Module List (Sheet 1) UML-D-167-0-UML

Utilization Module List (Sheet 2) UML-D-167-0-UML

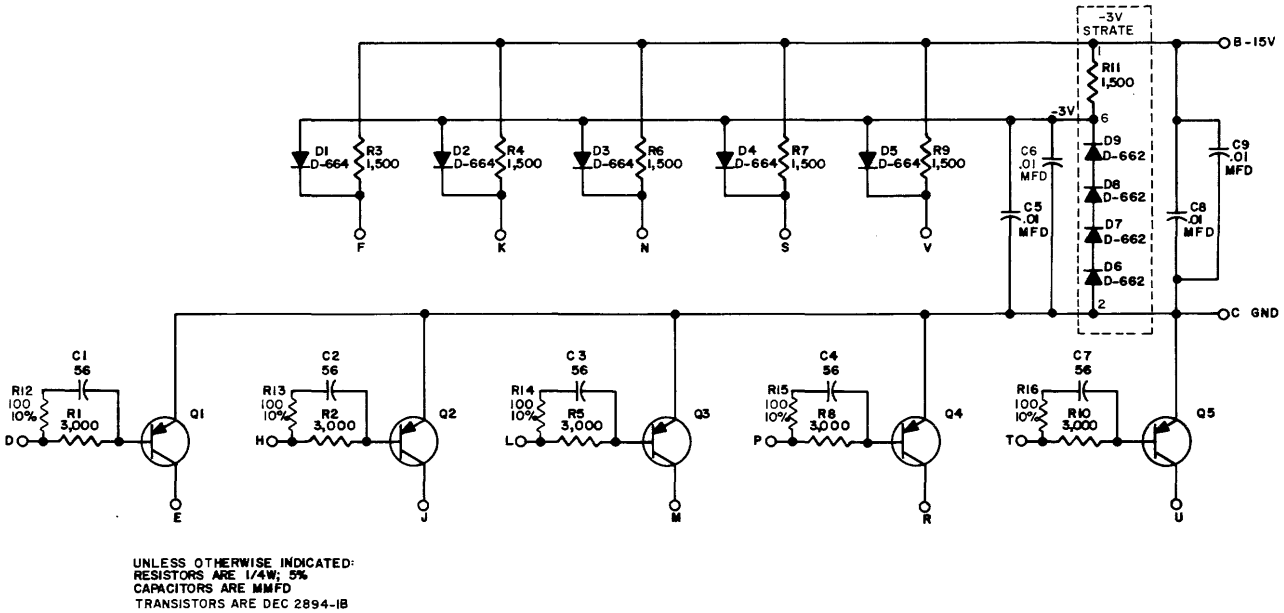
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
B201	B201	B113	B602	B104	B360	B104	B602	B104	B602	B104	W607	B104	W607	R123	B201	B201	B201	B201	B124	B117	W990	W035	W990	B124	B602	B124	B602	B113	W607	B104	W607
SBC BR RQ	SBC SR RQ	SBC BR RQ SBC SR ← BR(TØ)	SBC SR ↔ BR (TØ)	MC BUS ← BR(1) (Ø-11) IOPI DATA MISSED ← 1 IOPI DATA MISSED ← 1 MC BUS ← BR(1) (12-23)	SBC SR ↔ BR (T1)	SBC BR ← Ø SBC BR ← Ø SBC BR ← Ø SBC SR ← Ø Ø-17	SBC BR ← Ø SBC CHC	SBC SR ← Ø Ø-17 SBC SR ← Ø Ø-17 SRC SR ← Ø Ø-17 SBC SR ← Ø Ø-17	SBC SR ← Ø Ø-8 SBC SR ← Ø T8-26	SBC BR ← SR Ø-17 SBC BR ← SR Ø-17 SBC BR ← SR 18-35 SBC BR ← SR 18-26	SBC BR ← SR Ø-17 SBC BR ← SR 18-35 IOP DATA MISSED 1	SBC SR BR(1) Ø-17 SBC SR BR(1) Ø-17 SBC SR BR(1) 18-35 SRSH TØ	SBC SR ← BR(1) Ø-17 SBC SR ← BR(1) 18-35 IOPI ← Ø	IOB 25-26	CHC Ø	CHC 1	CHC 2	CHC 3	SRSH IOP LAST CHAR	CHC = Ø CHC = 7	CHC PRESET JUMPER 30APD	SR 33 SR 34 SR 35 SR 32 SR 33 SR 34 SR 35 WC=1	SRSH TØ-T4 SRSH SHIFT	SRSH 4 Ø-17 (1) SRSH 4 18-35(Ø) SRSH SHIFT	SRSH 4 Ø-17(1) SRSH 4 18-35(Ø)	SRSH 4 Ø-17(Ø) SRSH 4 18-35(1) SRSH IOP BUSY	SRSH 4 Ø-17(Ø) SRSH 4 18-35(1)	SRSH DATA XFER Ø-17 SRSH DATA XFER 18-35 SRSH SHIFT SRSH DCRW IOP	SRSH DATA XFER Ø-17 SRSH DATA XFER 18-35 SRSH DCRW IOP	SRSH DCRW ↔ IOP SRSH DCRW ↔ IOP SRSH IOP LAST CHAR SBC SR RQ	SBC COUNT WC
W028	W028	W028	W028	W102	W102	W102	W102	W102	W102	W102	W102	W102	B204	B113	B204	B113	B204	B113	B204	B113	B204	B113	B204	B113	B204	B113	B204	B113	B204	B113	W020
MB Ø THRU MB 8	MB 9 THRU MB 17	MB 18 THRU MB 26	MB 27 THRU MB 35	MB Ø MB 1 MB 2 MB 3	MB 4 MB 5 MB 6 MB 7	MB 8 MB 9 MB 10 MB 11	MB 12 MB 13 MB 14 MB 15	MB 16 MB 17 MB 18 MB 19	MB 20 MB 21 MB 22 MB 23	MB 24 MB 25 MB 26 MB 27	MB 28 MB 29 MB 30 MB 31	MB 32 MB 33 MB 34 MB 35	BR Ø BR 1 BR 2 BR 3	BR Ø BR 1 BR 2 BR 3	BR 4 BR 5 BR 6 BR 7	BR 4 BR 5 BR 6 BR 7	BR 8 BR 9 BR 10 BR 11	BR 8 BR 9 BR 10 BR 11	BR 12 BR 13 BR 14 BR 15	BR 12 BR 13 BR 14 BR 15	BR 16 BR 17 BR 18 BR 19	BR 46 BR 17 BR 18 BR 19	BR 20 BR 21 BR 22 BR 23	BR 20 BR 21 BR 22 BR 23	BR 24 BR 25 BR 26 BR 27	BR 28 BR 29 BR 30 BR 31	BR 28 BR 29 BR 30 BR 31	BR 32 BR 33 BR 34 BR 35	BR 32 BR 33 BR 34 BR 35	PAR ERR STOP 1 PAR(1) MC RD MC WR MC RQ SRSH SHIFT
B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201	B201
SR Ø	SR 1	SR 2	SR 3	SR 4	SR 5	SR 6	SR 7	SR 8	SR 9	SR 10	SR 11	SR 12	SR 13	SR 14	SR 15	SR 16	SR 17	SR 18	SR 19	SR 20	SR 21	SR 22	SR 23	SR 24	SR 25	SR 26	SR 27	SR 28	SR 29	SR 30	SR 31
B113	B113	B113	B113	B113	B113	B113	B113	B113	B113	B113	B113	B113	B113	B113	B113	B113	B113	B105	B201	B201	B201	B201	W028	W028	W028	W028	W028	W028	W028	W028	W028
SR Ø SR 1 SR 2 SR 3	SR Ø SR 1 SR 2 SR 3	SR 4 SR 5 SR 6 SR 7	SR 4 SR 5 SR 6 SR 7	SR 8 SR 9 SR 10 SR 11	SR 8 SR 9 SR 10 SR 11	SR 12 SR 13 SR 14 SR 15	SR 12 SR 13 SR 14 SR 15	SR 16 SR 17 SR 18 SR 19	SR 16 SR 17 SR 18 SR 19	SR 20 SR 21 SR 22 SR 23	SR 20 SR 21 SR 22 SR 23	SR 24 SR 25 SR 26 SR 27	SR 24 SR 25 SR 26 SR 27	SR 28 SR 29 SR 30 SR 31	SR 28 SR 29 SR 30 SR 31	SR 32 SR 33 SR 34 SR 35	SR 32 SR 33 SR 34 SR 35	PAR ODD	SR 32	SR 33	SR 34	SR 35	DATA Ø THRU DATA 8	DATA 9 THRU DATA 17	DATA 18 THRU DATA 26	DATA 27 THRU DATA 35	SR Ø THRU SR 8	SR 9 THRU SR 17	SR 18 THRU SR 26	SR 27 THRU SR 35	DCRW ↔ IOP BUS

Utilization Module List (Sheet 2) UML-D-167-0-UML

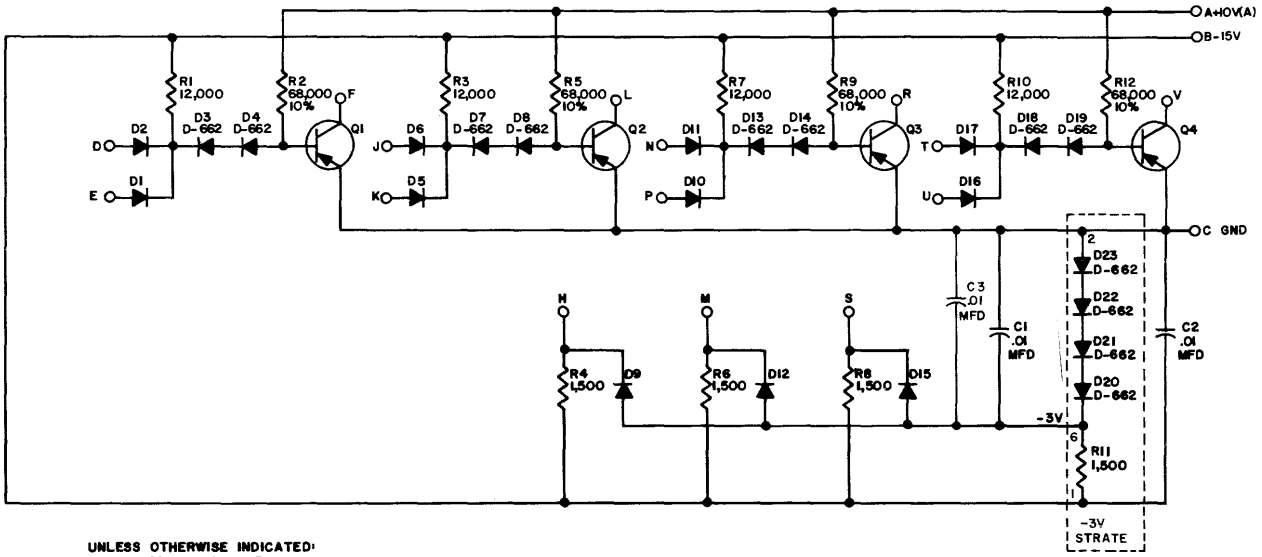
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
B105	B502	B201	B204	B105	B360	B602	B104	B130	B130	B130	B130	B130	B130	B130	B130	B130	B130	B130	B130	B130	B130	B130	B130	B130	B130	B105	B502	B24	B22	B23	B23
PAR ERR STOP				PAR RD DATA			PAR RD DATA																			SBC SR←Ø (9-7)				
				PAR RD DATA	PAR CK PULSE	PAR RD DATA																					SBC SR←Ø (9-7)	SRSH3 Ø-17	SRSH3 18-35	SRSH3 Ø-17	SRSH3 18-35
PAR ERR STOP	PAR ERR	PAR ERR					PAR RD DATA	PAR ODD Ø-8	PAR ODD Ø-8	PAR ODD Ø-8	PAR ODD Ø-8	PAR ODD 9-17	PAR ODD 9-17	PAR ODD 9-17	PAR ODD 9-17	PAR ODD	PAR ODD	PAR ODD 18-26	PAR ODD 18-26	PAR ODD 18-26	PAR ODD 18-26	PAR ODD 27-35	PAR ODD 27-35	PAR ODD 27-35	PAR ODD 27-35						
																											SBC SR←Ø (27-35)				
																											SBC SR←Ø (27-35)	SRSH3 Ø-7	SRSH3 18-35	SRSH3 Ø-17	SRSH3 18-35
PAR ERR STOP			PAR																												



Inverter CS-B-B104

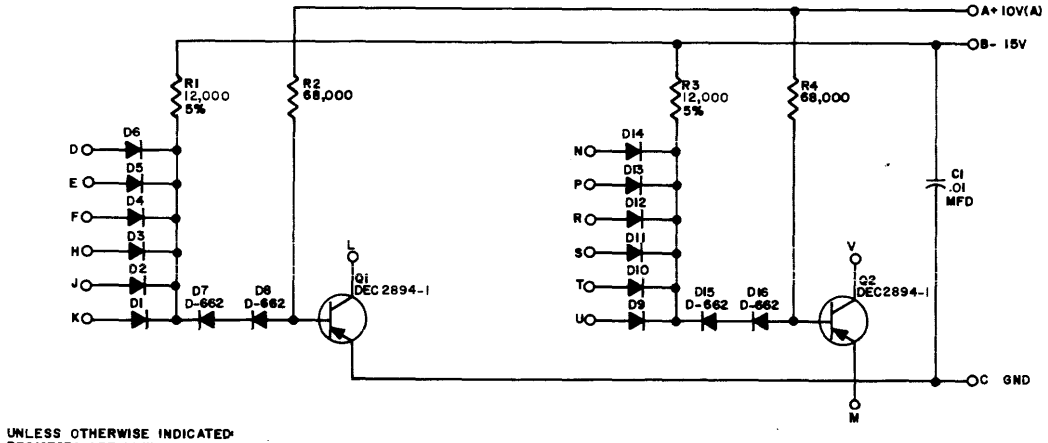


Inverter CS-B-B105



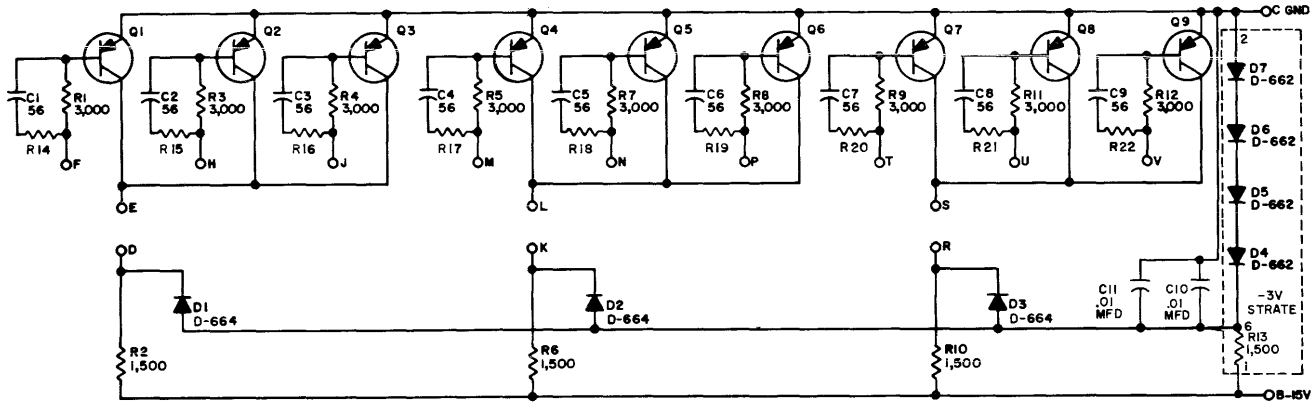
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 5%
TRANSISTORS ARE DEC2894-1B
DIODES ARE D-664

Diode Gate RS-B-B113



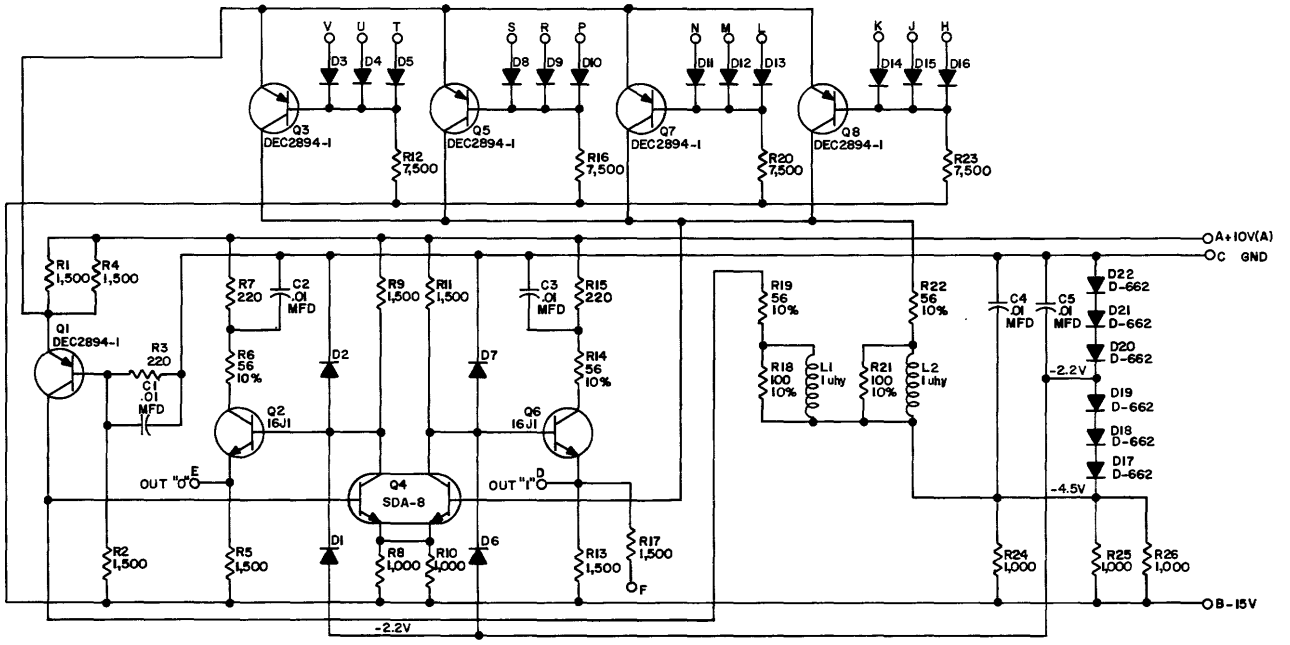
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%
DIODES ARE D-664

Diode Gate CS-B-B117



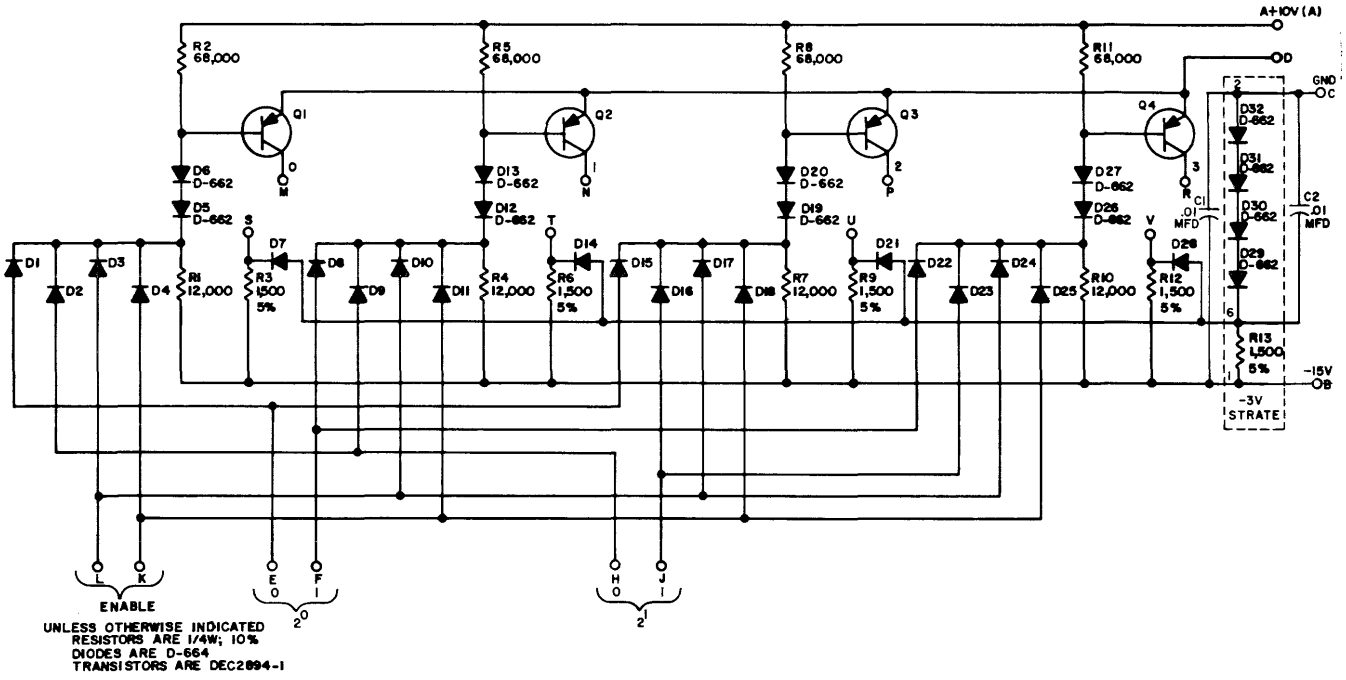
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W, 5%
CAPACITORS ARE MMFD
TRANSISTORS ARE DEC 2894-I
RESISTORS ARE 100; 1/4 W, 10%

Inverter RS-B-B124

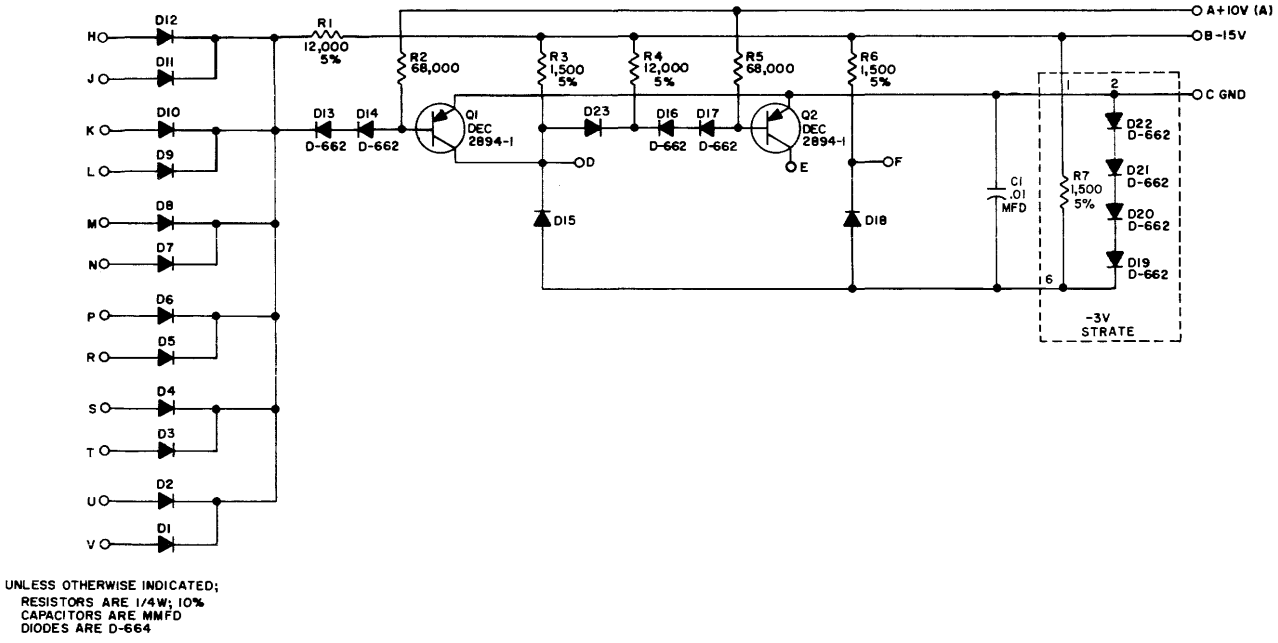


UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 5%
DIODES ARE D-664

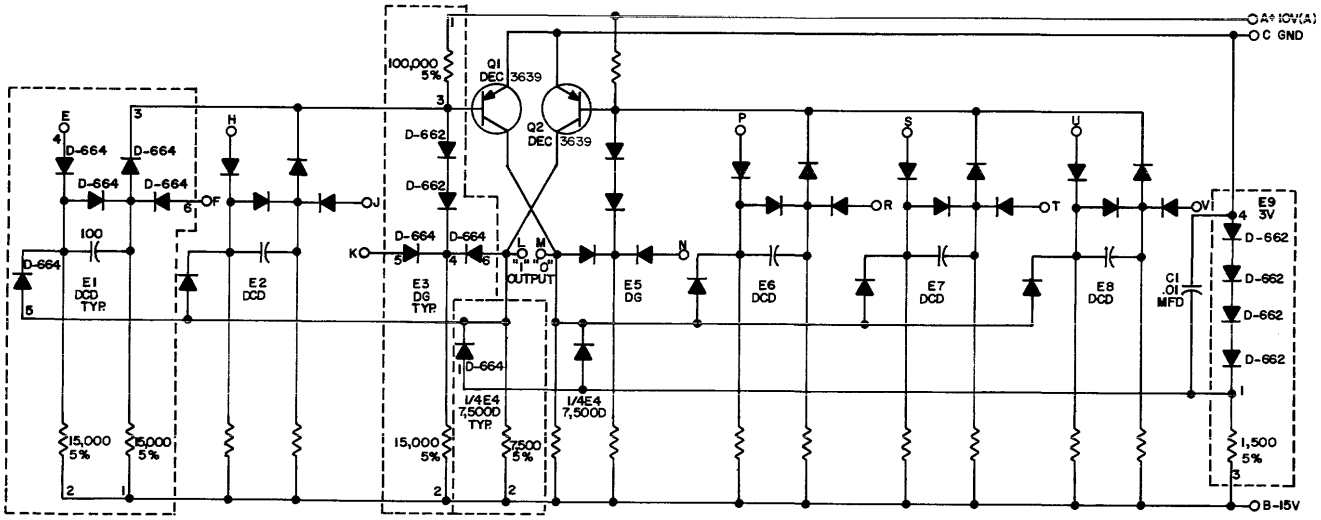
Three-Bit Parity Circuit RS-B-B130



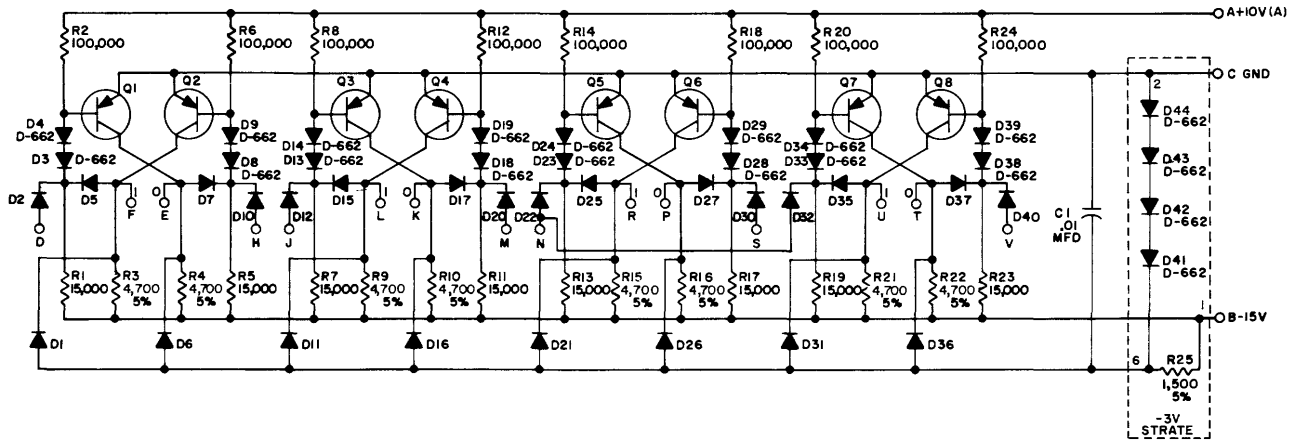
Half Binary to Octal Decoder RS-B-B155



Diode Gate RS-B-B171

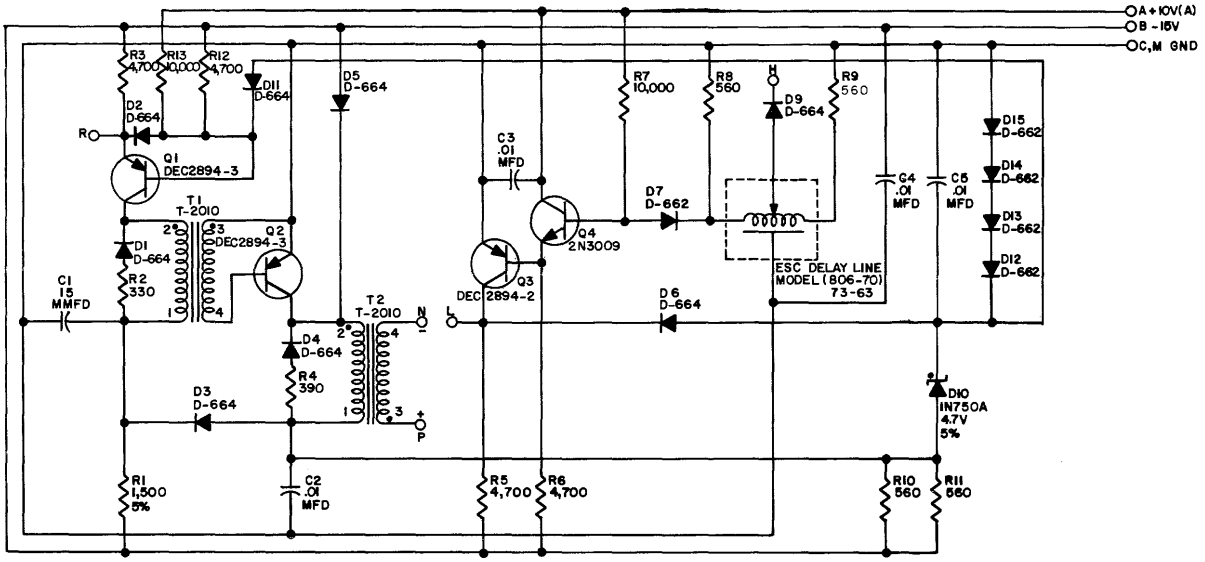


Flip-Flop RS-B-B201



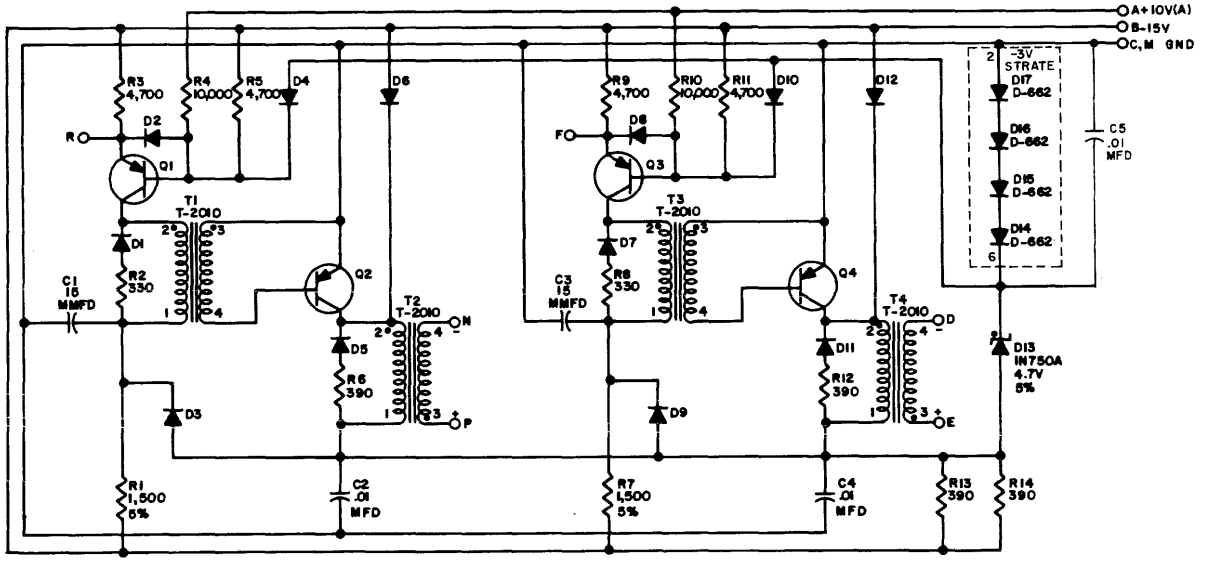
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%
DIODES ARE D-664
TRANSISTORS ARE DEC 3639

Four Flip-Flops CS-B-B204



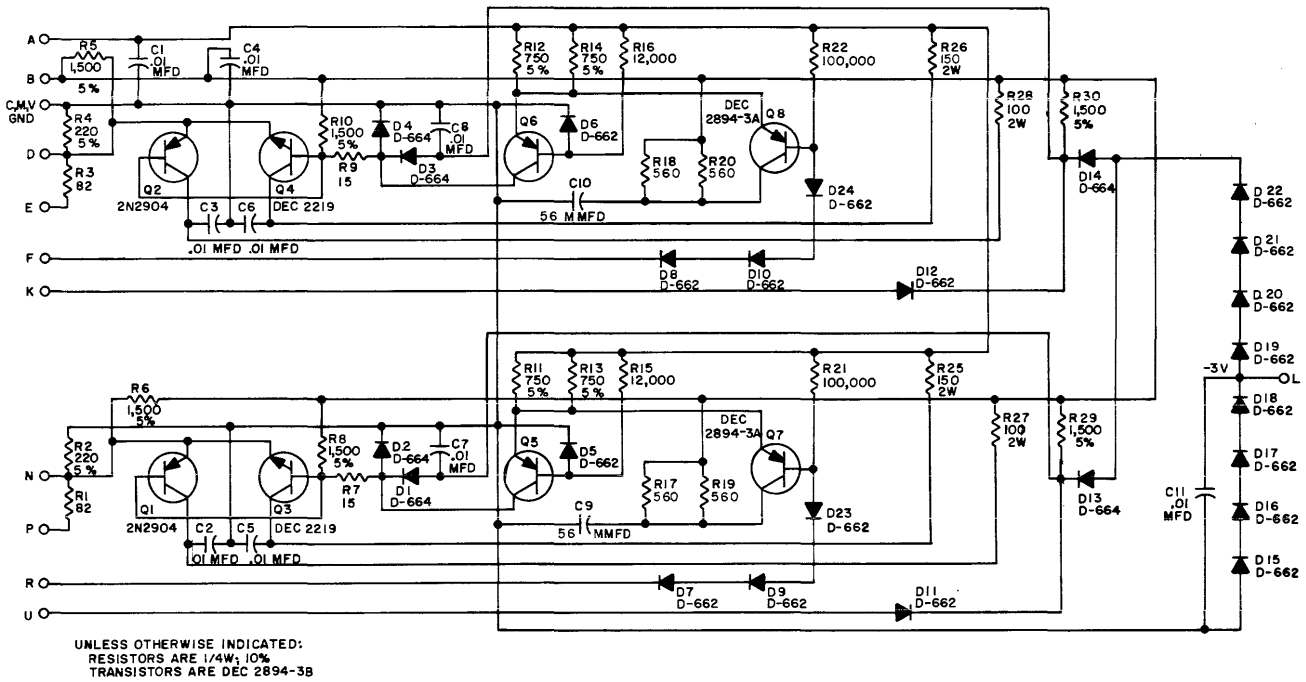
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%

Delay with Pulse Amplifier RS-B-B360



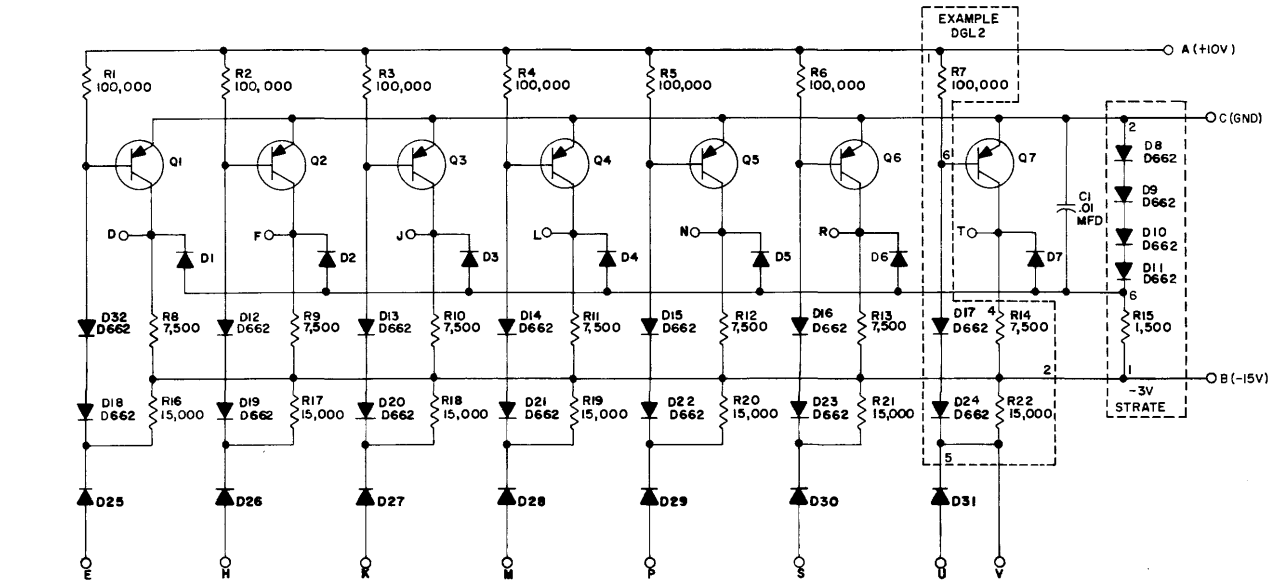
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%
TRANSISTORS ARE DEC2894-3
DIODES ARE D-664

Pulse Amplifier RS-B-B602



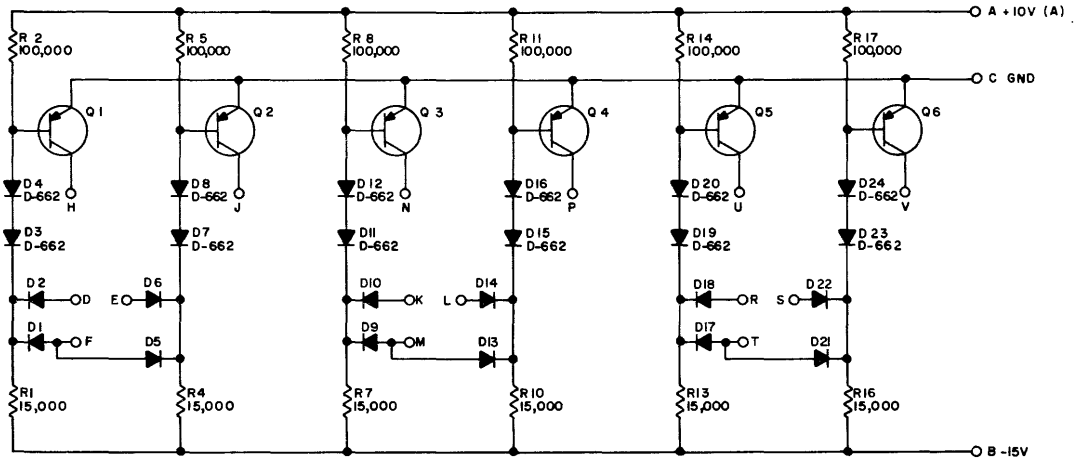
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%
TRANSISTORS ARE DEC 2894-3B

Two Bus Drivers RS-B-B684



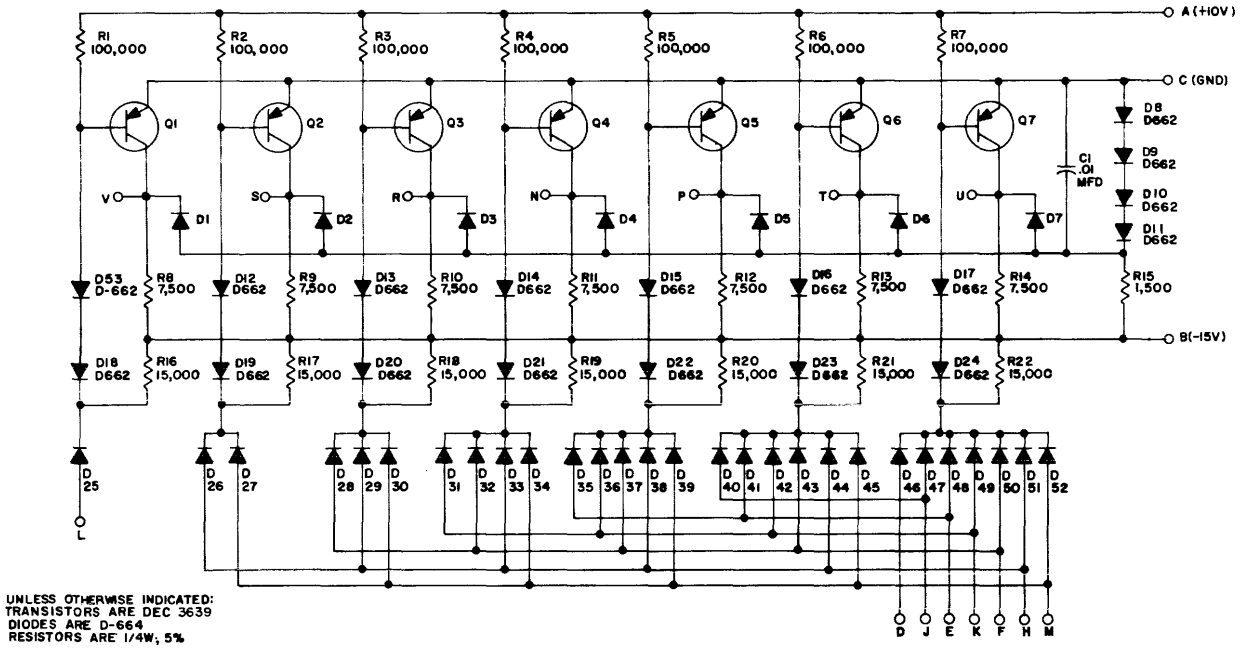
UNLESS OTHERWISE INDICATED;
RESISTORS ARE 1/4W; 5%
DIODES ARE D-664
TRANSISTORS ARE DEC 3639
PRINTED CIRCUIT REV. FOR
DGL BOARD IS SIA

Inverter CS-B-R107



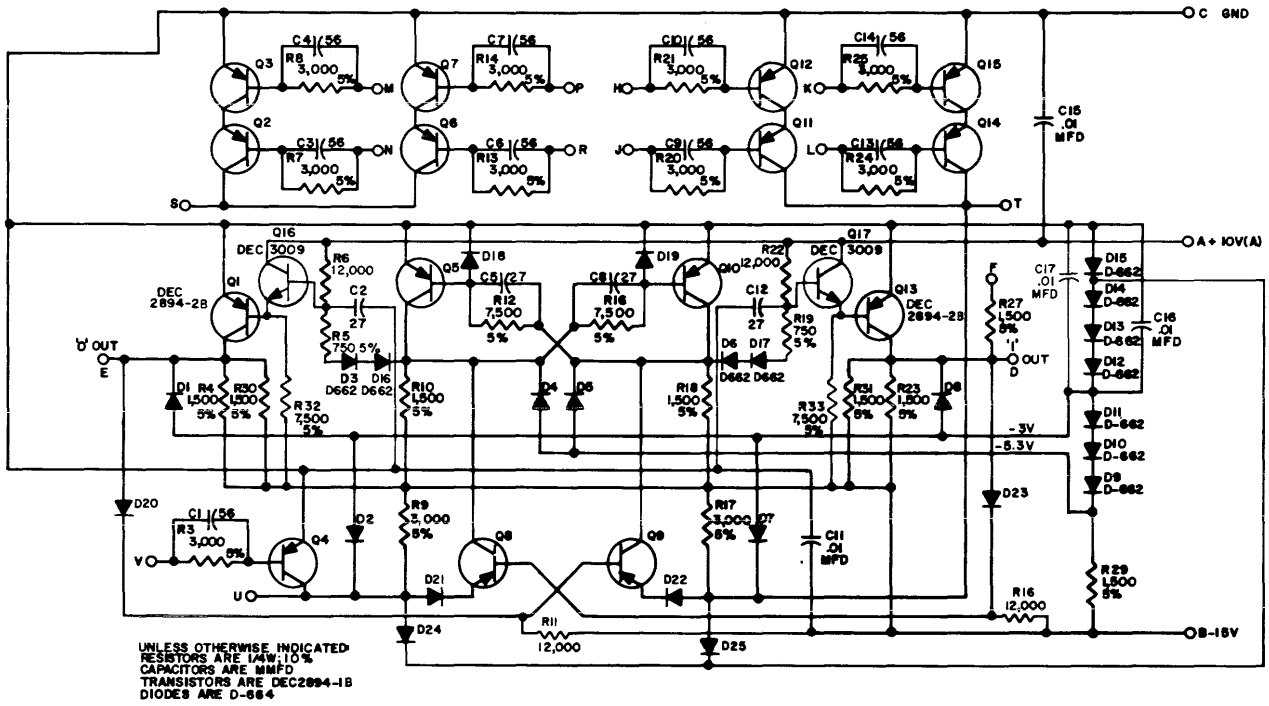
UNLESS OTHERWISE INDICATED:
TRANSISTORS ARE DEC 3639
RESISTORS ARE 1/4 W, 5%
DIODES ARE D-664

Diode Gate RS-B-R123

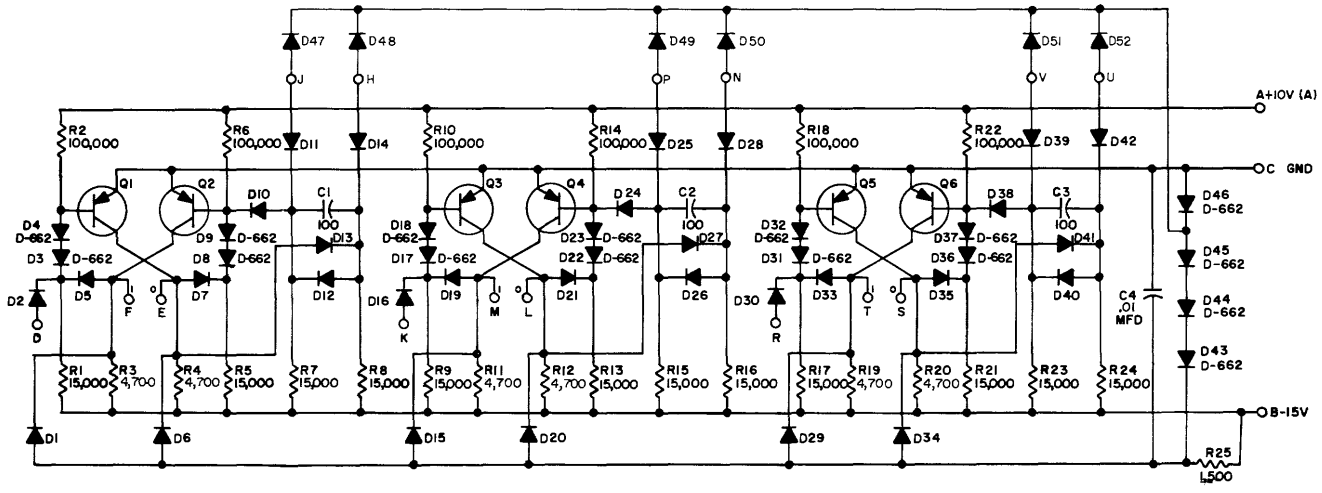


UNLESS OTHERWISE INDICATED:
TRANSISTORS ARE DEC 3639
DIODES ARE D-664
RESISTORS ARE 1/4W, 5%

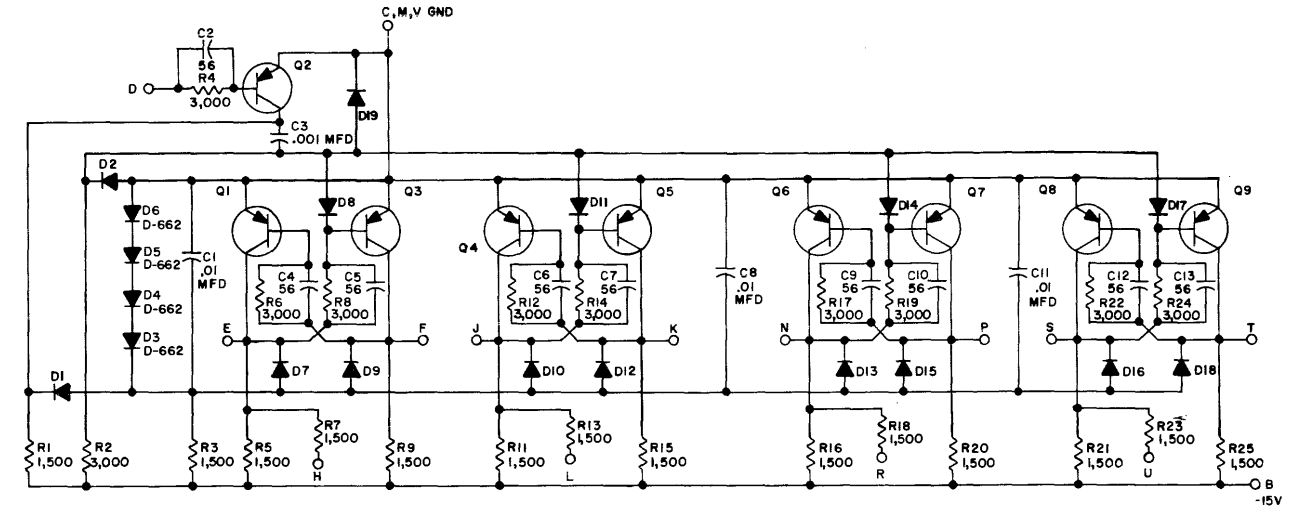
DC Carry Chain CS-B-R181



Flip-Flop RS-B-R201

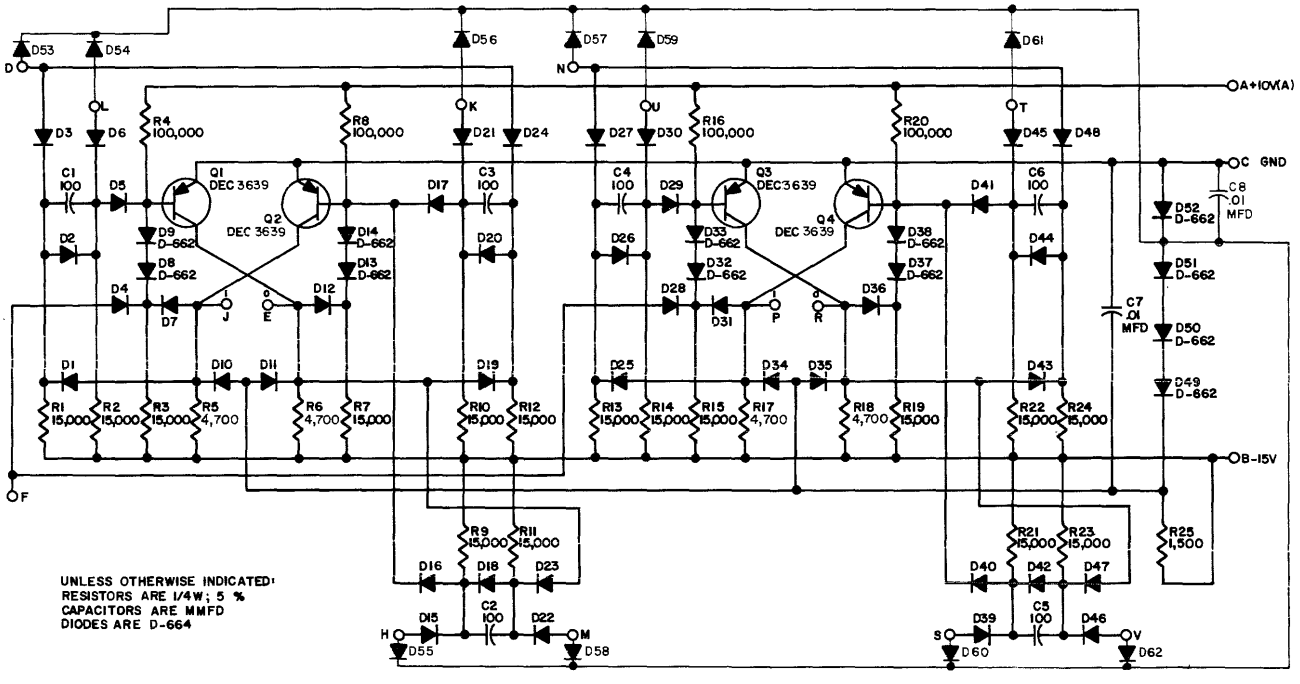


Triple Flip-Flop RS-B-R203



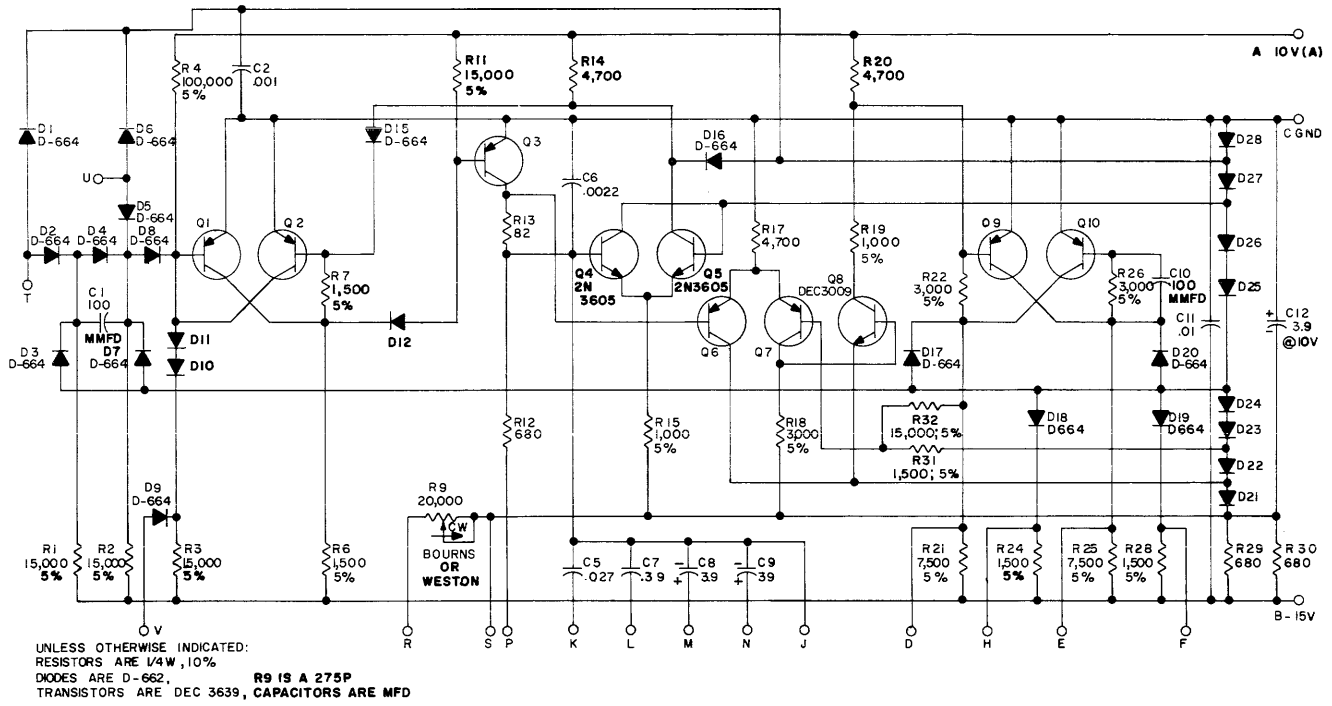
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 5%
DIODES ARE D-664
CAPACITORS ARE MMFD
TRANSISTORS ARE DEC2894-18

Quadruple Flip-Flop RS-B-R204

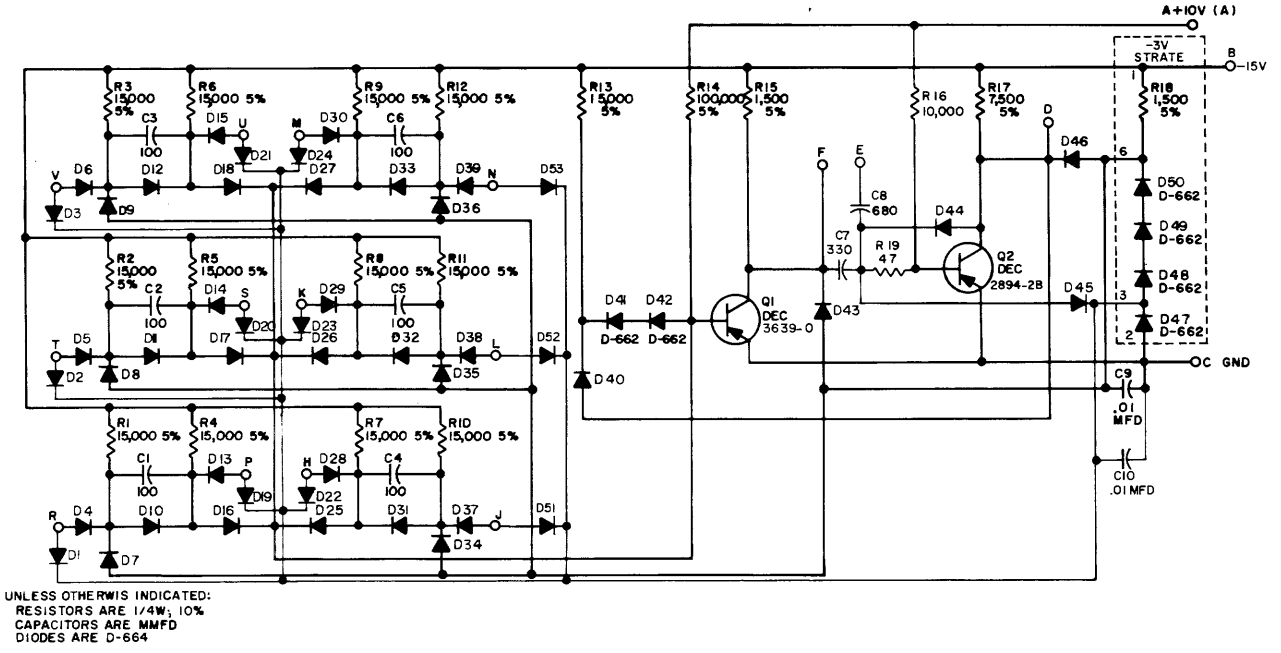


UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 5 %
CAPACITORS ARE MMFD
DIODES ARE D-664

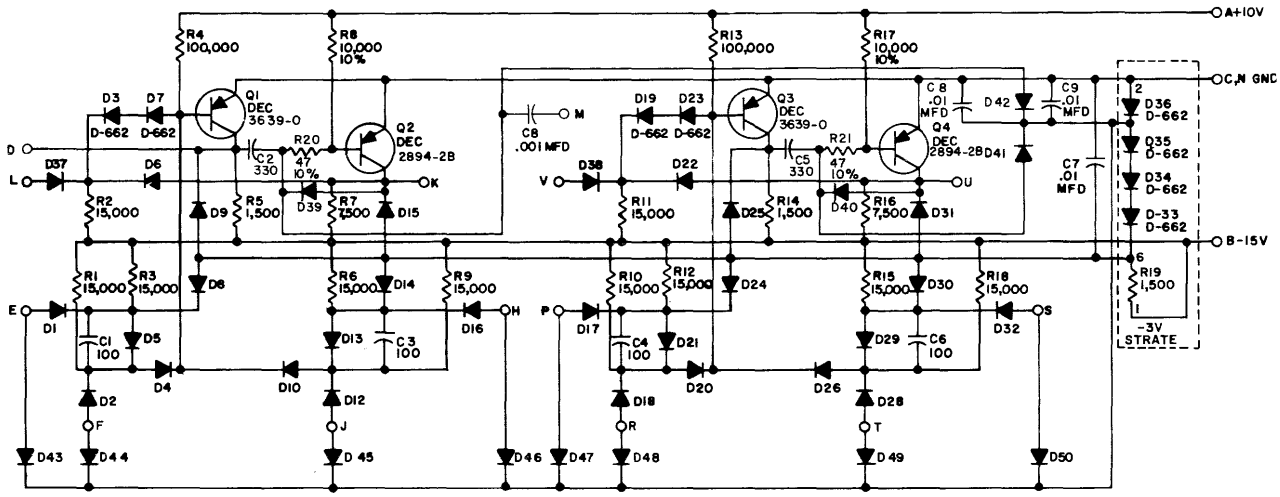
Dual Flip-Flop RS-B-R205



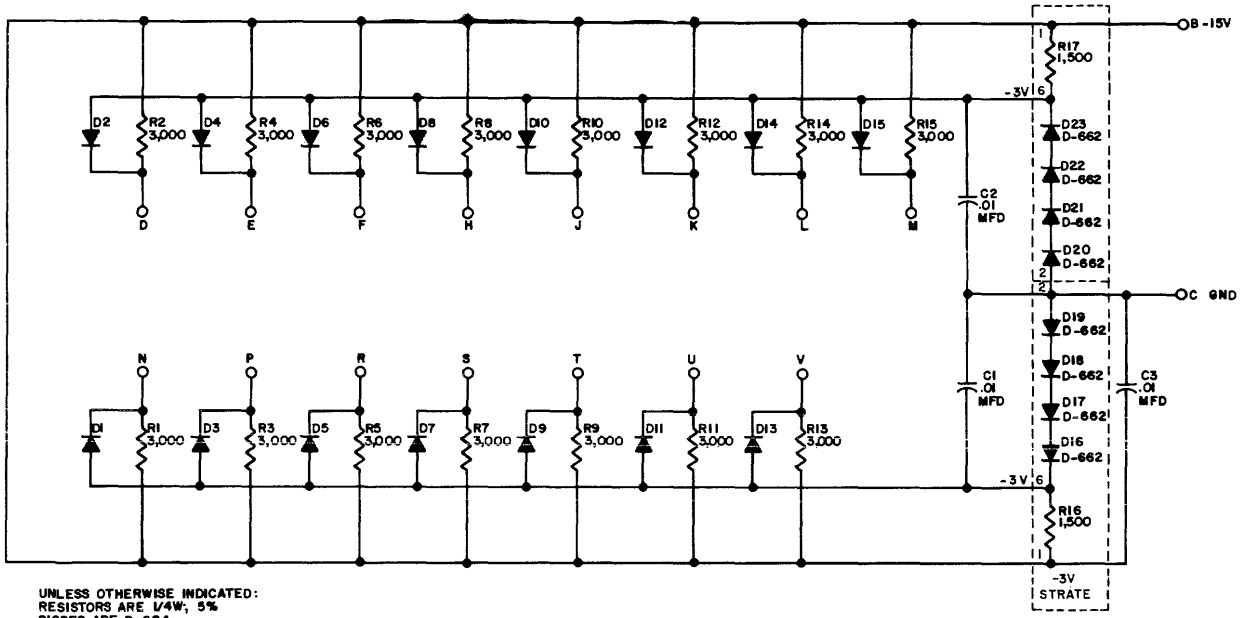
Intergrating One-Shot CS-B-R303



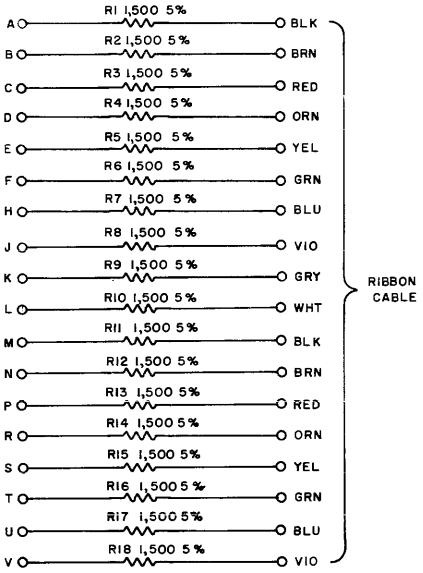
Pulse Amplifier RS-B-R601



Pulse Amplifier CS-B-R602

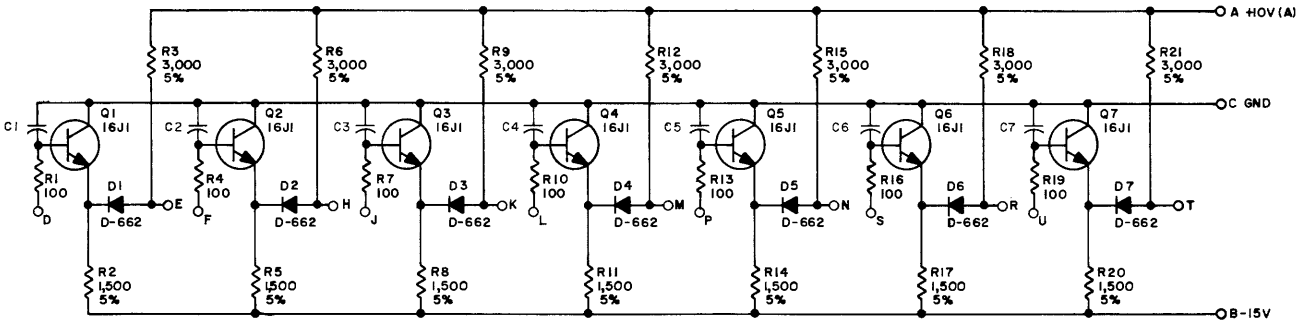


Clamped Loads CS-B-W005



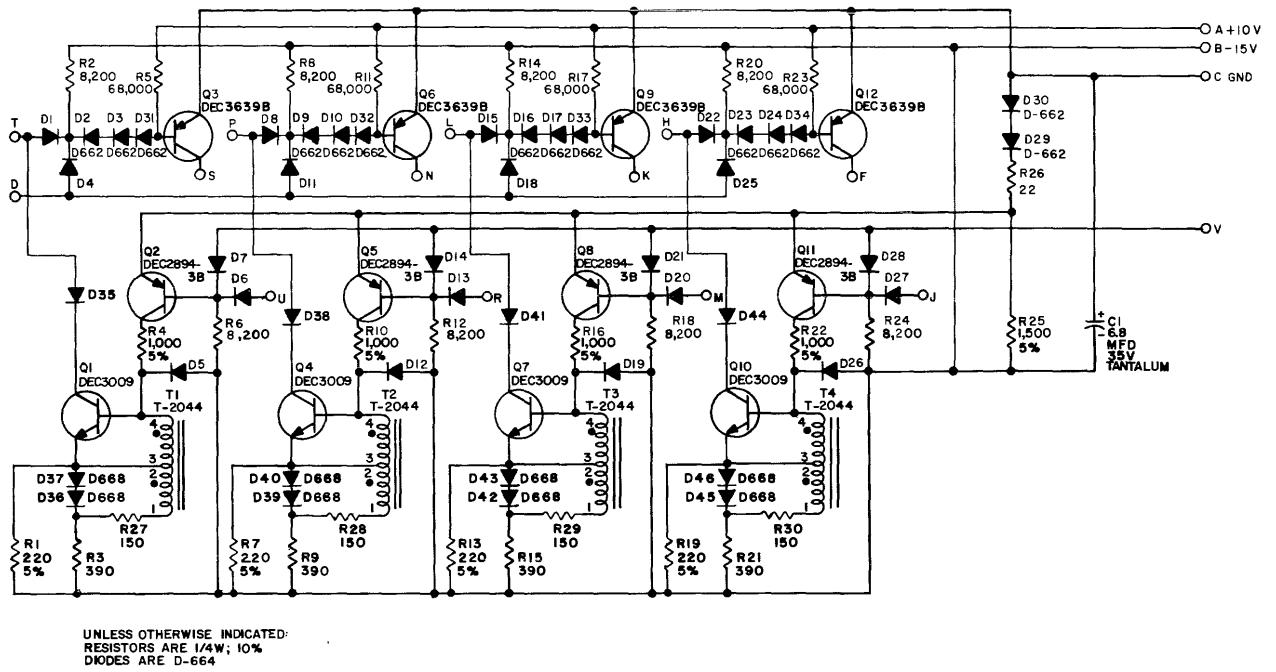
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W

Indicator Cable Connector RS-B-W020



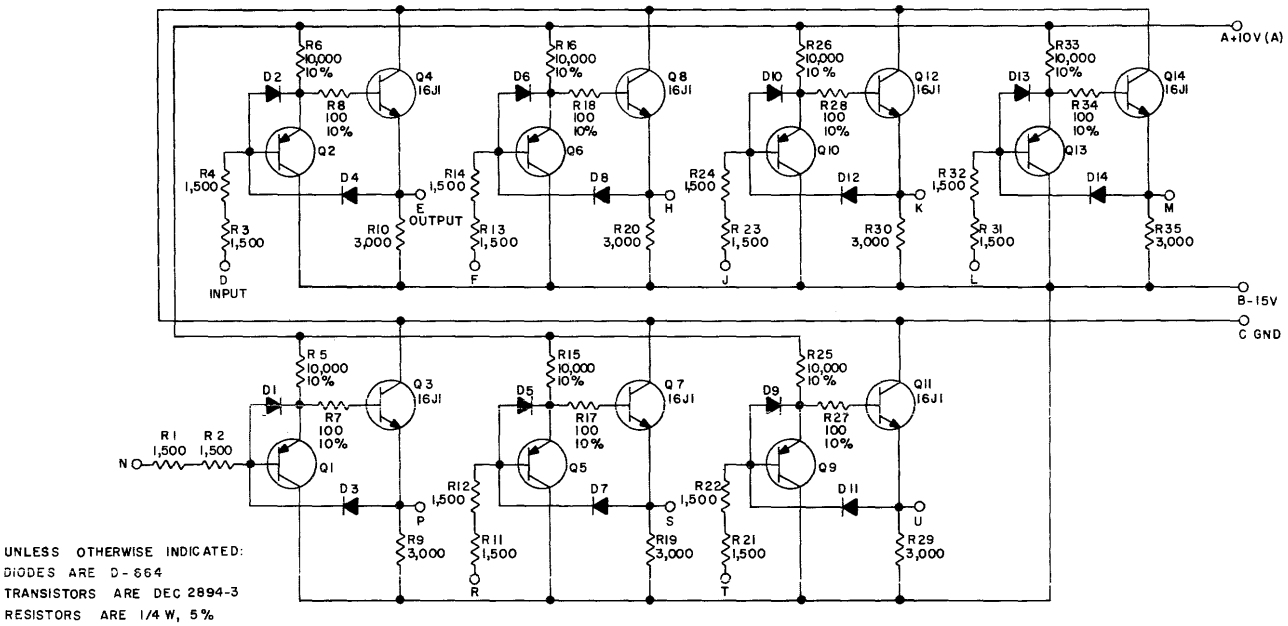
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%
CAPACITORS ARE 120 MMFD

Emitter Follower RS-B-W100



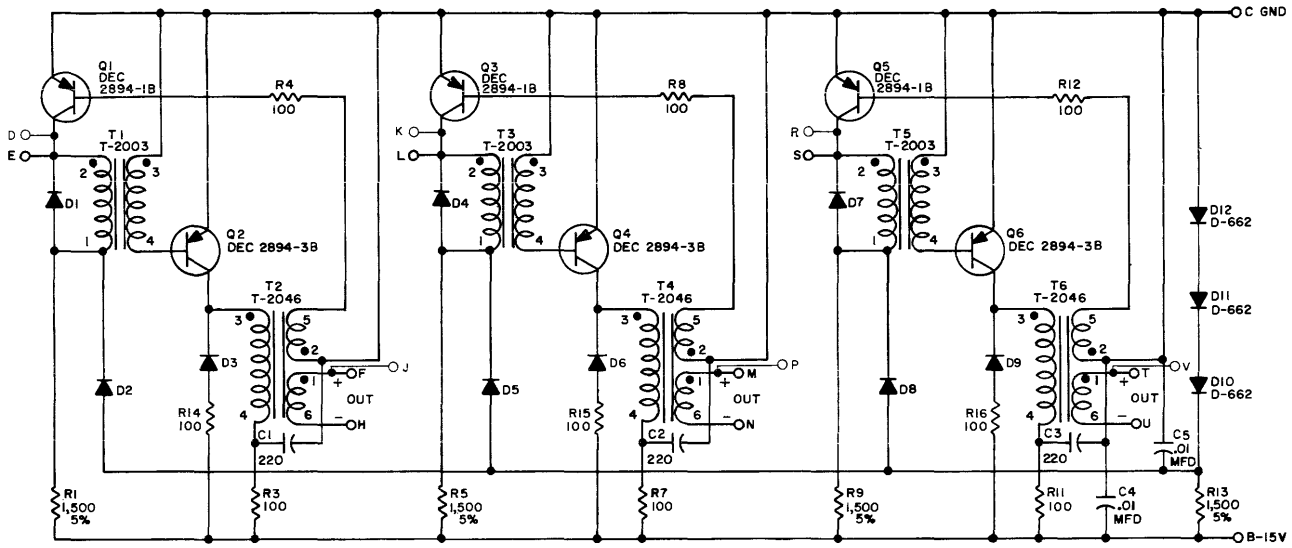
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%
DIODES ARE D-664

Pulse Bus Transceiver CS-B-W102



UNLESS OTHERWISE INDICATED:
DIODES ARE D-664
TRANSISTORS ARE DEC 2894-3
RESISTORS ARE 1/4 W, 5%

High Impedance Follower RS-B-W500



UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%
CAPACITORS ARE MMFD
DIODES ARE D-664

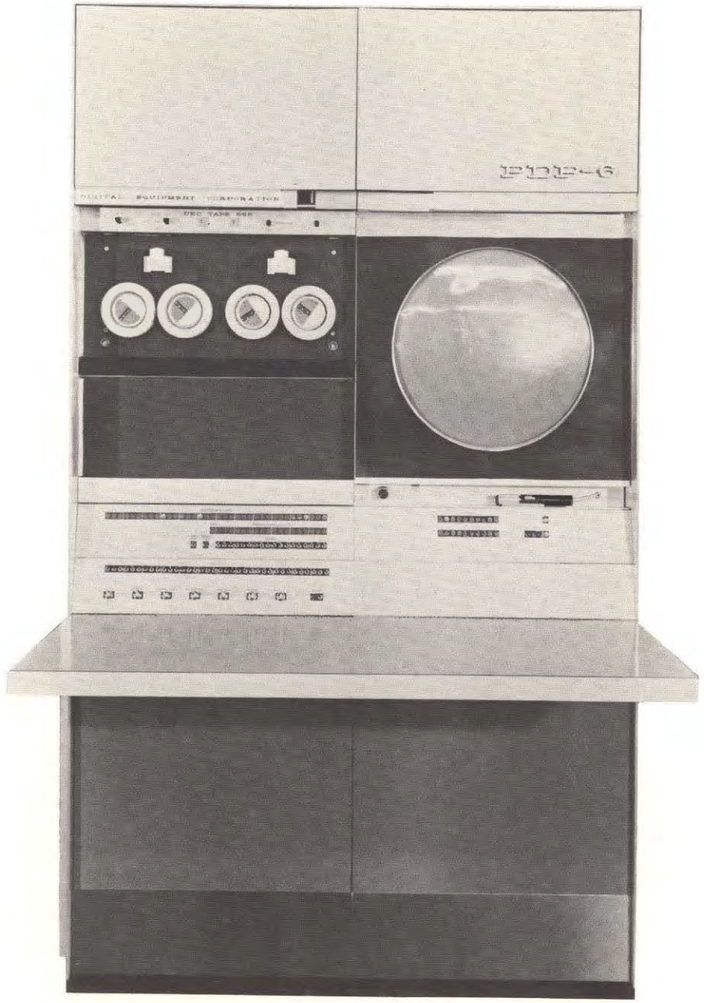
Pulse Amplifier CS-B-W607

DIGITAL EQUIPMENT CORPORATION

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

F-61 Printed in U.S.A. 5M 5/63


PDP-6



Programmed Data Processor-6 is a medium scale general purpose computer with a high operating speed, unique logical organization, and growth capability. It is designed for use as a research tool in controlling experiments, collecting data, and automatically analyzing data. In addition, the computer is exceptionally well suited for use in scientific computation centers. The PDP-6 prototype is now under construction, and production deliveries will begin in early 1964.

PDP-6

8 ACCUMULATORS — Increase effective computing speeds

15 INDEX REGISTERS — Provide extensive capability for address modification in program loops

16-WORD FAST MEMORY — Has a 0.5-microsecond cycle time, is used interchangeably for accumulators, index registers, list pointers, program flags, and small program loops

16,384-WORD MAIN MEMORY — Has a 4-microsecond cycle time, is asynchronous with Central Processor

262,144 WORDS MAXIMUM — All directly addressable. Memory can be expanded and mixed in speed. Overlapping increases system speed

520 INSTRUCTIONS — Well organized and complete, provide excellent manipulative power. Features include character manipulation, list processing, and floating point hardware

CENTRAL PROCESSOR

The PDP-6 Central Processor is designed to operate asynchronously with respect to memory. After the memory has transmitted the requested word to the Central Processor, each operates at its maximum speed independent of the other. Eight accumulators and 15 index registers are stored in the first eight memory addresses of the Fast Memory. Most instructions can simultaneously and directly address any of the accumulators and any memory word and use any index register. Thus in 8 microseconds the Processor can make up to four memory references in carrying out one instruction. The asynchronous design assures that all parts of the system are running at top speed and that waiting times are minimized.

The Central Processor also provides control for in-out transfers. For slow in-out devices, like typewriters, the information is moved from the typewriter buffer under program control. The typewriter subroutine normally would be initiated by a program interrupt originating with the typewriter.

For higher speed in-out devices, like magnetic tape, the Central Processor handles the data movement automatically to memory with no program required for individual word transfers. This type of block transfer is started by the program, but then continues with no interruption of normal computing other than the memory cycle required to store or fetch the in-out word.

For some real time applications where absolutely no interruption of normal computing can be allowed, a separate In-Out Processor is used. It moves information in or out of one memory module while the Central Processor is computing with another memory module.

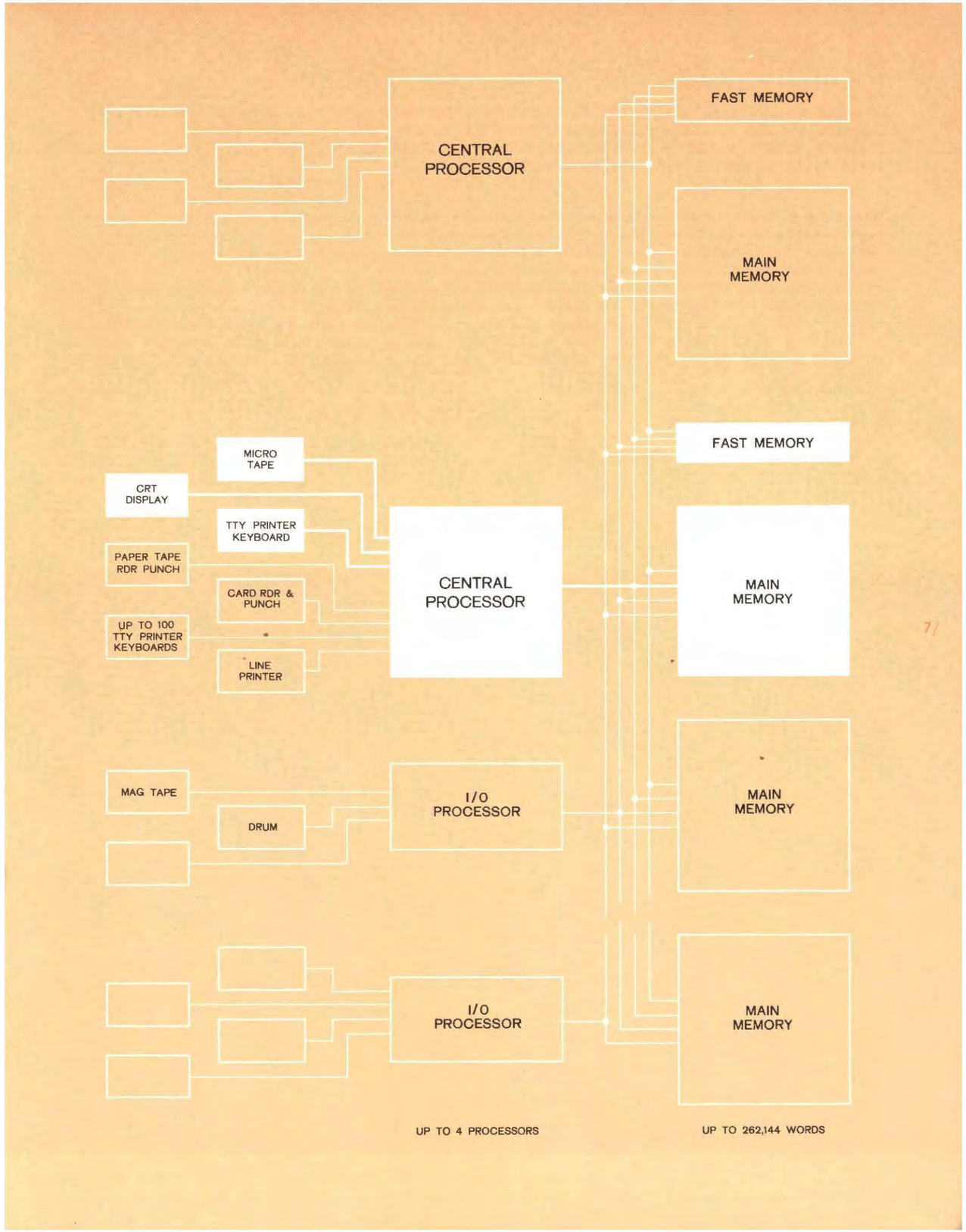
The instruction times for a basic configuration will vary, depending on the way in which the two memories are used. The following examples illustrate add times.

Example 1: add instruction is in Main Memory
operand is in Main Memory
indexing occurs
Complete Execution Time — 8 μsec

Example 2: add instruction is in Fast Memory
operand is in Main Memory
indexing occurs
Complete Execution Time — 3.5 μsec

MEMORY

PDP-6 memory is divided into two parts. Main Memory consists of 8192-word modules having a 4-microsecond cycle time. Fast Memory consists of 16-word modules having a 0.5-microsecond cycle time. Each memory module has facilities for connecting as many as four processors. These can be either Central Processors or In-Out Processors. Memory modules are self contained and can be operated simultaneously in parallel. This can occur with a single processor or with multiple processors.



The Fast Memory is addressable with normal memory addresses. Registers 0-7 can be addressed as index registers and/or accumulators. Memory overlap is automatically achieved due to the asynchronous relationship between the processors and memory.

This short program illustrates one use of the PDP-6 Fast Memory (0.5 microsecond) to increase the speed of performing iterative operations. Four instructions move Table 1 to Table 2 (table size 1000) in the times indicated. Times for performing the same operations exclusively in the Main Memory (4-microsecond) are shown for comparison.

Move "table 1" to "table 2," of length, size

	INSTRUCTIONS			FAST AND MAIN MEM.	MAIN MEM. ONLY
	6,mov	i	1000	(4 + 0.5)*	(4 + 4)*
y	5,mov		table 1,6	4 + 0.5	4 + 4
	5,mov	d	table 2,6	4 + 0.5	4 + 4
	6,soa	g	y	1	4
				10 μsec	20 μsec
				× 1000 = 10 ms	× 1000 = 20 ms

*occurs first time only

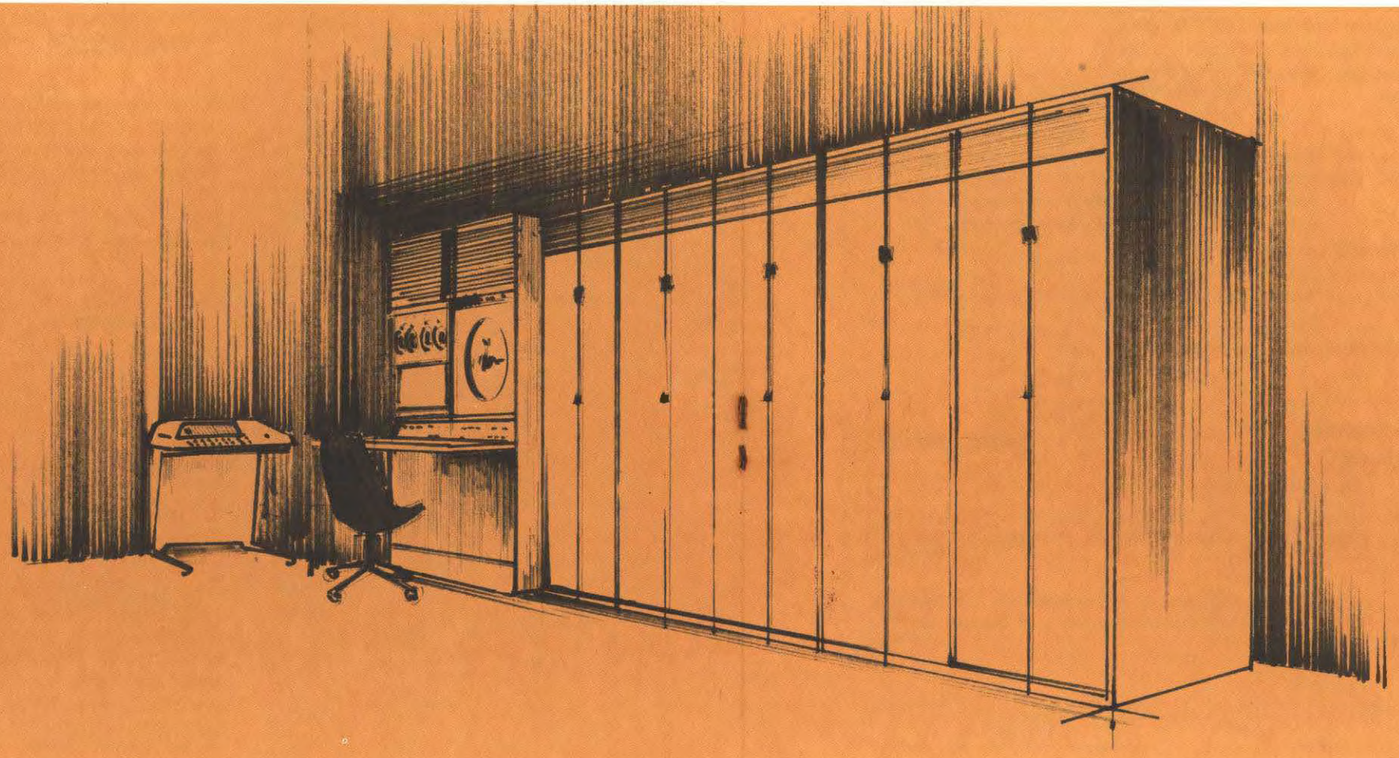
INPUT-OUTPUT EQUIPMENT

Three types of input-output devices are included in the standard PDP-6: a dual Micro Tape Transport (fixed address magnetic tape for high speed loading and readout, as well as program updating), a Teletype Printer-Keyboard, and a Precision CRT Display with Light Pen for direct, high speed man-machine communication.

A variety of optional devices can be added from Digital's standard line of peripheral equipment: high speed line printer, card reader and punch, perforated tape reader and punch, magnetic tape transports and controls, serial and parallel drums, ultra-precision and special purpose digital CRT displays, and other digital equipment compatible with PDP-6.

SOFTWARE

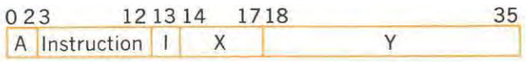
A complete software package is furnished with PDP-6. It includes an algebraic compiler, a symbolic assembler, an executive system, and utility routines. The executive system permits editing of Micro Tape, calls the compiler and assembler when needed, and calls subprograms that facilitate program checkout.



INSTRUCTIONS

PDP-6 has an exceptionally powerful set of 520 instructions, including such special features as immediate, direct, and indirect addressing, push-down list, character handling, memory modification and conditional branching, and operand preservation by one instruction. While the number of instructions is large, they are logically related and consistent. Good organization results in straightforward programming.

The primary instruction format is shown below. The instruction part is made up of the operation and the mode. Normally the operation specifies the action to be taken and the mode determines either the location of the results or conditions to be tested.



Each instruction word has an effective address, computed using I, X, and Y. Y is the address part, which may specify any of 262,144 locations. When the X, or index, part is non-zero, the contents of the specified index register are added to Y to form a new address. When the I, or indirect bit, is 0, the address thus computed is the effective address. In addition each instruction word specifies, with the three bits in A, one of eight accumulators to be used by the instruction.

When the indirect bit is 1, the address computed above is used to reference a memory location whose contents are used to form a new address. This procedure is repeated until a location is referenced whose I bit is 0.

Direct or indirect instructions designate as the effective address a memory register which contains the data to be used. A third type of instruction can specify (by choice of mode) an immediate address, in which case Y itself is taken as the number to be used.

The accumulators and index registers are stored in identical fast memory registers. Their addresses correspond as shown below.

Fast Memory Addresses	Accumulator Addresses	Index Register Addresses	
0	0	10	unused
1	1	11	1
:	:	:	
7	7	17	7
10			
:			
17			

DATA TRANSMISSION

FULL WORD TRANSMISSION

Operations

mov	Move full word
move	Move full word with right and left halves exchanged
movc	Move complement of full word
movn	Move negative of full word
movm	Move absolute magnitude of full word
mmov	Multiple move. Move full words 0, 1 . . .A

HALF WORD TRANSMISSION

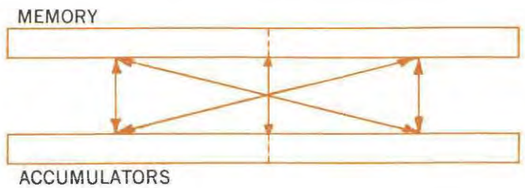
Operations

movr	Move right half only
movl	Move left half only
mvlr	Move left half to right half
mvrl	Move right half to left half

FULL OR HALF WORD TRANSMISSION

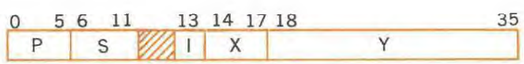
Modes

—	Load accumulator with direct address
i	Load accumulator with immediate address
d	Deposit accumulator with direct address
x	Exchange accumulator with direct address



Full flexibility for making full word or half word transfers between memory and accumulators is built into the PDP-6

CHARACTER MANIPULATION — Characters of arbitrary size are located by a position pointer word of the following format:



I, X, and Y are used in the usual manner to locate the word containing the character. P is the position of the character such that the rightmost bit of the character is 35-P. S is the size of the character, which may be from 1 to 36 bits.

ldc	Load accumulator (right-justified) with character specified by the pointer word. The effective address of the instruction locates the pointer word.
dpc	Deposit character (right-justified) of the accumulator in character position specified by pointer, as above.
cao	Add one character position to character pointer word. The position is moved to the right by the size of the character. If the result is off the right end of the word, the Y part is incremented and the P part is reset to the leftmost character position.
ldci	Same as ldc, but increments pointer word by one character after loading.
dpci	Same as dpc, but increments pointer word by one character after depositing.

PUSH DOWN LIST

psh	Takes the word at the effective address and puts it on the push-down list. The accumulator address part points to the next free location at the completion of the instruction. The left half of the accumulator is counted up by one.
pshs	Push and skip. Same as psh, but skips if the accumulator bits 0-17 are not zero
pop	Subtract one from each half of the accumulator. Take the word specified by the address of the accumulator and put it at the effective address.
pops	Pop and skip. Same as pop, but skips if contents of accumulator bits 0-17 are not zero.

INPUT-OUTPUT — Set initial conditions in I/O equipment, interrogate status, and transfer data between devices and memory.

inm	Input word from device n to memory
inz	Input, mask with immediate, and skip if zero
inn	Input, mask with immediate, and skip if non-zero
inch	Input character from device n to memory
outm	Output word to device n from memory
outi	Output immediate to device n
outch	Output character to device n from memory

ARITHMETIC

FIXED POINT ARITHMETIC — Data (either immediate or direct) and accumulator can be combined using 2's complement arithmetic. Mode specifies location of the results. In general the carry flag will be set if the result is greater than 2^{36}; the overflow flag will be set if the result is greater than 2^{35}.

Operations

add	Add full word
addr	Add to right half only
addl	Add to left half only
addrl	Add right half to left half
addlr	Add left half to right half
adds	Add both halves swapped
sub	Subtract full word
subr	Subtract from right half only
subl	Subtract from left half only
subrl	Subtract right half from right half
sublr	Subtract left half from right half
subs	Subtract right half from left half
iml	Integer multiply
idv	Integer divide
imlu	Integer multiply unsigned
idvu	Integer divide unsigned
mul	Multiply, giving double length result. Low order part goes to A + 1 when appropriate
umul	Unsigned multiply, double length result
div	Divide, giving quotient and remainder. Remainder goes to A + 1 when appropriate
undiv	Unsigned divide.

Modes

—	Direct address combined with accumulator, results to accumulator
d	Direct address combined with accumulator, results to memory
b	Direct address combined with accumulator, results to both memory and accumulator
i	Immediate address combined with accumulator, results to accumulator

FLOATING POINT ARITHMETIC AND CONVERSION — Floating point numbers can be combined using a word format of 1 bit for sign, 8 bits for exponent, and 27 bits for fraction. The mode specifies location of the results. Conversion is fixed-to-floating and floating-to-fixed. The effective address is used as a scaling constant.

Instructions

fad	Floating Add
fsb	Floating Subtract
fmp	Floating Multiply
fdv	Floating Divide

Modes

—	Accumulator is combined with the direct address, results to Accumulator
d	Same, but results to direct address
b	Same, but results to both direct address and accumulator
r	Same but results to accumulator, remainder and low significant bits to accumulator plus one

Instructions

flo	Convert fixed to floating
fix	Convert floating to fixed
dflo	Convert double length fixed to floating
dfix	Convert double length floating to fixed
fsc	Floating point scale

LOGICAL

Operations

and	And
ior	Inclusive Or
xor	Exclusive Or
anc	And complement. Clear selected bits

Modes

—	Direct address combined with accumulator, results to accumulator
d	Direct address combined with accumulator, results to direct address
b	Direct address combined with accumulator, results to direct address and accumulator address
i	Immediate address combined with accumulator, results to accumulator

SHIFTING

There are three kinds of shifting, as follows:

rotate — bits leaving one end enter at the other

arithmetic shift — performs 2's complement multiplication by powers of 2. The sign is unchanged. When going to the right, the first bit shifted in is the Exclusive Or of the sign and overflow, and remaining bits are the sign. When going left, overflow may be set.

logical shift — bits leaving one end are lost, and zeros enter the other end

Shifts may be of an accumulator, A, or of a combined word consisting of A and A + 1.

ral	Rotate accumulator left
rar	Rotate accumulator right
aal	Arithmetic shift accumulator left
aar	Arithmetic shift accumulator right
lal	Logical shift accumulator left
lar	Logical shift accumulator right
rcl	Rotate combined registers left
rcr	Rotate combined registers right
acl	Arithmetic shift combined registers left
acr	Arithmetic shift combined registers right
lcl	Logical shift combined registers left
lcr	Logical shift combined registers right

MEMORY AND ACCUMULATOR MODIFICATION AND TESTING

There are two sets of like instructions in this group:

Memory modification instructions — Modify the direct addressed register. The result is transmitted to the specified accumulator, except that a zero in the A position specifies no accumulator is to be changed. If the condition specified by the mode is met, the next instruction is skipped.

Accumulator modification instructions — Modify the specified accumulator. If the condition specified by the mode is met, control is transferred to the effective address.

Memory Skips	*Accumulator Transfers*	*Operations*
skp	tra	No modification, test full word
skpl	tral	No modification, test left half only
skpr	trar	No modification, test right half only
clm	cla	Clear register
aom	ado	Add one to full word
aoml	adol	Add one to left half, test left half only
aomr	ador	Add one to right half, test right half only
aomb	adob	Add one to both halves, test full word
som	sbo	Subtract one from full word
soml	sbol	Subtract one from left half, test left half only
somr	sbor	Subtract one from right half, test right half only
somb	sbob	Subtract one from both halves, test full word
msw	asw	Swap two halves of word, test full word
cmm	cma	Logical complement
ngm	nga	Negate (take 2's complement)
mgm	mga	Take magnitude (2's complement if negative)

Modes

Memory Skips	*Accumulator Transfers*	*Skip if Condition Met*
—	—	Never
s	t	Always
z	z	Equals zero
n	n	Not equal zero
ge	ge	Greater than or equal to zero
g	g	Greater than zero
le	le	Less than or equal to zero
l	l	Less than zero

COMPARE ACCUMULATOR WITH MEMORY AND SKIP — Compare a data word, which may be an immediate or direct address, with an accumulator by subtraction, and skip according to the mode selection. The results of the comparison do not affect memory or the accumulator.

Instructions

cfw	Compare accumulator with word
cfi	Compare accumulator with address
clh	Compare left half of accumulator with left half of direct addressed word
cli	Compare the left half of accumulator with zero (or left half of immediate address)
crh	Compare right half of accumulator with right half of word
cri	Compare right half of accumulator with address
clr	Compare left half of accumulator with right half of addressed word
clri	Compare left half of accumulator with immediate address
cao	Add one to accumulator, then compare the full words
caoi	Add one to accumulator, then compare the accumulator with the immediate address

Modes

—	Never skip
s	Always skip
e	Skip if equal
n	Skip if not equal
ge	Skip if accumulator is greater than or equal to data
g	Skip if accumulator is greater than data
le	Skip if accumulator is less than or equal to data
l	Skip if accumulator is less than data

ACCUMULATOR BIT MODIFICATION AND TESTING — Bits of an accumulator that are masked by bits of a data word can be tested and/or modified. The data word may be either direct or immediate. Mode specifies whether to use direct or immediate address for data word (normal or right and left halves exchanged) and conditions on which to skip.

Instructions

tst	Test only, do not change accumulator
clb	Clear bits in accumulator masked one by data word
cmb	Complement bits in accumulator masked one by data word
stb	Set bits in accumulator masked one by data word
stst	Test only with the data word, right and left halves exchanged. The data word and accumulator are unchanged
sclb	Clear bits in the accumulator masked one by the data word, right and left halves exchanged
scmb	Complement bits in the accumulator masked one by the data word, right and left halves exchanged
sstb	Set bits in the accumulator, masked one by the data word, right and left halves exchanged

Modes

—	Perform specified operation using directly addressed data
o	Skip if any accumulator bits masked one by the directly addressed data word are one, then perform the specified operation
z	Skip if all accumulator bits masked one by the directly addressed data word are one. Then perform the specified operation
s	Always skip, use directly addressed data word
i	Use immediate addressed data word
io	Same as o, but immediate addressed data word
iz	Same as z, but immediate addressed data word
is	Same as s, but immediate addressed data word

MISCELLANEOUS

SUBROUTINE CALLING

tsp	Transfer and save program counter. The PC and effective address are saved in the right and left halves of the accumulator, respectively. Control is transferred to the effective address.
tsa	Transfer and save accumulator. The accumulator is stored in the effective address. The PC and effective address are saved in the accumulator, as above. Control is transferred to the effective address plus one.
cal	Call subroutine. The accumulator is stored in memory register 20. The PC and effective address are saved in the accumulator as above.
trr	Transfer and restore accumulator. This is the return instruction that matches tsa. The accumulator is replaced by the word addressed in its left half. Control is transferred to the effective address.

EXECUTE

xct	Execute instruction at effective address

NO OPERATION

nop	No operation is performed

HALT

hlt	Halt and transfer

The above specifications are subject to change without notice.

FAST MEMORY TYPE 162
AND
CORE MEMORY TYPE 161C

PDP-6

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

F-67(162, 161C)

PDP-6 INSTRUCTION MANUAL

FAST MEMORY TYPE 162

AND

CORE MEMORY TYPE 161C

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

COPY NO.

CONTENTS

CONTENTS (continued)

ILLUSTRATIONS

ILLUSTRATIONS (continued)

TABLES

ENGINEERING DRAWINGS

ENGINEERING DRAWINGS (continued).

CHAPTER 1

INTRODUCTION

This instruction manual is published to aid personnel in the maintenance of two PDP-6 memories, the Fast Memory Type 162 and Core Memory Type 161C. The first chapter presents a general description of the two memories, lists their operating specifications and describes their physical and electrical characteristics.

The next two chapters present a complete, detailed description of the logic in the memories. The reader is strongly advised not to embark on either of these chapters without first gaining a thorough understanding of the conventions used in the system drawings and of the logic through which access to a memory is requested from a processor. These two topics are presented in Chapters 4 and 7 of the manual for the central processor.

Chapter 4 of this manual describes the circuits associated with the core bank; circuits in the fast memory and in the control portion of the core memory are described in PDP-6 Circuits. Chapter 5 contains information useful in maintaining the memories, including a descriptive listing of applicable Maindecs, and procedures for preventive and corrective maintenance.

All figures of Chapters 2 and 3 and the circuit schematics for Chapter 4 are in Chapter 6. Other figures are interleaved with the text. Following Chapter 6 is a glossary.

The memory system associated with a central processor is often a single fast memory that serves as a scratch pad for the processor, plus a number of core memories which may be shared with other processors of various types. All the memories available to a single processor are connected to a memory bus. Every memory in turn may be connected to as many as four memory buses. A core memory responds to a request for access over any bus, and includes within its logic a priority network that determines which processor has access in cases of conflict (simultaneous requests). The fast memory also may be connected to four buses, but since it functions as a scratch pad for a single processor, it accepts requests only over the bus that is switch-selected by the operator.

Both types of memory store standard PDP-6 words of 36 bits. Fast memory storage is a bank of 16 flip-flop registers which allow rapid access time and nondestructive readout; however, the stored information is lost when system power is turned off. On the other hand, each core memory provides storage for 16,384 words using permanent but slower ferromagnetic cores. Information retrieval from cores is destructive, and hence, for every memory access the core memory executes a complete read-write cycle, reading the contents of the register, then writing either the same or new information.

ACCESS TYPES

A processor may request three types of access to memory:

1. Read, in which the memory subroutine (a hardware subroutine in the processor) retrieves a word from memory.

2. Write, in which the subroutine sends a word to memory to be stored.

3. Read-write, in which the processor combines two memory subroutine calls into one memory access, the first to retrieve a word, the second to store a new one.

The way in which the processor requests access is dependent upon the type of access. The type of access also determines the response of the fast memory, since it may read or write independently. The core memory, however, must execute a complete cycle regardless of the type of access, although a single cycle (in which a pause occurs between the read and write parts) suffices for both subroutine calls of a read-write access.

SYSTEM ORGANIZATION

Data is transferred between processor and memory by pulses in either direction over 36 data lines. Although the processor selects any location out of 262,144 simply by specifying an 18-bit address, the memory system is organized so that the four most significant address bits select a single 16K memory, and the 14 least significant bits select a single location within that memory. The individual address of a memory is determined by its interface wiring with the bus so that only the properly addressed memory responds to control signals on the bus.

Since any octal address from 0 to 17 actually addresses a location in fast memory instead of in the first core memory (i.e., core memory 0), an additional signal is necessary for memory selection. The fast memory selection signal from the processor is asserted if all address bits other than the four least significant address bits are 0's; whereas core memory 0 responds to a request only if the fast memory is not selected. Consequently, the bottom 16 core locations are not available to the program except to store a readin loader, which the operator may call from the processor console.

The logical source of an address depends upon properties of the processor; e.g., whether it has relocation hardware. For convenience the address inputs at the memory end of the bus are labeled as buffered memory address signals (MAB). (To determine the source of these signals the reader should refer to the memory discussion in the appropriate processor manual.) Along with the address, the processor also sends a signal that requests a memory cycle and one or two other signals that specify the type of access. The address, selection, and request signals are supplied as levels. Acknowledgment and restart signals are pulses.

For read access the processor sends a read request and the memory responds by sending an address acknowledgment, the data, and a read restart that triggers the completion of the memory subroutine in the processor. For write access the processor sends a write request; when the memory returns the address acknowledgment, the processor sends the data and a write restart to the memory and completes its own memory subroutine. For read-write access the processor sends both read and write requests at the beginning of the first memory subroutine. The memory responds with the acknowledgment, the data, and the read restart that triggers completion of the subroutine in the processor. At this completion the read restart is suppressed and only the write request remains. In the memory, the cycle stops after completing the read part to await the write restart and new data. When the processor completes computations, it makes a second subroutine call, this time sending the data and the write restart to the waiting memory, and then returns to the processing operations.

FAST MEMORY

The fast memory is always available to the central processor since no other processor can gain access to it. Address bits 32-35 are decoded directly from the bus, so during every access to memory, core or fast, one of the fast locations is selected. However, the fast memory becomes active, i.e., its time chain starts, only when it is actually selected. For read, the fast memory immediately returns the address acknowledgment and the read restart, and strobes the contents of the addressed location onto the bus. For write it sends back the acknowledgment at the same time that it clears the addressed location and then opens a gate so that data pulses sent from the processor go directly to set the appropriate flip-flops. In read-write, fast memory first performs exactly the same functions as in read, then clears the location and awaits the arrival of data pulses sent by the second memory call.

CORE MEMORY

Since the core memory may be called by several processors, a processor making a request may have to wait until the memory is free. The memory is immediately available to unaccompanied requests from any one processor, but conflicting requests must gain access through the priority net. At the start of a cycle, the memory sends back an address acknowledgment and simultaneously loads the address and request signals from the selected bus into its own address register and read and write request flip-flops. The outputs of the address register CMA (core memory address) are applied to the core logic to select the appropriate register in the core bank.

In read access, the processor waits during the read portion of the memory cycle until the memory strobes the sense amplifier outputs, sending the data both over the bus and into its own buffer CMB (core memory buffer). After returning the read restart and disconnecting itself from the bus so that the processor may use some other memory, the active memory completes its cycle, writing the word contained in the CMB back into the same location.

For write access, the acknowledgment causes the processor to send the write restart and data immediately; the memory gates the data into the CMB and disconnects from the bus. It then performs its complete cycle, first reading to clear the location and then writing the CMB information into it.

For read-write access, the memory begins by performing the same function as in read access. The processor completes its first memory subroutine upon receiving the data, but the memory does not disconnect; instead it completes only the read portion of its cycle and then pauses with a gate open to allow data from the bus into the CMB. When the processor makes the second call, it sends the data and the write restart; now the memory disconnects and continues with the write portion of its cycle, writing the new data into the cleared location.

Interleaving

In normal mode the memory is selected by bits 18-21 of the address, and bits 22-35 select the location within the memory. However, to reduce processor waiting time, pairs of core memories n and n + 1, where n is even, may be interleaved. Consecutive addresses as supplied by the processor actually switch access back and forth between the two. Interleaving is accomplished by throwing a switch and making a pair of jumper changes in every memory involved. These changes substitute address bit 35 for bit 21 in the selection of the memory, and substitute 21 for 35 in the address of a location within the selected memory. Then in the first 16K address block from the processor, all even addresses address the even locations in the first memory, all odd addresses the even locations in the second. In the second 16K, all even addresses gain access to the odd locations in the first, all odd addresses to the odd locations in the second. Note that when a fast memory is used with a pair of interleaved core memories, the 16 fast memory registers replace the first 8 even locations in both.

ACCESS TIME

Total cycle time for the core memory is 4.7 μsec, but this time is not a factor in system operating speed unless calls to the same memory are so frequent that processor time is lost in waiting, in which case memories should be interleaved. There is no actual cycle time for the fast memory because it can read and write independently. Of importance when considering system operating speed is the total time required for access by the processor. This time is dependent both on the speed of the memory and the time required by the memory subroutine in the processor.

In Table 1-1 all times are in nsec. The left column lists the actual time consumed within the memory for read and write; this means the time from the appearance of the request signal at the memory until the acknowledgment or the read restart is generated. The right column lists the total access time for a typical processor, the Arithmetic Processor Type 166. The access time is the interval from the pulse that calls

the memory subroutine until the return pulse that restarts the waiting sequence. The times are given for a call without relocation and assuming a bus length of 10 ft. For each additional foot of cable add 3 nsec. If a core address is relocated, an additional 100 nsec allowance is required for access (fast memory addresses are never relocated).

TABLE 1-1 MEMORY OPERATING SPEEDS

	Time in Memory	Type 166 Access Time
Fast Memory		
Read	115	380
Write	155	420
Core Memory		
Read	2485	2750
Write	485	750

For read-write access the time required for the first memory call is that given in the table for read. No time is required within the memory for the second call because it is already waiting for data and the write restart and no response is expected of it. The time required for the call in the processor is merely that required to send the restart and terminate the subroutine (265 nsec in the Type 166).

PHYSICAL CHARACTERISTICS

The memories are housed in standard DEC steel bays with aluminum control panels. The front of each bay can accommodate up to twelve 19 by 5-1/4 inch panels lettered A to N downward (skipping G and I); the top is reserved for a control panel. The remainder of the bay front is enclosed by double doors. Each horizontal logic mounting panel can hold up to 25 DEC plug-in modules numbered from left to right when viewed from the front. Blank panels are mounted in any space not required by the logic.

The fast memory requires a single bay; the core memory requires two. The single fast memory bay is labeled bay 1 and is the left bay of the core memory. In both, bay 1 has a similar layout, a control panel at the top and a special vertical mounting panel occupying positions B and C. In this panel are the 16 oversized modules required for connection of the memory to four buses. Each module has two rear connectors that are bused together; one is connected to a cable from a processor, the other supplies the extension of the bus to other memories. At the last memory a special terminator is plugged into the second connector. In the fast memory, panels D and E hold the double-height modules that contain the flip-flop bank; the

next three hold the remainder of the logic. In the core memory the remainder of bay 1 has three horizontal panels holding the logic for the memory control portion of the system; bay 2 contains the core bank and its associated logic in ten panels. Inside the double doors at the back of every bay is an inner plenum door on which are mounted the required power supplies and power control panels.

Physical dimensions are as follows:

Fast Memory Type 162	Height 69-1/2 inches
	Width 22 inches
	Depth 27 inches
	Weight 250 lbs
Core Memory Type 161C	Height 69-1/2 inches
	Width 42 inches
	Depth 27 inches
	Weight 600 lbs

The above figures are for memories that stand alone. Whenever a memory is added to a frame containing either a processor or another memory, the width is decreased by 2 inches and the weight by 90 lbs. The clearance required for the double doors is 9 inches and for the rear plenum doors 15 inches.

Intake fans at the bottom of every bay cool the logic modules. Additional fans are mounted on the core banks. All equipment operates satisfactorily between temperatures of 50° and 100°F; an ambient temperature between 65° and 75°F is recommended. The floor should be capable of supporting approximately 150 lbs per square foot.

ELECTRICAL CHARACTERISTICS

All PDP-6 memories use standard line power at 105 vac, 60 cps, single phase. Power cables have Hubbell Twist-Lok connectors; both cable and connector are rated at 30 amp. Each memory uses a pair of power controls: the fast memory has Types 836 and 834; the core memory has Types 836 and 832. The 836 contains a diode gate that receives the −15v turnon signals supplied through the marginal check buses from the central processors. The output of the gate goes through the power switch on the control panel and back to the 836 to energize a relay that turns on the other power control. This second control, either 832 or 834, receives the local ac line voltage and provides it to the power supplies. The 834 is a one-step control that turns on the Type 728 Supplies to provide the +10 and −15 vdc required by the fast memory logic. The 832 is a two-step control that provides a fast-on/delayed-off output to the 728 Supplies for the control portion of the core memory, and a delayed-on/fast-off output to the Type 781 that supplies

special voltages for the circuits associated with the core bank. In some systems that have only a single central processor, the pair of controls in each memory is replaced by a Type 829 Two-Step Control, which uses the −15v turnon signal directly (then the fast memory uses only the undelayed output).

Current consumption of the equipment is as follows:

Fast Memory	4.4 amps,	440w
	Turnon surge,	14 amps
Core Memory	9 amps,	1000w
	Turnon surge,	11 amps

All memory logic is solid state; transistors and diodes operate on static logic levels of 0 and −3 vdc (tolerances are 0 to −0.3v and −2.5 to −3.5v). Most logic modules include an internal supply to derive the negative logic level from the −15v supply. PDP-6 uses pulse timing almost exclusively. Pulses are of either polarity depending upon gate input requirements; pulse amplitude is 2.5v from ground with tolerances of +2.3 to +3.0v and −2.3 to −3.5v. Pulses at inverter outputs may be from ground to −3v or vice versa. Pulse widths may be 70 or 40 nsec depending upon the module type. Occasionally an input may be triggered by a level transition instead of a pulse.

In the core memory a Type 781 supplies the special voltages required by the core circuits. These include both fixed dc reference voltages and temperature-variable drive voltages. The 781 provides −35v as bias for the sense amplifiers, drivers, and read-write switches. The read and write drivers receive a nominal −13 vdc, variable from −11.5 to −18.5v; this is referenced to −2.5v, resulting in a drive voltage variable from −8 to −16v. The inhibit drivers receive a nominal −25 vdc variable from −20 to −30v; this is referenced to −6.8v resulting in a drive voltage variable from −13 to −23v.

OPERATION

A standard control panel (Figure 1-1) is installed on all memories, even though some of the indicators may not be used. The controls and indicators are primarily for maintenance; most of the lights change too rapidly to be significant when the computer is running normally. On the upper left of the panel are two toggle switches and a push button. Memory power can come on only when the POWER switch is up, but this switch is usually left on at all times so that the memory comes on with system power. The POWER light at the lower left turns on immediately when power is applied to the fast memory but is delayed by a few seconds during the power clear in the core memory. If the SINGLE STEP switch is up, the memory stops with the STOP light on at the end of every cycle; it can be restarted by pressing the RESTART button. Following a stop the memory appears to the processor to be nonexistent when the next access is requested.

Thus in order to use single step mode at the memory, the technician should also latch on the MEMORY STOP key at the processor console, thus single stepping the processor by memory subroutine calls, or turn on the DISABLE MEMORY switch so that the processor stops whenever the memory does not respond to a request for access. If malfunction causes the memory to hang up with the STOP light off, it can be restarted by pressing RESTART while holding the stop override button on. This button is located behind the double doors on the front of bay 1, on a bracket above 1E7 and 1E8. The meaning and use of the remaining controls and indicators differ somewhat for the core and fast memories and are thus described separately.

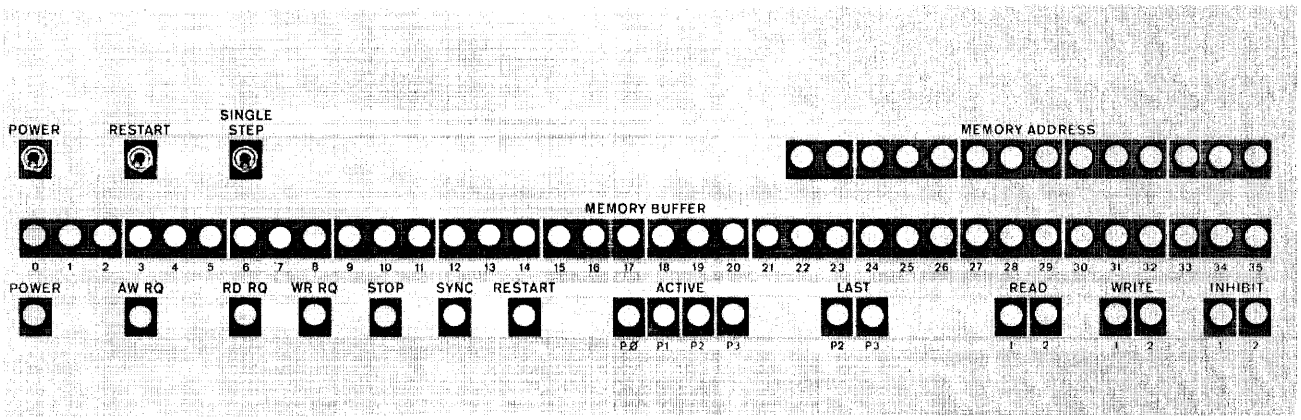


Figure 1-1 Memory Control Panel

Core Memory Controls

If the AW RQ light on a core memory remains on while the computer is running, that particular memory is not currently in use. One of the LAST lights may also stay on during operation; these lights indicate whether processor 2 or 3 was the last to have access. MEMORY ADDRESS indicates the location within the memory to which the last access was made. MEMORY BUFFER displays the word read or written, but the lights are all off following the read portion of a read-pause-write cycle. When a memory cycle starts, the AW RQ light goes off, and one of the four ACTIVE lights goes on indicating which processor gained access. The RD RQ and WR RQ lights indicate a read request, a write request, or both. At the end of the read portion of the memory cycle the SYNC light goes on, and if new data is to be written, the RESTART light goes on when the processor sends the write restart. The READ, WRITE, and INHIBIT lights (of which only the left one in each pair is used) represent the drive currents applied to the cores, and they can be on at the completion of a cycle only if a malfunction occurs.

If the memory stops independently of the processor, the ACTIVE lights are out, STOP is on, and the remaining lights reflect the cycle just performed. On an instruction stop or the stop following an examine or deposit from the processor console, the AW RQ light is on indicating that the memory is free and

awaiting a request, ACTIVE is off, and the other lights reflect the last cycle. On a memory stop at the processor (i.e., a stop following the completion of a memory subroutine) the lights reflect the type of stop. If the stop follows the read part of a read-write access, the memory is at the middle of its cycle, AW RQ is off, one of the ACTIVE lights is on, and RESTART is not yet on. For any other type of memory stop the lights are the same as for an instruction stop.

The remaining controls are located behind the double doors. The operator selects normal or interleave operation by means of a toggle switch mounted on a bracket above 1E11 and 1E12. In normal operation the switch is in the NC position, and the memory responds to a block of 16K consecutive addresses. To interleave the locations in a pair of memories the operator must switch to the NO position at both memories and make the proper jumper changes in the bus-connecting modules as described in the logic. Note that even when memories are interleaved, the MEMORY ADDRESS lights still indicate the internal location to which access was made. Memory bit 35 now corresponds to processor bit 21, and processor bit 35 determines memory selection. This exchange must be considered when working with the memories from the processor console.

At the bottom of the bay 1 plenum door are the power controls. Whenever the ac line is plugged in the red light is on. Adjacent to the light are the ac circuit breakers. Switching S1 to LOCAL turns on memory power; when the switch is in REMOTE, the memory goes on and off with system power provided the control panel POWER switch is on. S2 allows the delayed-on power to be turned on and off independently once main power is on. An elapsed time meter indicates the total time that memory power has been on.

Fast Memory Controls

On the fast memory control panel the SYNC, LAST, READ, WRITE, and INHIBIT lights are not used. A single ACTIVE light is always on indicating which processor the operator has selected for exclusive access. The processor selector is a rotary switch located to the right of the vertical mounting panel behind the front double doors. Since the fast memory contains no address register or buffer, the last four MEMORY ADDRESS lights on the right display bits 32-35 from the selected bus at all times (the other ten lights to the left are not used). Whenever any stop occurs, either in the memory or the processor, MEMORY BUFFER displays the contents of the addressed location provided all MEMORY ADDRESS lights are not off. The contents of location 0 are displayed only when the processor executes a memory stop following the read portion of a read-write access; at any other time all MEMORY BUFFER lights are on whenever location 0 is addressed. Of the remaining lights, RD RQ and WR RQ act as the address lights: they reflect the presence of read and write request levels on the bus regardless of which memory is addressed. AW RQ goes off

whenever the fast memory becomes active. The only time AW RQ remains off is after a stop following the read part of a read-write access. The RESTART light is on at any stop following a cycle in which the processor has sent a write restart to the memory.

The remaining controls are located at the bottom of the plenum door at the rear of the bay. The red light is on whenever the ac line is plugged in; beside the light are the main circuit breakers. Switching S1 to LOCAL turns on memory power; when the switch is in REMOTE, the memory goes on and off with system power provided the control panel POWER switch is on. An elapsed time meter indicates the total time memory power has been on. If the power control is a Type 829, there is another switch for the delayed output, but it is not used.

CHAPTER 2

FAST MEMORY TYPE 162

The system block diagram, Figure 2-1, shows the fast memory logic in two sections: memory control which includes the interface with the memory bus, address decoding, and timing and control logic; and the flip-flop memory bank which includes the registers with associated logic for clear, readin, and readout. The figure shows the connections between memory control and the flip-flop bank and the bus lines connecting the memory bus interface in the processor with the interface in memory control (only the lines used are shown). Each line is labeled with the function of the signal, the signal type and polarity, and the number of physical connections required to carry the information.

Except in the number of address bits, the signals used by the fast memory are identical to those for the core memory and are discussed in Chapter 1. Whenever the fast memory selection level is asserted, the fast memory responds to a 0 module address in bits 18-21. (Because there is no interleaving, bit 35 cannot be used in module selection). Memory control decodes address bits 32-35 directly from the bus and sends 16 address signals to the flip-flop bank to select an individual register. When access is requested, the time chain begins and supplies the necessary control functions to the registers. Signals sent to the flip-flop bank affect only the register selected by the cell address lines, and only that register is available to the memory bus. To read, memory control senses the output data without affecting the state of the register; to write, memory control sends a clear signal and then asserts a write signal that gates in the input data when it arrives over the bus.

The fast memory logic is shown in 14 block schematics and one flow chart; an explanation of the logic drawings and codes can be found in the Arithmetic Processor 166 Manual, F-67 (166).

ADDRESSING

Figure 2-2 shows the gates through which the processor selection level brings in the four address bits and the read and write request levels from the bus. The level for the selected processor is asserted at all times; thus some location in fast memory is always addressed even when access is being made to a core memory. Figure 2-3 is the address decoder; it translates the address into 16 signals to select a register. Each decoder output is buffered to drive two selection lines, but the decoder output for address 0 is also available directly (its use is described later). Figures 2-4 and 2-5 show the data connections to the bus. For reading, a pulse from the time chain generates a strobe that pulses the outputs of the flip-flop bank onto the selected bus; for writing, a selection level which is asserted only at the appropriate time gates data pulses from the bus into the flip-flop bank.

CONTROL LOGIC

The control connections to the bus are shown in Figure 2-6. Of the selection inputs at the left only that one chosen by the operator is asserted, and it is asserted continuously until the memory begins a cycle with the SINGLE STEP switch on. Hence, only the selected processor can gain access, and it can do so at any time unless the memory has stopped.

The left half of each bus-connecting module determines whether the memory is being addressed by decoding address bits 18-21 according to the jumper connections made in the module (ordinarily all bits are 0). A connection must also be made to the assertion of the fast memory selection signal to determine that the fast memory rather than core memory 0 is being addressed. The jumper-selected signals are ANDed with two other control inputs: one is the selection level already described; the other is the cycle request from the processor. Whenever all inputs are satisfied, the gate asserts the FMPC (fast memory processor control) processor request signal, starting the memory time chain. Control signals generated during the cycle are gated in and out of the bus through the logic at the right. Note that the selection level at the right, asserted at all times, is different from that which enables the gate at the left. The address acknowledgment and read restart pulses for the processor are triggered by pulses in the memory time chain. A write restart from the processor is gated in regardless of which memory is addressed, but unless the fast memory is active, the pulse has no effect.

Figure 2-7 shows the control for the fast memory registers. Except for location 0, the pair of address decoder outputs that is asserted enables the drivers and pulse amplifiers for the single addressed location. The read drivers are ungated, and the register (other than 0) whose address is equal to the four least significant address bits is displayed in the lights on the control panel. This does not affect the bus, however, since the levels from the registers are strobed only if the time chain is triggered. The same address levels also gate the PAs that clear a location; these are triggered by a time pulse that is generated only if information is to be written. The other decoder output is ANDed with the 1 state of the write flip-flop so that the write gate is enabled only after FMC WR has been set.

Note that the gates for location 0 differ from the others--the generation of the write gate is the same, but the read drivers are not enabled continuously, and the gate for the clear PAs is the direct output of the address decoder rather than a buffered output. The reason for this is that location 0 in fast memory is addressed whenever the memory address register in the processor is cleared prior to being loaded. To prevent the partial selection of information in location 0, the read drivers are gated both by an FMD0 level and by a flip-flop, FMC RD0, which is not set until shortly after a read request is made. Thus information from location 0 is made available only if MA bits 32-25 address that location, and not merely when MA is clear before access is requested. Since both buffered FMD outputs (read and write) are gated for location 0, the unbuffered output controls the clear PA.

REGISTERS AND BUFFERS

The final six logic drawings for this chapter, Figures 2-11 to 2-16, show the 16 fast memory registers. Each figure shows four Type 1250 modules, each of which contain three bits of eight registers plus the associated output gates. The read, clear, and write signals are applied only to the register selected by the address decoder. The sole asserted read level acts through the gates at the tops of the figures to make the outputs of a single register available at the module outputs, which drive the lights on the control panel and are available for strobing onto the bus. The selected clear pulse clears a single register, and the corresponding write level enables its input gates allowing data pulses from the bus to set the appropriate register bits.

TIMING SEQUENCE

The sequence of events that constitutes access to memory by a processor is shown in a flow chart, Figure 2-8; the logic that implements this sequence is shown in two block schematics, Figures 2-9 and 2-10.

Figure 2-9 shows the switch through which the operator selects the processor. At the right are two selection levels, one of which is asserted whenever power is on, the other when the STOP flip-flop is 0. At the top of the figure is that part of the switch that generates the selection level for writing; this level is asserted only during a fast memory access (i.e., the active flip-flop has been set) when the read request is negated. (All switch sections have rear plates through which the unselected levels are negated by shorting all unselected positions to pin 8.)

In the upper left of the flow chart appear all functions derived from bus signals that are gated only by the continuous selection level and not derived from pulses in the time chain. Two of these short sequences involve FMC RD0, the flip-flop that delays reading location 0. Any read request that appears on the bus generates a pulse that sets RD0 after a delay; whenever the request is negated (this happens at the end of every memory subroutine in the processor), RD0 returns to 0.

The third sequence depends upon level transitions in the memory logic nets. Whenever the address on the bus changes, bits 32-35 are decoded, and the asserted decoder output, provided it is not 0, asserts the read level for the addressed register. If the address is 0, the sequence continues only for a read request, in which case the read level becomes asserted when RD0 is set. Since the request is dropped after every access, location 0 cannot ordinarily be displayed.

The remaining functions in the flow chart depend upon the selection of the fast memory by the processor. Events are timed by the chain shown in Figure 2-10; control levels are supplied by the flip-flops in the lower part of Figure 2-9; signals to and from the processor are gated through the interface modules in

Figure 2-6; signals to the flip-flop bank are supplied through the drivers in Figure 2-7. The memory cycle begins (top center of the flow chart) when a processor request addresses the memory. The request signal from FMPC sets the active flip-flop (Figure 2-9:C3), whose transition triggers the time chain at the upper left in Figure 2-10. There are few delay modules in the chain; in most instances, delay between events is merely the transition across inverters and pulse amplifiers. Thus delay times shown are not settings of delay modules, but are true circuit delays (in nanoseconds unless otherwise indicated). The first pulse in the chain, FMC T0, affects two of the control flip-flops in Figure 2-9 as well as initiating the cycle. The pulse clears the restart flip-flop FMC RS and sets up the STOP flip-flop according to the SINGLE STEP switch setting.

The continuation of the chain from FMC T0 depends upon the type of access. If the read request is asserted (which is the case for either read or read-write access), the chain continues to FMC T1, which sends the read restart back to the processor and generates the read strobe. The latter pulses the bus data lines according to the outputs from the register selected by the address decoder. FMC T1 also enters a pair of delay lines to continue the chain along either of two branches depending upon whether a write request accompanies the read request. If there is none, the chain goes to FMC T4 for terminating operations. If there is a write request, the chain continues to FMC T3, which is also triggered directly from FMC T0 for write access (i.e., a write request without a read request). FMC T3 triggers the clear for the addressed location, and through a delay sets FMC WR, which asserts the gate. While this gate is open and the write selection level is asserted, any pulses arriving on the bus are gated into the addressed location. For write, WR SEL is asserted as soon as FMC ACT is set, but for read-write it does not begin until the processor negates the read request, following receipt of the read restart.

The response from the processor is shown in the rightmost flow line in the chart. FMC T0 returns the address acknowledgment to the processor immediately. For read, the processor merely accepts the acknowledgment, the data, and the read restart and there is no further communication. For write, the processor returns the data and the write restart immediately: WR SEL and the write level for the addressed location gate the data pulses from the bus into the register; the restart, gated through FMPC, sets the restart flip-flop and enters the time chain at FMC T4. For read-write, the write level remains asserted, and WR SEL becomes asserted when the processor negates the read request. The memory then remains quiescent until the processor makes the next memory call, at which time the processor returns the write restart and the new data.

Terminating Operations

Terminating operations begin with FMC T4, which is triggered by the response from the processor if there is writing, otherwise from FMC T1. FMC T4 clears the active and read-zero flip-flops; if the STOP

flip-flop has not been set, the chain continues to FMC T5, which clears FMC WR to negate the write level. If FMC STOP has been set, as in single step operation, the chain stops at FMC T4; it continues only when the operator pushes the RESTART button, triggering a RESTART signal through the pulse generator in the lower center of Figure 2-10.

Behind the double doors is a push button that overrides the stop condition for the restart, so the memory can be freed if it is hung up when FMC STOP is 0. The RESTART pulse triggers the one-shot in C3, whose output holds FMC STOP set and FMC ACT clear, so the memory cannot operate during the restart.

The termination of the clear generates a START signal that triggers FMC T5 to perform the usual cycle terminating operations and to clear FMC STOP so the memory is again available to the processor. The one-shot also supplies a clear at power turnon because it comes on in the 1 state. Connections to the power controls for memory power turnon are shown at the lower right.

CHAPTER 3

CORE MEMORY TYPE 161C

The system block diagram, Figure 3-1, shows the core memory logic in two sections: memory control includes the address and buffer registers, timing and control logic, and the interface with the memory bus; the core logic includes the drive and sense circuits for the core banks. The figure shows the bus lines connecting the memory bus interface in the processor with the interface in memory control (only the lines used are shown), and the connections between memory control and the core logic. Each line is labeled with the function of the signal, the signal type and polarity, and the number of physical connections required to carry the information.

Except in the number of address bits, the signals used by the core memory are identical to those for the fast memory and are discussed in Chapter 1. The core memory address register CMA has 14 bits, but 15 are supplied to it from the bus. For normal operation the register receives address bits 22-35, and the processor selection portion (CMPC) of memory control decodes bits 18-21 to determine when this memory is being addressed. For interleave operation bit 21 is substituted for 35 at CMA, and 35 for 21 in CMPC.

When access is requested, the time chain begins and supplies the necessary control functions to the core logic. Memory control actually governs two complete 18-bit core memories, each with independent drive and sense circuits. Memory control supplies single read, write, and inhibit signals to both sections but supplies separate strobes for the two sets of sense amplifiers. Each 18-bit core bank is composed of four 4K stacks (fields), only one of which receives drive currents during reading and writing at a given location. CMA supplies a 14-bit address to select a location within the memory. In each bank the two most significant address bits select the field, the remaining twelve bits select the X and Y windings within that field. Thus for each memory cycle only two of the eight 4K stacks are affected: one in the bank containing the left half of the addressed location; the other in the corresponding field in the bank containing the right half.

The core memory logic is shown in eight block schematics and one flow chart; an explanation of the logic drawings and codes is found in the Arithmetic Processor 166 Manual, F-67 (166).

MEMORY CONTROL

The first three logic drawings for memory control show the 14-bit address register CMA and the 36-bit data buffer CMB. Input connections from all four memory buses are shown; information is strobed in only over the bus which connects the processor that has been selected by memory control.

Figure 3-2 shows CMA and the associated read and write request flip-flops. All are cleared at the beginning of the memory cycle and then loaded from the appropriate bus by an address strobe. CMA receives address bits 22-35 for normal mode; for interleaving a switch, substitutes bit 21 for bit 35. At the same time that the address is loaded, the request flip-flops are set according to the read and write request levels on the bus.

Figures 3-3 and 3-4 show the left and right halves of CMB. The clear pulse for the register always is generated at the beginning of a cycle, and for read-write access, shortly after the information read is sent out. There are two logically equivalent gating levels for each bus. One level gates the output pulses from the core bank sense amplifiers onto the bus, the other gates pulses from the bus into CMB. Since pulses from the sense amplifiers appear on the bus for transmission to the processor, the second level gates information from the core bank into CMB in case the same information must be written back in, as well as gating in information that arrives over the bus.

Timing Sequence

The sequence of events that constitutes access to memory by a processor is shown in a flow chart, Figure 3-5; the logic that implements this sequence is shown in three block schematics, Figures 3-6 to 3-8. Figure 3-6 shows the control connections between memory and the buses. The left half of each bus-connecting module determines whether the memory is being addressed by decoding address bits 18-21 according to jumper connections made in the module. For interleaving, bit 35 may be jumpered in place of bit 21. A jumper connection must also be made to the negation of the fast memory selection signal if the core memory has the same address as a fast memory. The jumper-selected signals are ANDed with two other control inputs: one is the cycle request signal from the processor, the other is the await request signal generated within the memory. Whenever the memory completes a cycle, the await request flip-flop is set to enable these AND gates and allow the generation of the CMPC processor request signals (center of the figure) whenever a processor request for this memory arrives over the bus. The four CMPC request signals are ORed at the top left in Figure 3-8 to trigger the time chain and they are also applied to the processor selection logic in the upper left of Figure 3-7.

Four processor active flip-flops determine which processor has access for the memory cycle just beginning. During any given cycle only one active flip-flop can be set, and it generates a pair of selection levels that gate address, data, and control signals between bus and memory. If P0, the processor with highest priority, is requesting access, the CMPC P0 request level sets the P0 active flip-flop. A P1 request sets CMC P1 ACT, but only if there is no simultaneous P0 request. A similar situation occurs in the other two flip-flops, but here the priority is not based strictly on the order of the processors; if it were, the processor with lowest priority might never gain access at all. Just to the right of the active flip-flops

is the last processor flip-flop, which remembers whether P2 or P3 was the last to have access. Whenever both request access simultaneously (and neither of the top two is requesting access), the current cycle is given to that one which has been longest without access.

If there is only one CMPC request when the memory becomes free, the corresponding active flip-flop is set and all others are left clear. If there are conflicting requests, the nets below the flip-flops determine which processor has priority. Every processor currently requesting access attempts to set the corresponding active flip-flop by grounding its unbuffered 1 output. But the highest priority flip-flop that is now set grounds the 0 sides of all those having lesser priority through the nets below. Thus for any lower priority request both flip-flops outputs are grounded. The situation becomes stabilized when the first pulse in the time chain clears the await request flip-flop, negating all processor request signals, so that the indeterminate flip-flops become clear. Thus the only flip-flop set is the one corresponding to the highest priority processor that has requested access. All those of lower priority have been cleared. The flip-flop that remains 1 generates a pair of selection levels that connect CMB to the selected bus and gate the control signals shown at the right in Figure 3-6. Thus the address acknowledgment and the read restart are sent back to the selected processor, the write restart from that processor is received in the memory, and the address strobe generated by the time chain loads CMA from the selected bus.

The remaining flip-flops at the right in Figure 3-7 supply gating levels for the drivers in the core logic and control functions for the time chains. In the upper right are three flip-flops that control the read, inhibit, and write drivers. The four flip-flops in the lower right include the await request flip-flop which indicates when the memory is free, a flip-flop that receives the write restart signal from the processor, another that syncs the write portion of the memory cycle, and a STOP flip-flop used for single stepping.

The time chain with associated logic is shown in Figure 3-8. Any CMPC processor request signal triggers the chain by generating CMC T0. This pulse clears CMA and the read and write request flip-flops directly, generates a clear for CMB at the left in Figure 3-8, and clears the flip-flops at the lower right in Figure 3-7. In particular it clears CMC AW RQ, allowing the active flip-flops to stabilize and thus select the processor that has priority. Then CMC T1, gated by the selection level at the right in Figure 3-6, generates the address strobe to load CMA and sends the address acknowledgment to the processor. If the access is for write, the processor immediately loads CMB and returns the write restart; this generates the restart strobe in CMPC to set CMC RS at the right in Figure 3-7. Since the memory now has the information it will need for writing during the second part of the cycle, the flip-flop transition triggers the state clear (Figure 3-8:B8), which clears the active flip-flops disconnecting the memory from the selected processor.

The next pulse in the chain sets up the last processor flip-flop and turns on the read driver in the core logic by setting CMC RD (Figure 3-7:B5). From this point the time chain continues at the top of Figure 3-8,

but if information is to be returned to the processor a separate branch (B3) triggers a strobe for the sense amplifiers at the proper time during the read level (there are separate strobes for the left and right core banks). Slightly after information is strobed into CMB and out over the bus, the left strobe triggers the read restart (C2) for the processor via CMPC, and if new information is to be written, the right strobe triggers the clear signal for CMB (B2). The gating condition for this clear is the 1 state of the write request flip-flop, but this indicates read-write access, because the strobe itself is generated only if CMA RD RQ is 1. In a cycle initiated for write access there is no strobe of the information sensed from core; whereas for read-write CMB is cleared after the outgoing information has been sent over the bus.

In read access, the condition CMA WR RQ(0) allows the right strobe to trigger the state clear, which disconnects the bus. This same condition also allows the final pulse in the read part of the cycle (CMC T5, which also ends the read level by clearing CMC RD) to trigger the chain for the write part (A5) in order to write the word just read back into core. If new information is to be written, CMC T5 instead sets CMC PSE SYNC. Since for write, the read portion only clears a location and the processor returns the write restart immediately upon receiving the address acknowledgment, CMC RS is already 1 and the transition in the sync flip-flop immediately triggers CMC T6 to start the write portion of the cycle. For read-write, the memory has sent information back to the porcessor, and it actually pauses after CMC T5 to wait for new information. When the information is available, the processor sends it and the write restart that triggers the restart strobe in memory. The strobe sets CMC RS whose transition generates the state clear and, since the sync is 1, triggers CMC T6 to begin writing.

Terminating Operations

In the second part of the cycle the word now in CMB, whether supplied by the processor or read from memory during the first part, is written in the addressed location. CMC T7 starts the inhibit level by setting CMC INH, and it also sets the STOP flip-flop if the single step switch is on (Figure 3-7, lower right). The next pulse, CMC T8, begins the write level by setting CMC WR, and core driving is terminated at CMC T12, which clears all of the core control flip-flops.

This is the end of the chain in so far as it affects the core logic, but a separate branch (Figure 3-8:B5), starting at CMC T9A, handles termination of memory control operation. If the STOP flip-flop has not been set, this branch goes to CMC T10, which triggers the state clear, and then CMC T11, which sets CMC AW RQ so that memory is again available for processor requests. If CMC STOP is 1, as in single step operation, the chain stops. It continues only when the operator pushes the RESTART button, triggering a restart signal through the pulse generator in C5. Behind the double doors is a pushbutton that overrides the stop condition for the restart so that memory can be freed if it is hung up when CMC STOP is 0.

Memory Power Turnon

The remaining logic in Figure 3-8 is associated with memory power turnon. When power comes on, the 4303 Integrating Delay in the lower left comes on in the 1 state and supplies a power clear enable of approximately 4 sec. During this time the clock just to the right generates power clear pulses, which trigger a second 4303 and also trigger CMC T12 (upper right) to clear the flip-flops that control the core drivers. During the power clear and for 15 msec thereafter, the second 4303 supplies a power turnon level that holds CMC AW RQ in the 0 state to prevent the memory from responding to any processor requests. The return to 0 of the power turnon delay generates a power start, which triggers the terminating part of the time chain at CMC T10. Connections to the power controls for memory power turnon are shown in the lower left of Figure 3-7.

CORE LOGIC

The logic associated with the core banks is in two logic drawings, Figure 3-9 and 3-10, which together show a single 18-bit core bank including core selection, read and write drivers, inhibit drivers, and sense amplifiers. All modules shown are in bay 2, and module locations are given only by panel letter and slot number. In each case the first letter indicates the panel for the upper (left) core bank; the letter following in parentheses is the location of the equivalent module in the lower (right) bank. Wherever an input or output differs for the two banks, both signals are shown, the upper being for the left half of the memory, the lower for the right. On all lines labeled by bit number, the bits for both banks are listed.

In the upper right of Figure 3-9 are the control signals generated by memory control for the core logic. Memory control generates separate left and right strobes, each of which triggers an internal strobe in the corresponding core logic. The 1 states of the core control flip-flops are buffered to develop the inhibit, read, and write levels for the core drivers.

The inhibit level and the strobe pulse are used by the inhibit and sense portions of the logic, shown in the left and right halves of Figure 3-10. The read and write levels turn on the read and write drivers shown with the 781 Power Supply at the left in Figure 3-9. The drivers use the RW outputs of the supply. The read level turns on the top driver for the read bus; the write level turns on the lower two drivers. There are separate drivers for the X and Y write buses. The INH outputs of the 781 Supply go to the inhibit drivers at the left in Figure 3-10.

Addressing

Figure 3-9 shows the logic that selects the single addressed core register in the 16K core memory. All internal addressing is done in terms of a 4K memory; i.e., in each memory cycle the drivers affect only

one 4K stack selected from the 16K bank by the two most significant bits of the 14-bit address. The field selection logic, shown in the bottom center of the figure, decodes CMA bits 22 and 23 to select a single stack. The output levels go to pin N of the read-write switches, enabling only those associated with the selected field.

Read-Write Switches

The remainder of the logic in Figure 3-9 implements the selection of one location within the selected field. For the X windings five sets of read-write switches select a single winding according to CMA30-35. Four of these decode the three more significant bits and are shown in A1, C1, A7, and C7 beside the large blocks representing the core stacks seen end on; the fifth, in A4, decodes the other three bits. The first four sets are connected to the read bus, but only one set is enabled by the field selection logic. The fifth set is connected to the write X bus, and is permanently enabled. When the read level is asserted at pin J, the read driver connects the read bus to the (nominally) -13v read-write power; when READ becomes ground, the driver holds the bus steady at the -2.5v memory power return. The write level produces the same effect through the write X driver. When WRITE is asserted, the bus is connected to -13v; when ground, the driver holds the bus at the -2.5v return.

All read-write switches are identical, with every configuration of CMA bit inputs selecting a pair of output lines, one of which is a current source, the other a current sink. The multiple connections for some CMA inputs are required by the internal circuitry; the circuit operates on two CMA bits similar to a binary to quaternary decoder, but requiring two enabling levels: one is the third CMA bit; the other is the field selection level. The selection levels are not used on the fifth set of switches in A4: when an input is floating, it acts as if asserted.

The switches beside the large blocks have source and sink outputs paralleled by external wiring. For example, in A1 the 1987s have pins Y and Z jumpered together. These switches thus act as simple line drivers: they connect a selected output directly to the read bus regardless of the direction of current flow. The switches in A4 however are not so connected; the source and sink outputs separately drive all four stacks through diodes on the Type 1020 modules (shown by dashed lines beside the stacks). Each configuration of CMA bits selects a pair of these lines such as those labeled X0, which come from pins Z and Y of the left switch in A4. The X0 pair is connected to all four stacks, but current flows into and out of only the selected field because an unselected switch keeps its output floating whether it is "unselected" by an incorrect address or by a negated field selection level.

System Operation

The selected set of switches that receive CMA30-32 selects a group of eight X windings out of the eight available groups on one side of the stack. The switches in A4 choose the single winding out of the selected group of eight through which current is to flow (the actual selection is performed in the diode units, one of which is associated with each stack). A source output, e.g., pin Z of the left switch in A4, supplies current to eight diodes that conduct into the stack and are connected to windings 00, 10,..., 70. Eight more diodes connected to the same windings conduct out of the stack and are bused together to current sink pin Y of the same switch. To select address 00, for example, the upper switch in A1 connects its jumpered Y and Z pins to the read bus, thus connecting windings 0-7 to the read bus through the resistors in the 1976. The switch in A4 connects its outputs Y and Z to the write bus, but Y acts only as a current sink and Z acts only as a current source.

During the read portion of the cycle, current flows from B10A through the only forward-biased diode (the one for 00), through the stack, through pins Z and E of the resistor module, and into B2Y. Current flow stops when the read bus returns to -2.5v; i.e., when the read level is negated. During the write portion of the cycle, the read bus is held at -2.5v while the write bus falls to the nominal -13v. Now B10Y functions as a common sink, but the only possible source is B2Z. Hence current flows from the latter pin, through the resistor module, through the stack, through the other of the two 00 diodes, and into B10Y.

The selection of a Y winding is done in exactly the same way except that there is a separate write Y driver and bus. The four sets of Y switches that receive the field selection levels are shown above and below the blocks that represent the fields; the fifth set is in B7.

Inhibit

The left half of Figure 3-10 shows the inhibit logic. The inhibit level from memory control, the inhibit power (variable from -20 to -30v), and levels from CMB come in at the left. Each 1992 contains six drivers that receive inhibit power over separate lines via the resistors in the 1994. The assertion of the inhibit level turns on all drivers that correspond to 0 bits in CMB; the output of each driver is connected to the corresponding plane in all four fields. However, current flows through the inhibit windings in only the single stack that is selected according to CMA bits 22 and 23 by the 1991's at the right. Between the stack and the field select modules are four diode boards, one for each stack, and a single resistor board. The diodes merely prevent any current from flowing in the wrong direction; the resistor board contains eight voltage dividers that hold the unselected field select outputs at about -30v. Current flows from the selected outputs of the field select module through the diode board and the stack into the inhibit drivers that are turned on. Depending on stack temperature, a selected inhibit driver draws up to 200 ma at a voltage between 0 and -6v.

Sense

The sense amplifiers for the core banks are in the right half of Figure 3-10. Selection of a field is performed within the sense amplifiers modules: each amplifier has four preamps, each of which is connected to the sense winding for the corresponding plane in a single stack (the single pair of lines coming from each stack actually represents 18 pairs). Selection of a stack, i.e., a particular preamp of the four on each module, is made according to CMA bits 22 and 23, applied to pins T and U of the amplifiers. In the drawing field 0 is shown selected; the 0 states of bits 22 and 23 asserted negative at T and U select the preamps for field 0.

While read current is supplied to a stack, a signal appears on the sense winding for each plane in which a core of the selected register switches from 1 to 0. If information is to be sent over the memory bus, the strobe derived from the memory time chain is applied to pin W of the sense amplifiers at the appropriate time during the read interval. The strobe produces an output pulse at pin Z of each amplifier whose winding has sensed a core transition. The output pulses are gated onto the appropriate bus and are also gated into CMB in case the information read must be rewritten.

CHAPTER 4

CIRCUITS

All logic circuits, i.e., those that accept and generate standard DEC logic levels and pulses, are discussed in PDP-6 Circuits. Described here are the eight special circuits that provide drive, power, and sense capabilities for the core memory. Interleaved with the text are block diagrams of all but the power circuits; schematics are at the back of the manual and reference to them is implied. The core logic includes a number of modules containing passive components that perform various switching and current regulating functions; schematics for these are included but the circuits are not described. The circuit descriptions will be more understandable to the reader who has a secure understanding of the memory control and addressing logic (Chapter 3).

DRIVE CIRCUITS

The drive circuits include Memory Driver 1989, which supplies timed current pulses to the address selection circuits; Read-Write Switch 1987, which selects the stack X or Y winding through which the current pulse passes; Inhibit Driver 1992, which selects the planes that receive inhibit current; and Inhibit Field Select 1991, which chooses one of the four inhibit windings associated with each plane.

Memory Driver 1989

Three Type 1989's supply read, write X, and write Y current pulses. When input J is negative, the driver connects output V to the nominal -13v memory drive voltage; when J is grounded, the circuit returns V to the -2.5v clamp level. The -13v compensated memory power enters the module at X. An external 60-ohm resistor from M to -35v at T forward biases D8 and D9 to create an internal -14.5v supply. Since this supply is always two diode drops (1.5v) more negative than the -13v RW supply, the -14.5v level also varies with core stack temperature. The Q3 collector is clamped at 0.3v below the negative RW supply; consequently the bases of output transistors Q5, Q7, Q9, and Q11 are at the negative RW supply level when turned on. The more negative collector supply for these transistors prevents saturation. On every module the external resistor is in series with internal diodes D8-D9. These three resistance-diode networks are in parallel and together form the series resistance for the RW regulator in the Type 781 Memory Power Supply.

When input J is ground, R2 holds Q2 cut off, allowing R3 to turn on Q4 and Q1. The Q4 collector and the direct-coupled Q3 emitter base junction are therefore close to ground potential; Q5 to Q11 odd are consequently cut off, so the -13v power is disconnected from the output. Since Q1 is on, its emitter draws

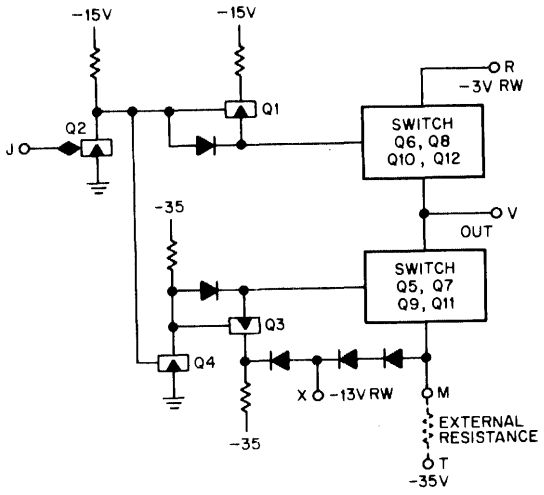


Figure 4-1 Memory Driver Type 1989

current through the bases of Q6 to Q12 even, which are thus saturated, connecting the output to the -2.5v RW return. When J becomes negative, Q2 turns on to cut off Q4 and Q1, and thus cut off Q6 to Q12 even. With Q4 cut off, R9 turns on Q3 which turns on Q5 to Q11 odd. The output then falls to the compensated memory power level.

Read-Write Switch 1987

The four circuits on this module connect X or Y windings from the stack to the read or write bus. When the bus (input S) is quiescent at -3v, the selected switch supplies drive current to the stack windings; when the bus is active at the nominal -13v, the switch draws current from the stack. Throughout a memory cycle a single pair of outputs is selected by ground levels at the corresponding 4-diode AND gate input. In most applications, F is jumpered to H, and K to L; pin F then selects one pair, whereas L selects the other. At one side of the stack, the two outputs of a switch are jumpered together so that the switch selects a winding for both read and write; at the other side they are used separately with additional diode selection at the input to the stack.

Consider the switch at the upper left on the schematic, with inputs E, F, M and N, outputs Y and Z. When any input is negative, Q1 conducts and its collector is about 1v below ground (D19 and D20 reduce the level of saturation in Q1). The Q1 collector drives the base of Q2 directly and is coupled through D23 and D24 to the bases of output transistors Q3 and Q4. Since the stack windings at pins Y and Z are never more positive than -2.5v, Q2 to Q4 are all cut off. The windings thus float at whatever negative voltage is applied at the other end, which may be between -2.5 and -20v, depending upon the state of the switch at the other side of the stack. Since this switch is unselected, however, no current can flow through the windings regardless of the voltage at the switch output.

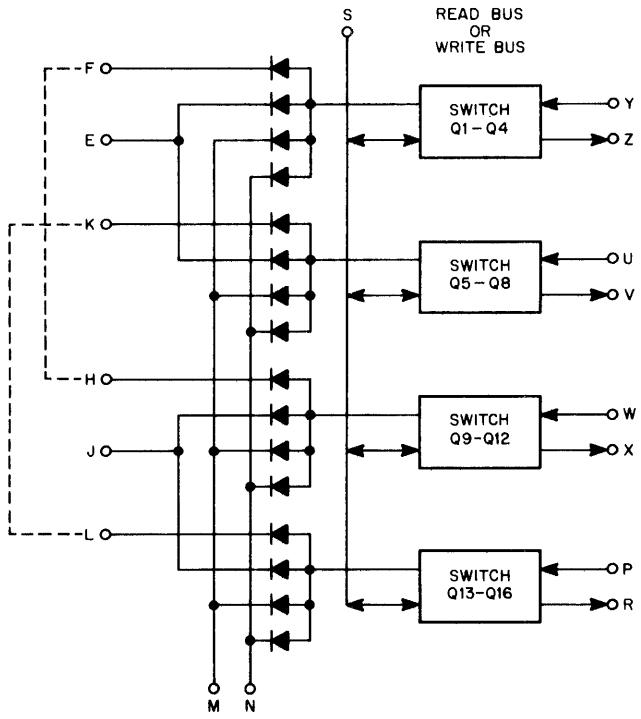


Figure 4-2 Read-Write Switch Type 1987

When all four inputs are ground selecting the switch, Q1 is cut off by R2 and the drop across D20, allowing R3 to act as a current sink for the Q2 base. Q2 thus saturates, and its emitter accepts enough current to saturate both Q3 and Q4. The switches at one end of the windings are fed by the read bus; those at the other end, by the write bus. At the time of switch selection, both buses are quiescent at -2.5v, and no current flows through the stack. However, base current is established through Q3 and Q4, so that current may flow whenever one of the buses becomes negative. During the read portion of the cycle, the read bus drops to the nominal -13v compensated memory power level, while the write bus is held through low impedance to -2.5v. On the "write" side of the stack, current flows from the -2.5v bus through Q3 of one module and into the stack from output Z. On the "read" side, current flows out of the stack through Q4 of another module, onto the negative bus. When the read bus returns to -2.5v, current flow ceases. During the write portion of the cycle, the write bus drops to -13v while the read bus is held at -2.5v; current then flows through Q3 on the "read" side through the stack and through Q4 on the "write" side onto the negative write bus.

Inhibit Driver 1992

A negative level at input E enables all six circuits on this module: individual drivers are turned on by negative levels at separate pins. Each circuit is a gated current sink controlled by an AND gate. Six external 110-ohm resistors (on Inhibit Driver Resistors 1994) route inhibit power separately to each circuit.

When a driver is turned on, the current sink emitter clamps the supply resistor at approximately -6.3v; since the inhibit current from the collector is almost precisely that which flows through the resistor, current amplitude is varied to suit stack temperature conditions by varying the inhibit supply voltage (nominally -25v) at the other end of the resistor.

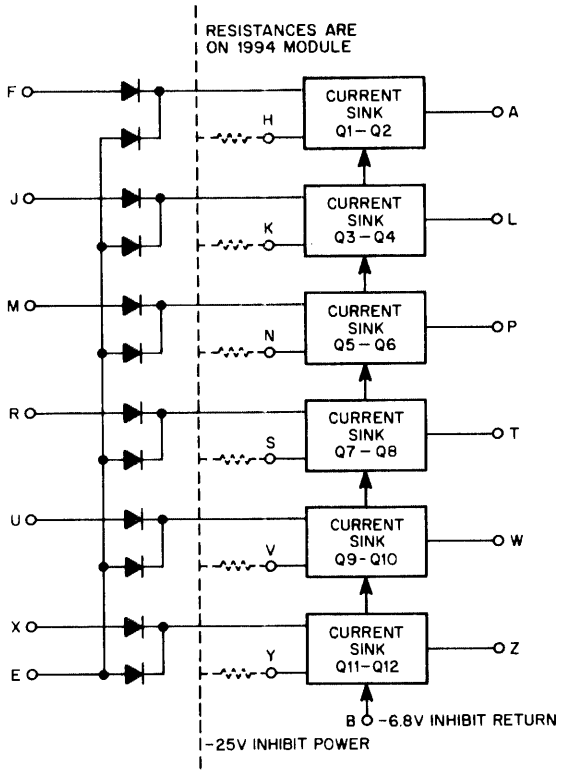


Figure 4-3 Inhibit Driver Type 1992

Consider the circuit at the left on the schematic, with inputs E and F, output A. This circuit is quiescent with ground levels at one or both inputs to AND gate D1-D2. The Q1 emitter is held at -1v by divider R2-R4; consequently Q1 is off and its collector is about -25v; clampled there by D4 and the external inhibit power supply resistance at H. The drop across D4 is sufficient to ensure that Q2 is also off; consequently Q2 collector current into A is limited to leakage. Negative levels at E and F allow R1 to saturate Q1, which provides base current for Q2 at approximately -6.5v, held by the D3 clamp to the -6.8v inhibit power return. Since the Q2 emitter is returned through the external supply resistance to the -25v inhibit power, conduction is established in Q2 at an amplitude governed by the inhibit power voltage and the external resistance value. Since Q2 operates in the common-base mode, collector current is very nearly independent of collector voltage, as long as the collector remains reasonably far from saturation, i.e., remains within the range from ground to.-6v.

Inhibit Field Select 1991

This module contains four switches that operate in pairs governed by negative AND gates. A negative input at L, common to both gates, enables the entire module. Pins J and K then turn on one pair of switches; pins N and P the other. When turned on, each switch maintains its output at -2.0v against output currents up to 2 amp.

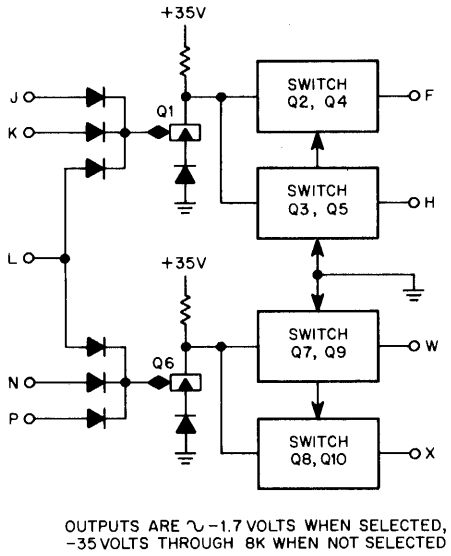


Figure 4-4 Inhibit Field Select Type 1991

The left circuit on the schematic is quiescent when any input J, K, or L, is at ground. The Q1 emitter is held at -0.7v by D4 and R2, so Q1 is off. R3 holds the four output transistors off, so both outputs are returned to -35v through 8K (the series combination of R3 with R4 or R7). The outputs are held at approximately -30v by an external voltage divider of much lower impedance (on the Inhibit Field Selector Resistor 1993). When all three inputs are negative, R1 saturates Q1, bringing the bases of Q2 and Q3 up to approximately -1v. Whenever, for example, the pin H output becomes more negative than -1v less the Q3 emitter base drop (i.e., about -1.8v), Q3 conducts drawing base current through Q5, which in turn supplies sufficient collector current to maintain the output at approximately 1.8v. Q3 and Q5 conduct only to the extent necessary to maintain the output close to ground; when the current demand is slight, conduction through Q3 diminishes lowering the base current through Q5 which therefore conducts less. Since the Q5 collector is always more negative than its base due to the Q3 emitter-collector saturation drop, Q5 is never saturated.

The L2-R9 network partially isolates the emitter of Q3 from fast transients in the output current at H. When current is suddenly drawn from H, the voltage drop is initially coupled to the Q3 emitter only by R9;

the coupling increases as the L2 field builds up. If the network were absent, the series combination of Q1 and the Q3 base-emitter junction would have to supply a heavy current surge during the Q5 turnon delay. The effect at H of the L2-R9 time constant is reduced by the forward current gain of Q5.

SENSE CIRCUITS

Associated with the 4-input DC Sense Amplifier 4552 is the Master Slice Control 4527, which supplies dc reference and bias levels to 18 sense amplifiers in parallel.

Sense Amplifier 4552

This module has four dc-coupled gated preamplifiers fed in common by an adjustable current source, a decoding network to select the active preamp, a rectifying slicer that generates a ground level in response to satisfactory sense winding input, and a pulse amplifier gated by the slicer output.

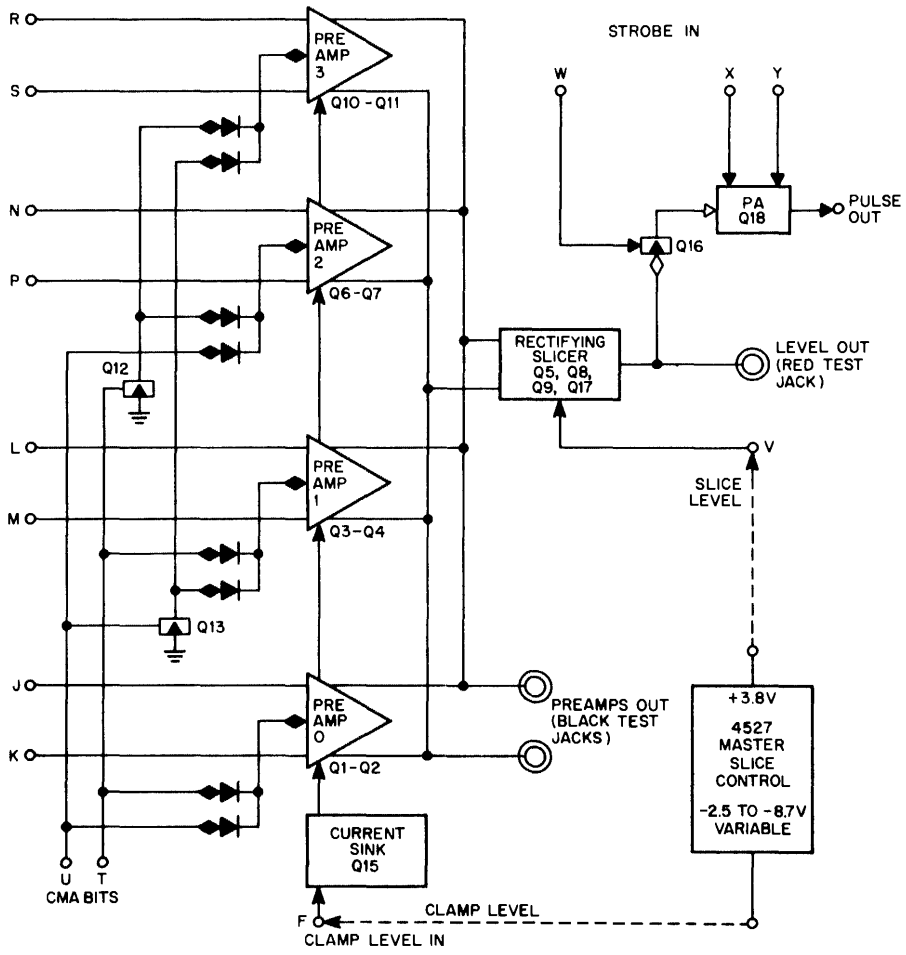


Figure 4-5 Sense Amplifier Type 4552

The two inverters, Q12 and Q13, and the four negative diode AND gates are at the bottom of the schematic. Pins T and U receive memory address 1 levels asserted at ground; consequently field 0 (the preamp at the far left on the schematic) is selected when both are negative. The negative level at the common cathode connection of D1 and D2 allows R5 (returned to -15v) to turn on Q1 through D3. Q1 saturates, and its collector rises to ground, bringing the sense winding (pins R and S) as well as the Q2 bases close to ground level through the resistance network R7-R10-R11. The Q2 emitters follow this positive level transition, and Q2 thus provides the entire 10-ma current through current sink Q15 (far right of the schematic). The remaining three preamp stages are cut off because the Q15 collector, connected to the taps of the four balance adjustments R12, R22, R33, and R44, is held by Q2 at about -0.5v, and the bases of the three unselected preamp difference amplifiers, Q4, Q7, and Q11, are clamped (e.g., by D11) at approximately -3.5v. Thus Q2 is the only preamp that functions when field 0 is selected. Absence of logic signal connections at T and U results in selection of field 0, since R5 draws enough current through D1 and D2 to saturate Q12 and Q13. The differential gain from pins R-S to the Q2 collectors is approximately 40. The Daystrom Transitrim resistors are adjusted at the factory for optimum preamp balance; there is no way to calibrate them properly in the field, and their adjustments are thus purposefully unavailable when the module is in the mounting panel.

The current through the selected preamp is regulated by the voltage at pin F (far right); making this voltage more positive causes an increased drop across R60, resulting in increased current through current sink Q15. The voltage at F is normally between -6.8 and -7.2v, causing a nominal 10-ma bias current. In the absence of signal at the selected input, this current is split equally through R3 and R4 (top left); thus the preamp collector voltage is nominally +4v. Resistors R1 and R2 allow marginal checking the preamps by varying the +10 A line. The +10 E line, to which R3 and R4 are returned, is a fixed voltage from the memory power supply and is not available for marginal checking.

The slice level, approximately +3.8v, is supplied to emitter follower Q9 at V. In the absence of an input signal, approximately 6 ma flows through R30 and Q9, so the Q9 collector is about +0.3v, clamped by D10. This slightly positive level cuts off Q17, whose collector is therefore about -3.2v, held by voltage divider R51-R56. Since the Q16 emitter is at the same voltage, negative pulses at the strobe input (W) cannot turn on Q16, so there is no pulse output at Z without a sense signal input to the selected preamp.

A sense signal causes an increased drop across R3 or R4 depending on the sense signal polarity. If the drop across R4 increases, Q5 turns on; an increased drop across R3 turns on Q8. Thus an input causes either Q5 or Q8 to draw current through R30, borrowing current from the Q9 emitter. When conduction through Q9 is less than 1.2 ma, the D10 clamp breaks allowing R29 to forward bias Q17, grounding the Q16 emitter. Ideally, conduction through Q9 should fall to less than 0.8 ma in order to guarantee enough base current for Q17 to carry the strobe pulse load.

Voltage divider R57-R63 (right) clamps R54 at about -8v through D24. This -8v supply is coupled to the Q16 collector through the T2 primary. When Q17 is saturated, the Q16 emitter is close to ground, and the negative strobe turns on Q16. The T2 secondary immediately supplies a negative pulse through D27 to the Q18 base. Q18 saturates, providing a negative pulse at output Z through T1. When the current in the T2 primary reaches 3 ma, the D24 clamp breaks, and the voltage across the primary rapidly decays. The T2 secondary voltage falls, and R59 cuts off Q18 ending the output pulse. The resistance-diode networks R55-D26 and R58-D25 suppress reverse voltage transients in T1 and T2. Pulse duration is governed by the T2 primary inductance in conjunction with the D24 clamp voltage. D27 acts as an isolating diode, allowing negative pulses at either Y or X to generate a negative pulse at Z by saturating Q18 as described above; these inputs are not gated by the presence of sense signal.

There are three test jacks on the aluminum module frame. The two black jacks supply the common collector connections of the four gated preamplifiers; the jacks are dc coupled, allowing observation of both the collector bias voltage and collector variations in response to sense inputs. The red jack supplies the logic level output of the slicer; this level is quiescent at approximately -3.2v, and rises to ground in the presence of adequate signal input at the preamp.

Master Slice Control 4527

This module contains a number of passive components that act as a voltage divider to provide two dc reference levels for the sense amplifiers. The clamp level at pin F is variable by adjusting the R1 tap. R1 is paralleled by D7, so the voltage adjustment range is 6.2v; the lower end of R1 is held at -8.7v by diode string D1 to D6, so the clamp level range is -2.5 to -8.7v. D10-D11 holds the cathode of D9 at +3.1v, and the slice level output at V is more positive than this by the drop across D9; consequently the slice level is approximately +3.8v. The clamp level is purposefully dependent on the -35v raw memory supply, because it is to this same point that the emitter of current sink Q15 in the sense amplifier module is returned. Similarly the slicer emitters in the sense amplifier are returned to the +10 E memory supply line, so the slice level developed by this module is made dependent on that supply voltage.

POWER CIRCUITS

The special power circuits are the Memory Power Supply 781 and the Type 1711 Power Supply Control module contained within the supply. The standard DEC logic power supplies and controls are described in PDP-6 Circuits.

Memory Power Supply 781

This supply consists of a -35 vdc power source, a series regulator for inhibit power, a shunt regulator for read-write power, and a 1711 module that contains circuits governing the regulator control circuits. Forward-biased diodes D5-D6-D7, in the ground leg of the read-write shunt regulator, provide a -2.5v read-write return. The series combination of R1 and zener diode D8 provides the -6.8v inhibit return. The shunt regulator that produces the nominal -13v RW power consists of Q1 and Q4, the latter is the shunt transistor. The Q4 base is returned through R2 to +10v; consequently whenever conduction through Q1 diminishes, the conduction through Q4 also lowers, allowing the RW level to become more negative. Current through Q1 is regulated by varying its base current at pin F of the control module. The series impedance from -35v to the Q4 collector consists of three 60-ohm resistors in parallel across pins M and T of the memory driver modules. Zener diode D9 clamps the RW output at -17v in case Q4 opens.

In the series regulator that supplies the nominal -25v for inhibit power, Q3, the pass transistor, acts as a high-power emitter follower. When the control module draws increased current at N, Q2 increases conduction, causing a greater drop across R3. The Q2 collector thus becomes less negative, and the decreased level is coupled to the output through Q3. R6 and R7 provide a minimum load on Q3 to prevent output voltage increase due to insufficient Q3 cutoff in the absence of inhibit load.

Power Supply Control 1711

This module contains two control circuits for the inhibit and read-write supplies. Each consists of a difference amplifier and a single-ended output stage. In either, drawing more current through the output stage (Q3, Q6) makes the power supply output less negative. In the RW control (bottom of the schematic), the difference amplifier is Q4-Q5. The Q4 base receives a constant (but adjustable) level from a voltage divider across a double-anode reference diode. The Q5 base is fed by a voltage divider composed of R16, R17 and an external thermistor across pins J and K. The bottom end of the divider receives the negative RW power at H; the top end is returned to the -2.5 RW return at E.

Assume constant stack temperature: If the RW voltage becomes more positive, the change is coupled through the thermistor and R17 to the Q5 base. Q5 conducts less and Q4 takes a larger proportion of the current through R14. The Q5 collector thus supplies less current to the Q6 emitter-base junction; consequently the Q6 current decreases and the power supply responds by making the RW output more negative. If it becomes too negative, conduction through Q5 and Q6 increases, causing the supply to make it more positive.

Now assume a constant load: As the stack temperature increases the thermistor resistance across J and K becomes lower. This has the effect of increasing the coupling between the negative RW output and the

Q5 base, so conduction increases in Q5 and Q6, making the power supply output less negative. Similarly, as the stack cools off, conduction in Q5 and Q6 decreases and the supply output becomes more negative. The temperature coefficient is normally 0.8% per °C. An additional divider network, R18-R19, may be introduced by jumpering V and W; the Q5 base is then coupled to the RW output at lower impedance, so the thermistor has less effect on the circuit. This lowers the coefficient to 0.5% per °C.

CHAPTER 5

MAINTENANCE

Preventive and corrective maintenance at the system level are discussed here, but information is included for the circuits described in Chapter 4; for a discussion of troubleshooting and repair of other modules refer to PDP-6 Circuits. Since no memory can operate independently of a processor, the material herein must be used in conjunction with that in Chapter 9 of the central processor manual. In fact many memory maintenance procedures involve use of the processor, which, for purposes of illustration, is assumed to be the Arithmetic Processor Type 166 (section references preceded by 'AP' are to the manual for the 166). Keys and switches on other processors are usually identical; for equivalent pin connections refer to the appropriate logic drawings.

Engineering drawings for the memories are listed on the MDLs (for information about the different types of drawings refer to Appendix 1 in the processor manual). Spares lists are provided by DEC Field Services. The following test equipment is required, in addition to that listed in Chapter 9 of the processor manual (suggested commercial brands are given for purposes of specification and do not constitute exclusive endorsement):

Oscilloscope current probe	Hewlett Packard 1110A, calibrated to 1 mv per ma
Dual-channel oscilloscope preamplifier	Tektronix Model CA or equivalent; must have algebraic-add facilities

PREVENTIVE MAINTENANCE

Preventive maintenance consists of performing specific procedures at scheduled times, using maintenance programs, marginal checks, and other electrical and mechanical checks, including cleaning and inspection.

Maintenance Programs

There are eight Maindec programs for the memory system. All use the processor to exercise either fast or core memory facilitating check-out or diagnosis of malfunctions. Each consists of a program tape and a write-up in the 6-MD-DEC reference manual (an introduction to Maindec writeups can be found in Section 9.2 of the processor manual). Maindecs for the memories are as follows:

Maindec 602 Mircocheckerboard - A 16-word rapid checkerboard exercise for core memory. Installations having both fast and core memories may use the program most efficiently by loading it into the fast memory starting at address 000 000. The program is then immune from malfunctions in core, allowing considerable latitude in troubleshooting.

Maindec 603 High-End Address Test - A unique-contents test for core memory in installations without a fast memory. The program, which is loaded into the top of memory (the highest available addresses), checks each location for unique contents by loading it with its own address.

Maindec 605 Memory Speed Test - Generates a one or two location indirect addressing loop anywhere in memory, except locations 10-177 which contain the program. Locations are chosen at the DATA switches. After the processor enters the loop, memory speed is checked with an oscilloscope at pin 1M09Z of the processor.

Maindec 606 Power Transient Retention Test - Checks that every memory cycle deposits correct information even with power interruption (the memory must perform a complete cycle with accurate readout and writein or else perform no cycle at all).

Maindec 607 Memory Overlap - (Information not yet available.)

Maindec 613 Core Data Test 1 - Tests all locations from 40 up except the top 27, which are occupied by the program. Each time the test word, originally supplied by the DATA switches, is read in and out, the contents of core are tested for accuracy; then the test word is rotated once and the same location is tested again. After 36 rotations the address is indexed and the next location is tested in the same manner. Installations having two core memories can use this program most efficiently by modifying its address limits to hold the program in one memory while testing the other completely. The program is then immune to marginal checking or troubleshooting procedures on the memory under test.

Maindec 622 Four Memory Checkerboard Programs - Two check a 4K memory from the high and low ends; the other two check a 16K memory with and without interleaving. These programs are more versatile than Maindec 602 in that each allows selection among four different checkerboard patterns, and allows exclusion of selected planes or selected addresses from the test. Except for the 16K interleave checkerboard, installations having more than one core memory use these programs most efficiently by modifying them so they reside in one memory while testing the other.

Maindec 662 Fast Memory Test - Tests all 16 locations in fast memory as accumulators, as memory registers, and except for AC0, as index registers. The program resides in core and thus cannot be affected by marginal checking the fast memory.

Always consult the program write-ups for information on start addresses, address modification, or details of program operation. For example, the start addresses for the eight tests in Maindec 662 and the addresses of two locations governing optional program performance on error and Teletype identification of the current test are given in page 1 of the write-up. However, every test may be started at its own first instruction; test 1, for example, begins in location 1040 (see page 5 of the write-up).

As another example consider tests 5 and 6 of this program. Part 5 uses the contents of the DATA switches as information and tests the fast memory locations consecutively; part 6 is the same but tests only one location repeatedly. When going from 5 to 6, the particular location tested will be that which was currently under test in part 5. The operator can, however, select a particular accumulator without single stepping through part 5 until it becomes current. Page 9 shows that part 5 loads a fast memory register by executing the instruction in the location specified by the contents of location 115, then tests it by executing the instruction in the location specified by the contents of location 140. If part 5 has been running, locations 140 and 150 point, respectively, to one of the arithmetic compare instructions in locations 100-117 and one of the move instructions in locations 120-137. Page 3 indicates 0's in 140 and 150, because they are loaded during part 5. The user may select fast memory address part 6 by adding 100_8 to it, resulting in a number from 100 to 117, then deposit this in 140. He adds the same address to 120_8 and deposits this in 150. Now he starts part 6 by setting the ADDRESS switches to 001261 and pressing START.

Almost all large test programs consist of a number of small routines which lend themselves to analysis of the type exemplified above. Such analysis generally saves considerable time over the alternatives of single stepping or the synthesis of a new program.

PM Schedules

As recommended in AP 9.3a, PM tasks for the entire system should be spread out over the scheduled interval for each task, with a short period of time set aside each day exclusively for PM procedures. The following schedules, which pertain only to the Types 161C and 162, should accordingly be dovetailed with the recommendations in AP 9.3b.

Unlike the processor, a core memory requires a short "warm up" time. The memory is functional immediately after turnon, but the core stack is still at room temperature; thus the thermistor-regulated drive

currents are somewhat higher than after a short period of operation. PM procedures for the memories therefore should follow those for the processor and in-out equipment; current measurements and response to marginal check are then more indicative of normal operation.

Marginal checking during PM procedures is performed while running the Maindecs. Failures caused by varying bias are detected by the program, which itself aids in locating the sensitive circuit. When preventive replacement is unnecessary, margins are varied above and below specifications to discover limits beyond which operation is impossible. These limits are plotted on DEC preventive maintenance voltage charts, which are cross referenced to the maintenance log by page number. Minimum margin specifications for the fast memory, as for all standard logic circuits, are ±7.5v at the +10A and +10B lines, ±3v at the −15C line; exceptions to these minimums are given in the marginal check specification (part of the drawing set accompanying each machine). For the core drive and sense circuits, minimum negative margins are the same as those just mentioned; positive margins are ±6v. Component failures occurring within the minimum margins must be investigated and corrected.

Daily Operator Maintenance

1. Check that all cooling fans are running and cooling air flows freely through the filters.

2. Run the comprehensive Fast Memory Check-out (Maindec 662) without margins. Enter all error halts in the maintenance log, noting cause if known.

3. Run all seven core memory Maindecs, with the two checkerboards last; use no margins. Log error halts, noting cause if known.

4. Replace any noncritical components such as indicators, fuses, etc; note any replacement in the log.

The remaining procedures are to be performed by trained personnel only.

Weekly

Extend the weekly processor PM task to the memory. As with the processor, the provisional use of moderate margin voltages on small sections of logic is acceptable if the memory can thereby be kept operational in spite of a malfunction. Faulty components in the resistor boards, the diode selection modules, or the stacks generally show up as malfunctions involving only one location. In these cases, the processor relocation hardware can often be used to maintain operation. Remember to describe all provisional measures in the maintenance log.

Every 500 Hours

Deposit the Microcheckerboard Program (Maindec 602) in the fast memory. (If there is no fast memory, deposit it in core; since here it may be affected while running, it is highly desirable to have the program tape in the readin format.) For every Type 161C run the Microcheckerboard with margins applied to the following panels, one at a time.

Panels		Function
Left 18 bits	Right 18 bits, parity	
B	H	Read-write switches, read driver
C	J	X-Y and inhibit field selection, inhibit drivers, write X & Y drivers
F	M	Sense amplifiers, sense field selection, strobe; read, write, and inhibit buffers

Plot the results on the preventive maintenance voltage charts. In case of failure at abnormal margin levels, note all circumstances in the maintenance log and cross reference the log to the chart by page number.

Every 1000 Hours

1. Run the Fast Memory Check-out Program (Maindec 622) with margins applied to one panel at a time. Refer to the UMLs (utilization module locations) for the function and location of logic circuits.

2. Run all seven core memory maintenance programs, applying marginal power to the control logic as well as the core driving logic. Test memory control first, using the UMLs to determine the function and location of logic circuits. Then test the core driving circuits, one panel at a time as listed under the 500-hour PM procedure. Plot the results on the preventive maintenance voltage charts. Log all failures and cross reference the log to the charts by page number.

3. Change and clean the air filters in every bay using the procedure given as part of the 1000-hour PM task for the processor (AP 9.3b).

CORRECTIVE MAINTENANCE

The processor manual contains a general discussion of troubleshooting and repair for DEC logic, with a corrective maintenance procedure designed to make most efficient use of field engineering time. Reader familiarity with that material is essential: the fast memory and much of the core memory consist of conventional solid state logic to which the procedures of AP 9.3 apply. Information given here is limited to specific suggestions for troubleshooting the core logic, plus adjustment and calibration procedures for the drive-current power supplies and sense amplifiers.

Before troubleshooting, always check the power supplies for proper voltage output and ripple content as follows.

TABLE 5-1 POWER SUPPLY OUTPUTS

Measurement Terminals	Nominal Output Voltage	Permissible Range	Permissible Peak-to-Peak Ripple
		Type 728	
Orange (+), Black (–)	+10	9.5 to 11.0	1.0v
Blue (–), Black (+)	–15	–14.5 to –16.0	1.0v
		Type 781	
(Jones Strip)			
1 (+), 2 (–)	–17*	–13 to –20	250 mv
1 (+), 3 (–)	–6.8	–6.4 to –7.2	100 mv
1 (+), 4 (–)	–2.5	–2.2 to –2.8	100 mv
1 (+), 5 (–)	–13*	–9 to –17	100 mv
1 (+), 6 (–)	–35	–32 to –38	2.0 mv

*Exact voltage dependent on stack temperature

The 728 Supply is not adjustable, so if output or ripple is not within the tolerance specified, the supply should be considered defective and replaced. The 781 Supply is adjustable, but if outputs are outside tolerance, do not attempt to restore the voltages given above without first checking the core drive currents (see the core stack adjustment and calibration procedures below). If all supply outputs are within tolerance, continue with logic troubleshooting procedures.

The most efficient method of troubleshooting a memory is signal tracing with an oscilloscope. To set up repetitive, predictable behavior, deposit test routines derived from a Maindec in a memory other than the one under test, so that even the most severe malfunctions in the latter do not affect the testing routine. For example the Microcheckerboard (Maindec 602) for core is normally deposited in fast memory; the much larger fast memory test program (Maindec 662) is normally deposited in core. After diagnostics have categorized a malfunction, a small part of the memory logic can be operated repetitively by using the processor EXAMINE or DEPOSIT key with the repeat switch on and the speed controls set to the desired repetition rate. These keys make requests only for read or write access, but read-write access can be requested from the console by executing an appropriate instruction, preferably the slowest (e.g., subtract in memory mode).

Fast Memory Troubleshooting

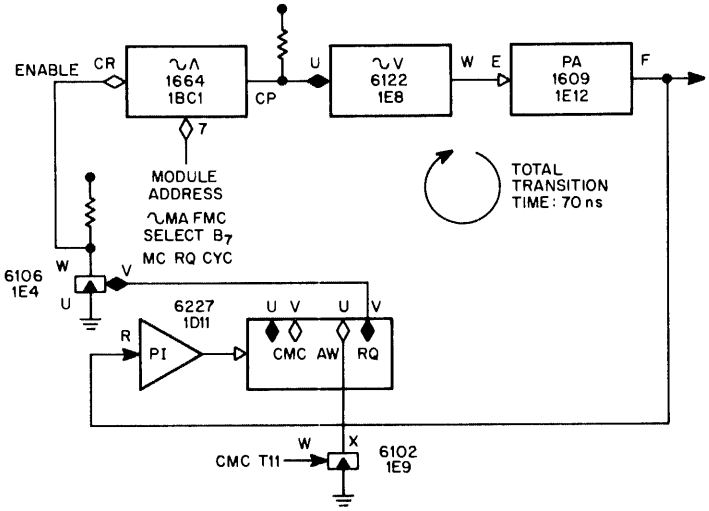
For apparent malfunctions in fast memory, check that the processor is not at fault by replacing fast memory with core (by turning on the processor RIM maintenance switch) and examining one of the first 16 locations. To check for the presence of signals on the bus without disturbing the bus, wheel the scope down to the last memory where the probe can be attached to the load resistors on the bus termination module. Alternatively there are many points in the processor wiring where memory return signals or their derivatives appear; there signal tracing determines whether the memory sends back the appropriate responses.

Core Logic Troubleshooting

Although this section is concerned primarily with troubleshooting the core logic, there is one type of malfunction in memory control that bears discussion because of its effect on that logic. All events in the core logic depend upon timing supplied by the chain in memory control. A malfunction in processor selection that retriggers the time chain while it is already in progress would disturb greatly the core drive circuits. The assertion of any processor request level through the gates at the left in Figure 3-6 triggers the time chain. The first pulse in the chain then clears the await request flip-flop which negates the request level. The assertion of the request level at pin CP of the 1664 corresponding to the selected processor should be 70 nsec, the transition time through the logical loop shown in Figure 5-1. If the await request flip-flop should fail to clear, the request level will last for 400 nsec; i.e., until the cycle request signal from the processor is negated. If CMC AW RQ is then still 1, the memory accepts requests from other processors, retriggering the time chain before the current cycle has finished.

Since the Type 161C has two 18-bit banks with common control logic, a symptom evident in both left and right halves of a data word is not likely to be caused by a core logic malfunction. When margin checks discover errors that affect random bits in all locations, the adjustment procedure given in the following

section should be performed before continuing troubleshooting. As margins are increased from the nominal bias values, a single bit generally fails first. Check for random errors by increasing margins still further. Whenever a very slight increase produces errors in several additional bits, the errors can be considered random in the above sense. Single bit errors that occur at margin levels substantially below those at which most bits fail are <u>not</u> random errors and are probably due to sense amplifier or inhibit driver malfunctions. In the following, a few specific address and data troubles are correlated to the most probable causative malfunction. Remember to check the power supply output voltages according to the table at the beginning of this section before starting troubleshooting procedures.



1664 AND 6122 MODULE LOCATIONS APPLY TO PROCESSOR 0;
FOR OTHER PROCESSORS:

	1664	6122∿V
1	1BC2	1E8W
2	1BC3	1E8Z
3	1BC4	1E8Z

Figure 5-1 Core Memory Selection Timing

A symptom evident in the entire half word in all locations is due to malfunction in one of the three memory drivers, or in the power supply, or failure to generate the strobe or the read, write, or inhibit levels from the signals supplied by memory control. Single address faults that affect only one bit are highly unlikely; they result from a single core whose characteristics have drifted out of specifications, and such changes can occur only under extreme punishment (such as dropping the bay). Since every memory is checked out thoroughly before shipment, it can be assumed that the cores themselves are not malfunctioning. Faults in more than one address are of two types: those that affect an entire half word, and those that affect only a single bit. Symptoms affecting all bits in a group of eight consecutive addresses of a single field can be caused by failure of a read-write switch on the read bus side of the stack, an open resistor on a 1976 Resistor Board or an open diode on a 1020 Memory Diode Unit. A fault that occurs in

every eighth address in all four fields is due to malfunction in a read-write switch on the write bus side. Errors that involve a half word throughout a single field are due to malfunctions in field selection, either for switches, for sense amplifiers, or in the inhibit field select module. A single-bit fault throughout a plane may be caused by a short between an X or Y winding and a sense winding; this is easily recognized by checking the sense preamp outputs (as outlined in the next section).

Multiple Addressing

The problem of detecting a shorted diode in a Type 1020 is difficult. These diodes ensure the selection of only one X or Y winding; when a diode is shorted, multiple addressing occurs in the write portion of the cycle. Figure 5-2 shows a simplified addressing matrix for X only. This hypothetical memory contains 256 locations with 16 X windings. During the read portion of a cycle, one of the four read-write switches in modules 1 and 2 becomes a current sink while one of the switches on the "write" side (in modules 3 and 4) is a current source. The source and sink roles are reversed during write, although the same switches are selected. When, for example, address 0000 is selected, current flows from pin Z of switch 3 through the left diode at 0000 (the lower left corner of the matrix), through the winding between the two pin connections and into pins Y-Z of switch 2 on the read side. During write, current flows out 2Y-Z through the winding at 0000, through the right diode and into 3Y. At every unaddressed location, the diodes prevent spurious current flow during write. If only one location were addressed forever, the two diodes there could be replaced by wires without affecting the operation of the memory; the diodes at all unaddressed locations would still prevent multiple current paths.

Consider the symptom of a shorted diode. Suppose the left diode at location 0001 is shorted and thus conducts current in either direction. When location 0001 is addressed, the shorted diode has no effect: in read, current flows properly through it; in write, although the short makes the entire line at 3X a potential current source, no current sink other than 3W establishes flow. A little effort reveals that the shorted left diode at 0001 also has no effect for any location whose more significant pair of address bits is other than 00. However, consider the case of some other location whose address begins with 00, e.g. 0000. During read, 3Z is a source and 2Y-Z a sink; since the X line at switch 3 cannot become a source (it is not selected), a short still has no effect. But during write, 2Y-Z is a source while 3Y is a sink. Current then flows through the 0000 winding and its right diode, but also flows through winding 0001 and the shorted left diode onto the line at 3X. Since this line is now also a source, a current can flow through the left diode in the winding at 0101, through the winding and right diode at 0100, and into 3Y, so that there are two current paths even though only one location is addressed. As another example, suppose 0010 is addressed. Write current from 2Y-Z then flows properly through winding 0010 and the right diode into 4Y; but it also flows through the winding and shorted left diode at 0001, through the diode and winding at 0101, and through the winding and the right diode at 0110.

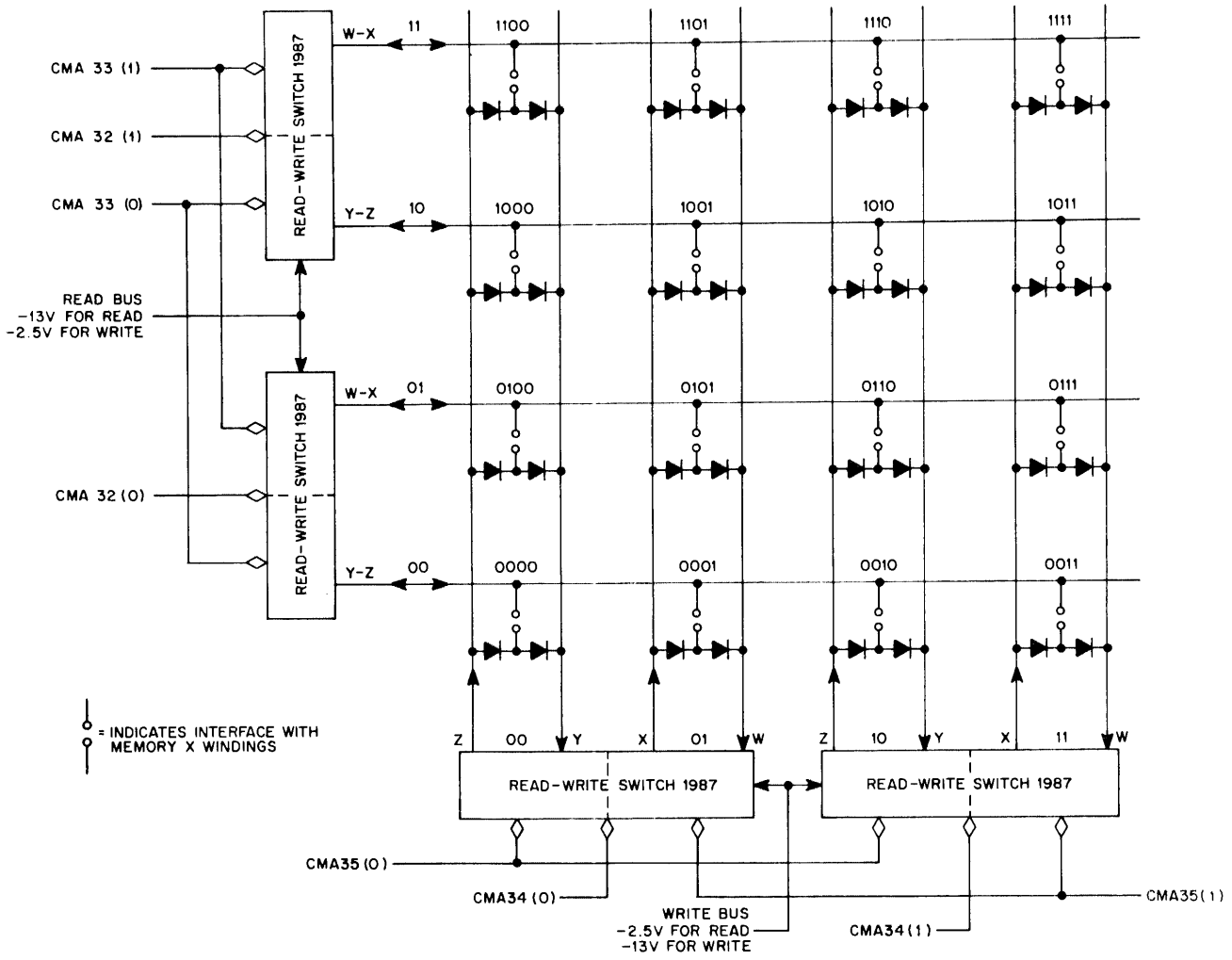


Figure 5-2 Addressing Matrix, X Only

The result of multiple current paths is failure to switch the addressed core fully to the 1 state. This means that the shorted left diode at 0001 results in poor positive margins at all locations whose X addresses begin with 00, except for 0001. In the Type 161, therefore, a shorted X selection diode pointing into the stack causes poor margins in groups of eight consecutive addresses beginning at every 64th in a single field. Within each group, the one address containing the shorted diode operates correctly. By similar reasoning it can be seen that this type of short in Y results in a group of 512 addresses with poor positive margins except in a subgroup of 64; the less significant three address bits in Y (CMA bits 27-29) indicate the location of the short.

A shorted right diode also exhibits symptoms during the write portion of the cycle. Consider a shorted right diode at 0001. Whenever line 00 (2Y-Z) is a source, the short has no effect because the healthy diodes in other locations prevent multiple current paths. Also, whenever lines 01 (3X-W) are selected, the shorted diode has no effect. During read, 3X is a current source, and thus a spurious current path

would again require a bad diode elsewhere. However, consider the situation in which some location diagonally removed from 0001 is selected, e.g., 0100. In read there is still no effect because 3Z is a current source while 2W-X is a sink; there is no way for the line at 3W to be a source and establish current through a shorted diode. However, in write, when 2W-Z is a source and 3Y a sink, current flows properly through the winding at 0100, but also through the winding and right diode at 0101, through the shorted right diode and winding at 0001, and through the winding and right diode at 0000. A similar spurious path is established for any location whose most significant and least significant address bits both differ from those of a location containing a bad diode.

The spurious path creates a current division, so that less write current flows through the addressed location; this produces poor positive margins at any location for which such a spurious path exists. In the Type 161, therefore, a shorted right diode results in generally poor positive margins throughout a memory field except in two groups of eight locations for every Y: a group of consecutive addresses over which the more significant three bits address one coordinate of the short; and a group containing every eighth address when the less significant three bits are the other coordinate. If the short is in Y, the memory exhibits poor positive margins except for two groups of 512 locations: a contiguous group for which the more significant three bits address the short; and a group consisting ot eight subgroups of 64 each determined by the lower three bits.

Single-Bit Errors

Single-bit errors involving all four fields may be due to malfunction in either an inhibit driver or a sense amplifier. The leads to the inhibit windings in the stack are purposely wired to give sufficient length for insertion of a current probe between the stack connectors and the Inhibit Diode modules Type 1021. The oscilloscope may check for the presence of inhibit current on an individual line. The sense amplifiers have three test jacks for checking internal waveforms. The two black jacks are the differential output of the selected preamplifier. For a 1, these show a half-sinusoid pulse of either polarity, approximately 1 μsec long at its base, and superimposed on noise that occurs in the beginning and end of the read and write current waveforms. For a 0, only the noise is present; absence of the half-sinusoid indicates that the selected core has not changed state. The red test jack shows a waveform quiescent at −3v, which rises to ground whenever the preamplifier waveform has sufficient amplitude. Consequently the level is ground for approximately 0.75 μsec for a 1, and quiescent at −3v for a 0.

ADJUSTMENTS

There are four adjustments: read-write current, inhibit current, sense amplifier clamp level, and read strobe timing. The memory current adjustments are screwdriver trimpots located on the 1711 Power Supply

Control module within the 781 Supply. Holes in the aluminum module frame provide access to the trimpots; the upper regulates inhibit current, the lower RW current. Clockwise rotation increases current (i.e., makes the supply output voltage more negative). The clamp adjustment is a similar trimpot on the 4527 Master Slice Control in 2F(M)19; clockwise rotation raises the clamp level (makes it more positive) resulting in increased current through the selected preamplifier stage. A lower sense input then satisfies the slicer as an indication of a change of state in core.

Read Strobe Timing

The read strobe is timed by the 1310 in 1E13: since this is a delay line, adjustment is made by reconnecting the jumper to another tap. It is rarely necessary to do this in the field; if conditions seem to indicate that such adjustment is necessary, DEC Field Services should be consulted to ensure that the symptom of maladjustment does not indicate some other remedy. Normally adjustment is necessary only when memory operation is required over most unusual temperature extremes. In a normal environment it should be required only in the highly unlikely event that core bank characteristics should change drastically.

Read-Write and Inhibit Current

The read, write X, and write Y buses are each brought out as an inch length of wire between a pair of standoffs. For the left half-memory these are at the bottom edge of panel B; for the right half, at the bottom edge of H. In each set, read is at the left, write X in the center, write Y at the right. The currents vary ±20% depending on stack characteristics; nominal values are 400 ma for read, 200 ma for write. Under no circumstances should these values be construed as goals for the adjustment of read-write current; the proper object of the adjustment procedure is to maximize the range of marginal check voltages over which the memory will operate.

The first step is to take margins on the memory. Whenever the margin levels are not symmetrical (that is, the permissible positive margin is greater than the negative or vice versa) the range can be made symmetrical by adjusting the clamp level at the 4527 Master Slice Control. This level should be readjusted to regain symmetry after any other adjustment is changed. There are two possibilities:

1. If the positive deviation that results in dropping bits is inadequate, the RW power may be raised or the inhibit power lowered. Before adjusting either, note the present value at the 781 Supply. Raise the RW power first; if the positive margins are not improved by this restore the level to its former value and try lowering the inhibit voltage. Again, if positive margins are not improved, restore it to its original value. In general, if the positive deviation for reliable operation is less than 8v, one of these adjustments should improve it.

2. If the negative margin deviation is inadequate, inhibit power may be raised or RW power lowered. As above, note the present values and restore them if there is no improvement. If the permissible negative deviation is less than 8v, either raising inhibit power or dropping RW power should improve it. After this step, readjust the clamp level at the 4527 to make the permissible margin range symmetrical about +10v.

Since the objective is to maximize the permissible margin range over which the memory operates properly, an adjustment that improves the deviation in one polarity by 2v while degrading deviation in the other by only 1v is a step in the right direction. There is a wide range of RW levels and corresponding inhibit levels at which the margin range is greatest; the lowest power levels within this adjustment range should be used.

Sense Amplifier Clamp Level

If the marginal range for proper operation can be increased to ±8v by tuning the RW and inhibit levels alone, the adjustment procedure is complete. If there is difficulty achieving this range, observe the slice level at the red test jack on one of the sense amplifiers. The ground excursion of this level should be symmetrical with respect to the superimposed strobe pulse evident near its center. As the marginal deviation is increased, the level narrows until it eventually disappears; however, it should never be significantly asymetrical with respect to the superimposed strobe. If it is, check all sense amplifiers to be sure that the symptom is universal, in which case the strobe may have to be readjusted, and preliminary tuning of RW and inhibit levels must be done by observing the marginal behavior of the slice level on at least three typical sense amplifiers. The narrowing of the level with negative margin deviations must parallel as precisely as possible the narrowing with positive deviations. The objective now becomes achievement of the maximum marginal range over which the slice level is present. To achieve this maximum, the RW and inhibit power levels may be adjusted as above.

When the slice level is present over the maximum possible marginal range, the strobe must be readjusted so that it is as close as possible to the center, regardless of marginal deviations. More noise is on the slice level at the trailing than at the leading edge. The strobe should be centered in the reliable portion, which may not necessarily be at the centroid of the level including its noise. If no amount of repetition of these procedures can achieve a 16v margin range for proper memory operation, a DEC Field Serivce representative should be called to determine the source of the deterioration.

CHAPTER 6

ENGINEERING DRAWINGS

Reduced copies of engineering drawings follow in this chapter. For an explanation of Digital Equipment Corporation's logic, symbology, and information on drawings the reader can refer to Chapter 4 and Appendix 1 of the Arithmetic Processor 166 Instruction Manual, F-67 (166).

SEMICONDUCTOR SUBSTITUTION

Most DEC semiconductors used in modules of the Fast Memory Type 162 and Core Memory Type 161C can be replaced with standard EIA components as specified below. Exact replacement is recommended for semiconductors that are not listed.

DEC	EIA
D-003	1N994
D-007	1N277
D-662	1N646

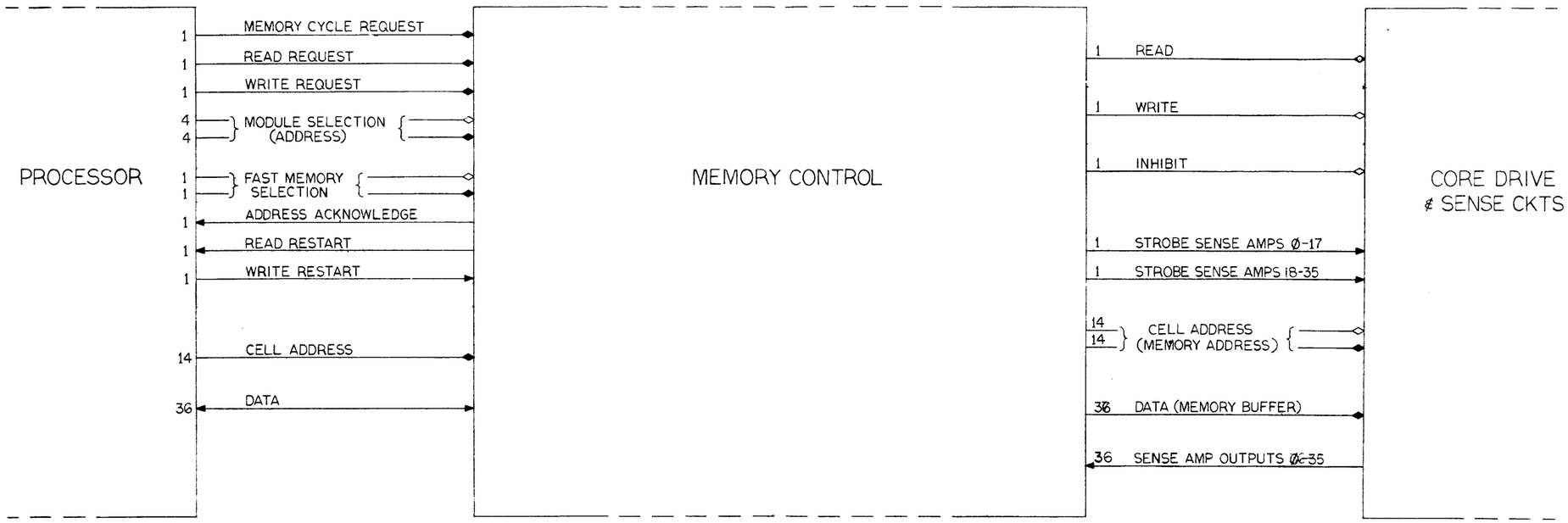
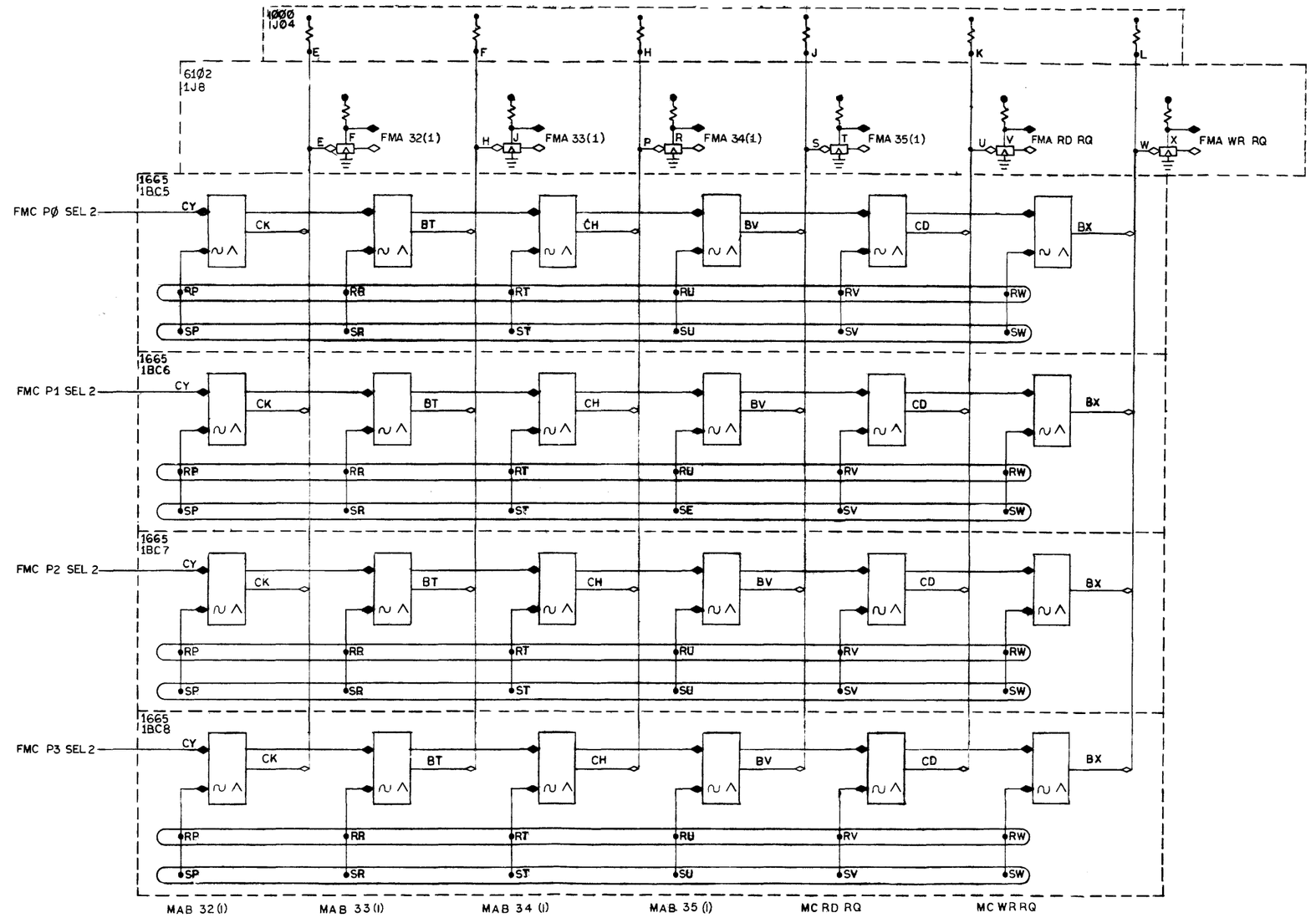
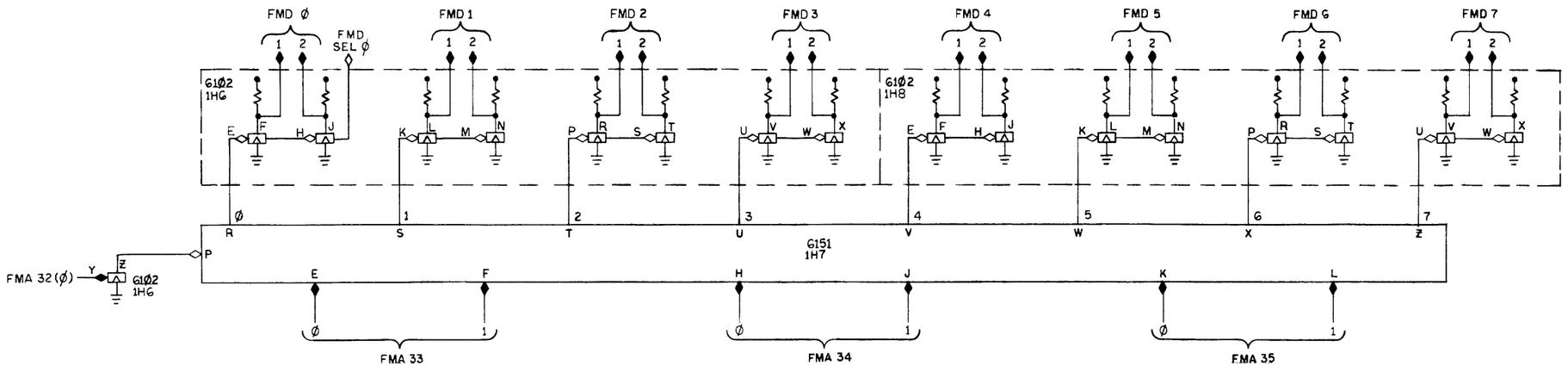


Figure 2-1 System Block Diagram, Fast Memory Type 162 (SD-D-162-0-SBD)



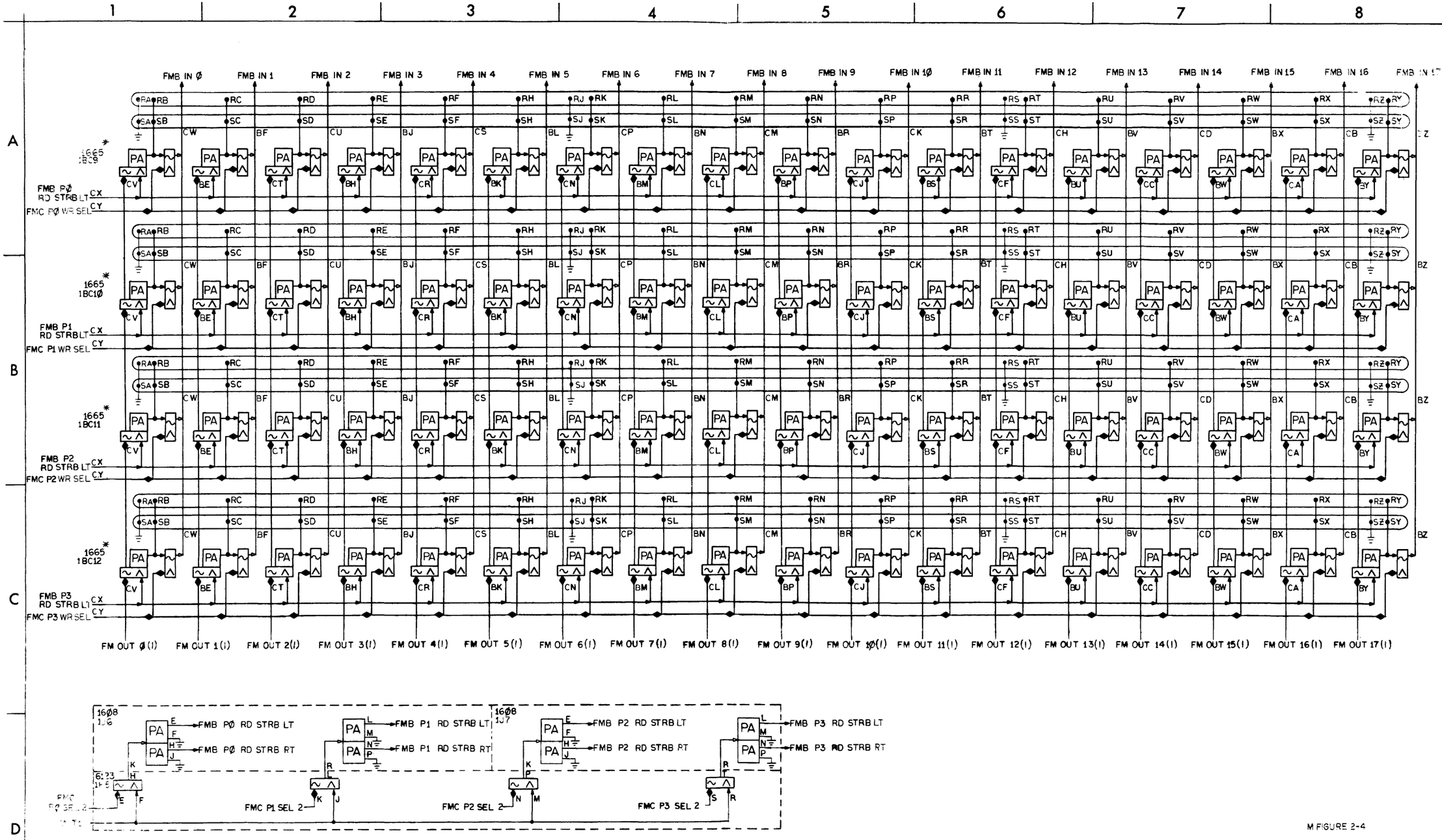
M FIGURE 2-2

Figure 2-2 FMA - Fast Memory Address, RD RQ, WR RQ (BS-D-162-0-FMA)



M FIGURE 2-3

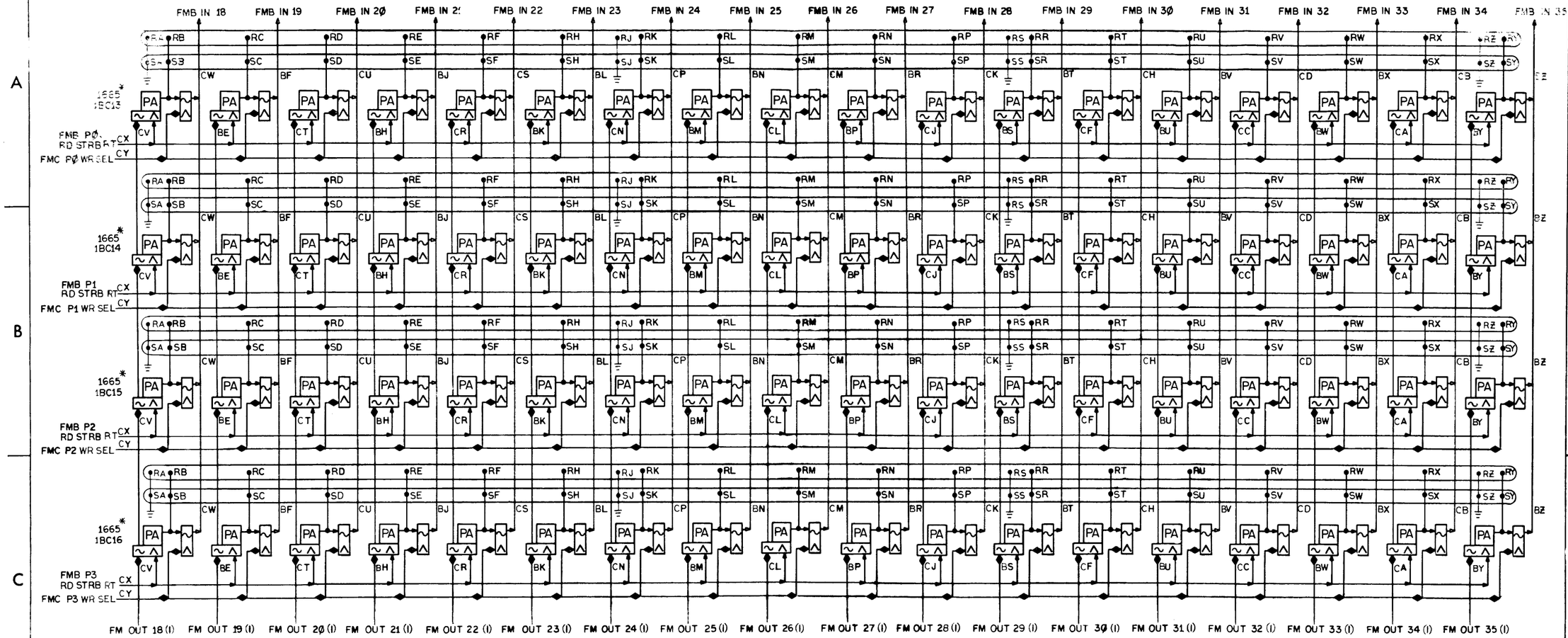
Figure 2-3 FMD - Fast Memory Address Decoding (BS-D-162-0-FMD)



M FIGURE 2-4

NOTES
1. 1665 GND'S BD, CE, CZ
2. * THESE MODULES INSTALLED AS REQUIRED

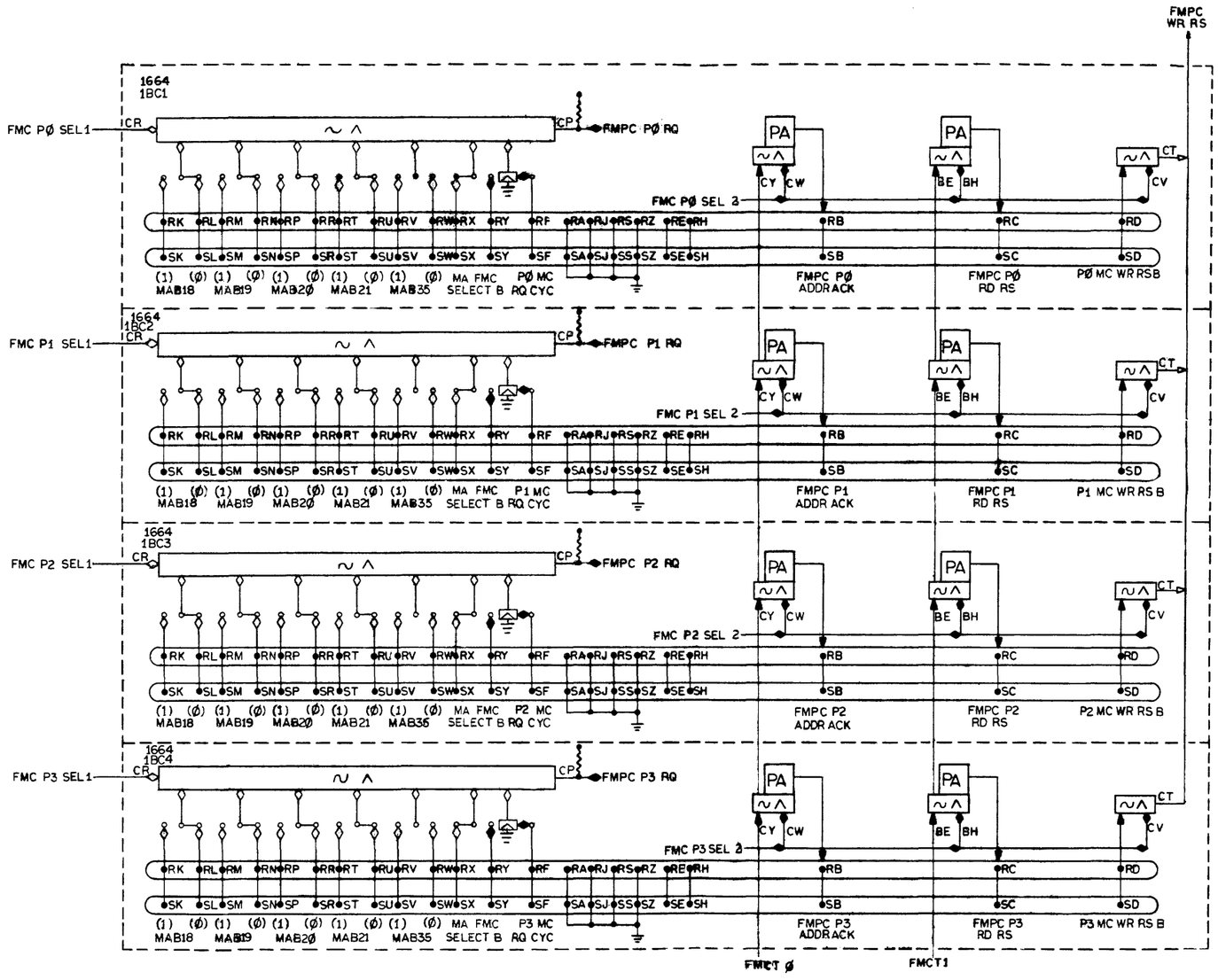
Figure 2-4 FMB - Fast Memory Bus 0-17 (BS-D-162-0-FMBL)



NOTES:
1. 1665 GND'S BD, CE, CZ.
2. * THESE MODULES INSTALLED AS REQUIRED.

M FIGURE 2-5

Figure 2-5 FMB – Fast Memory Bus 18-35 (BS-D-162-0-FMBR)



NOTE
1. 1664 GND BD,BP,BR,BZ,CA,CL,CM.

M FIGURE 2-6

Figure 2-6 FMPC - Fast Memory Processor Control (BS-D-162-0-FMPC)

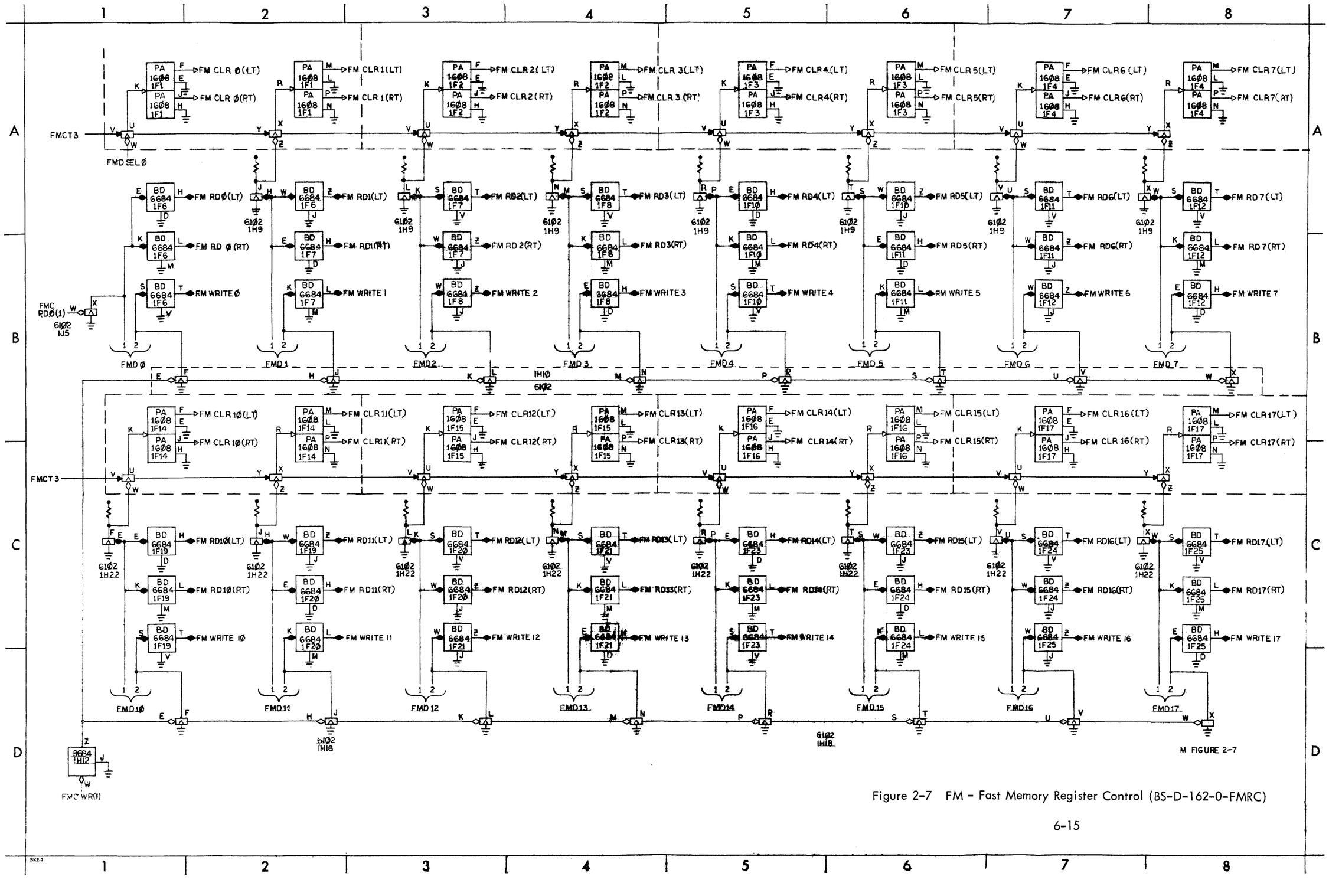


Figure 2-7 FM - Fast Memory Register Control (BS-D-162-0-FMRC)

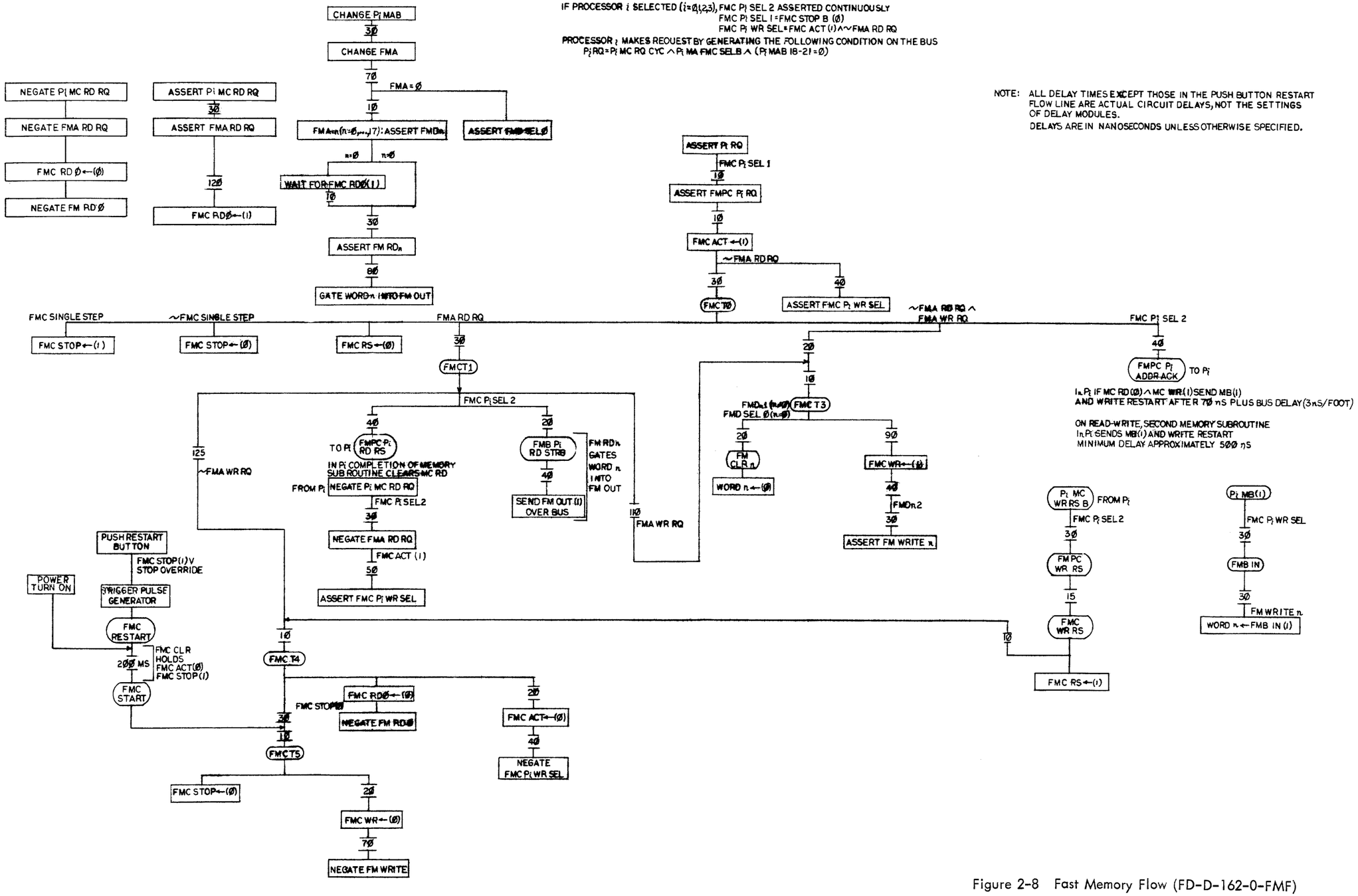


Figure 2-8 Fast Memory Flow (FD-D-162-0-FMF)

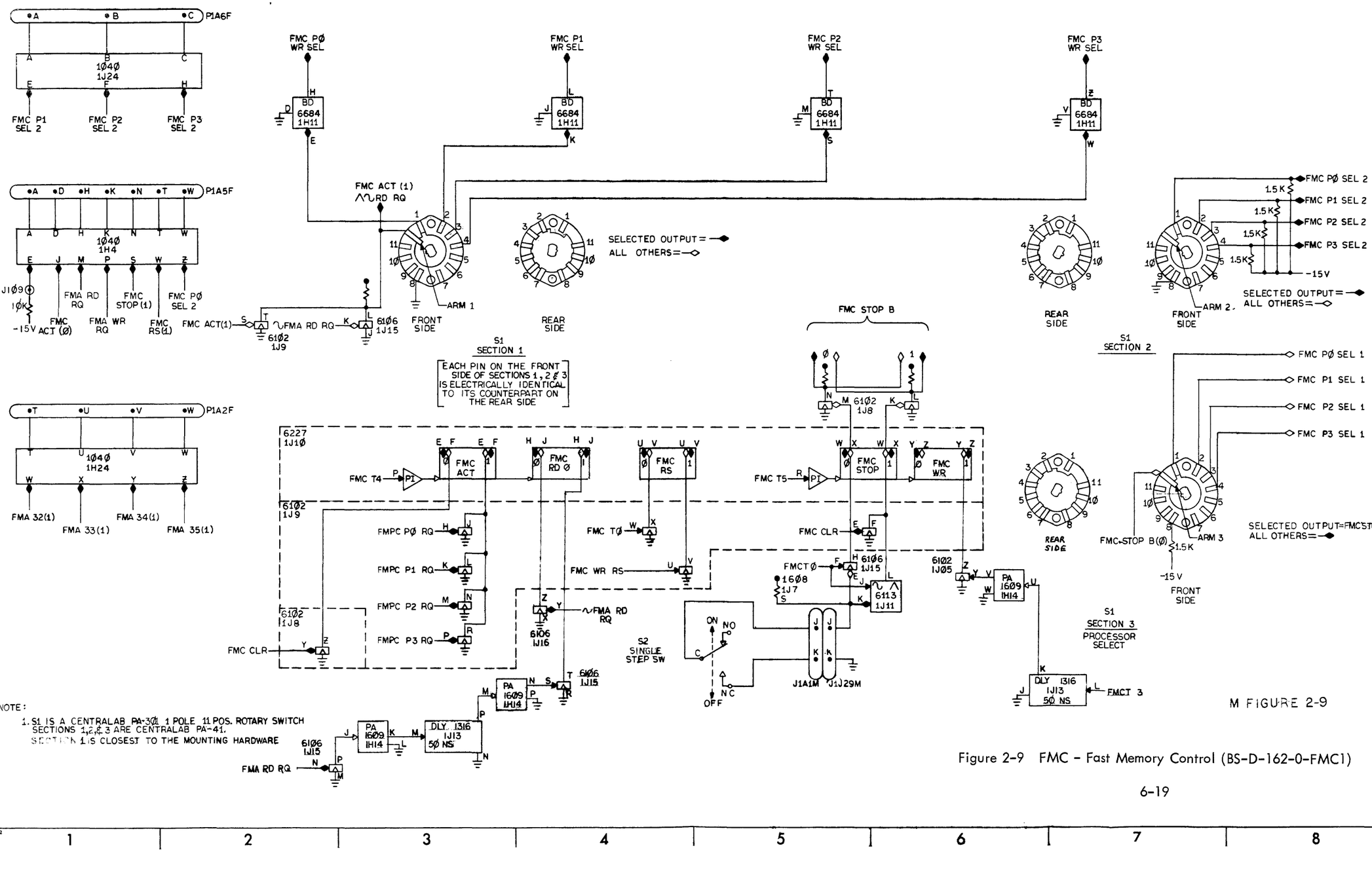
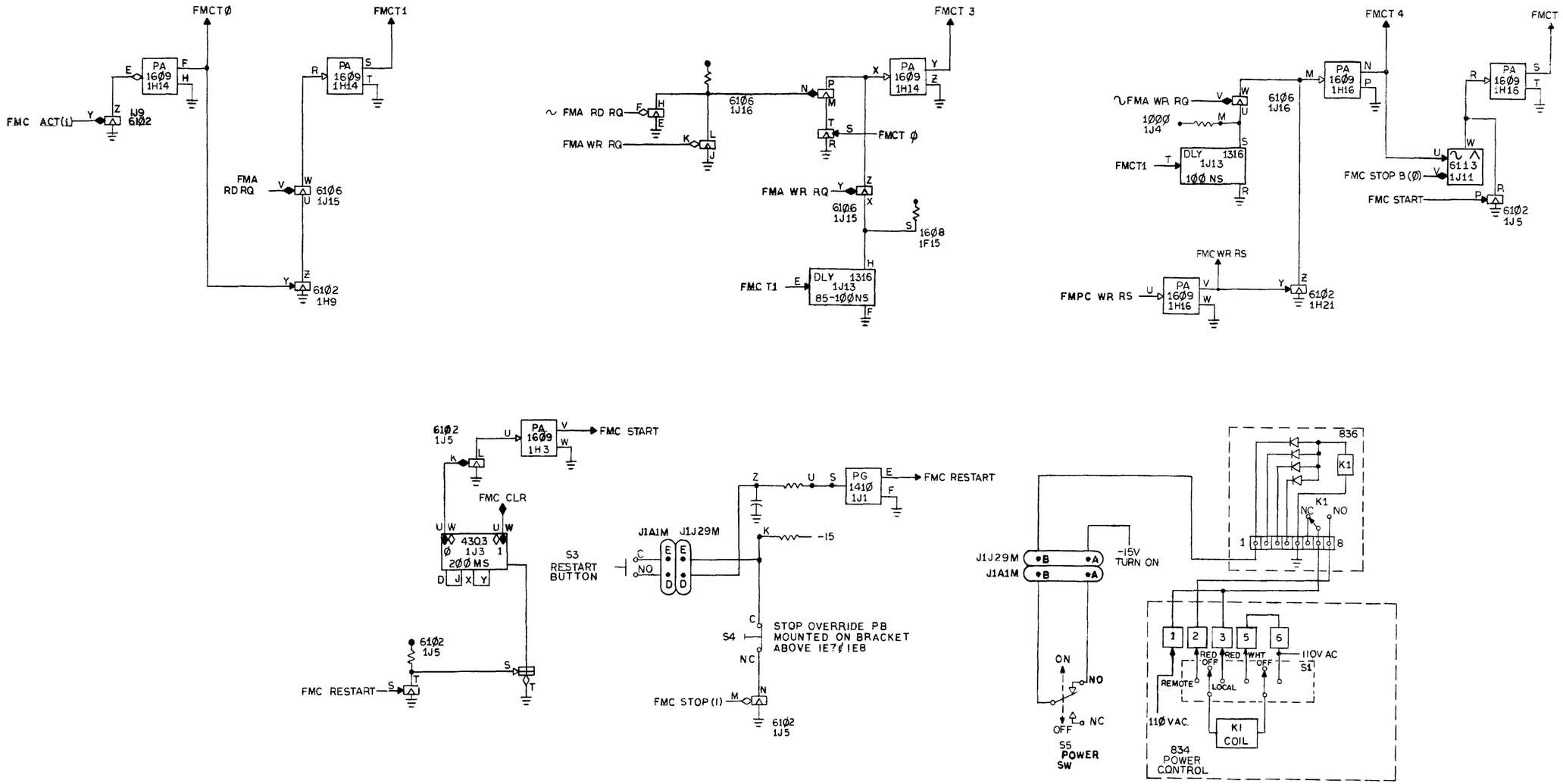
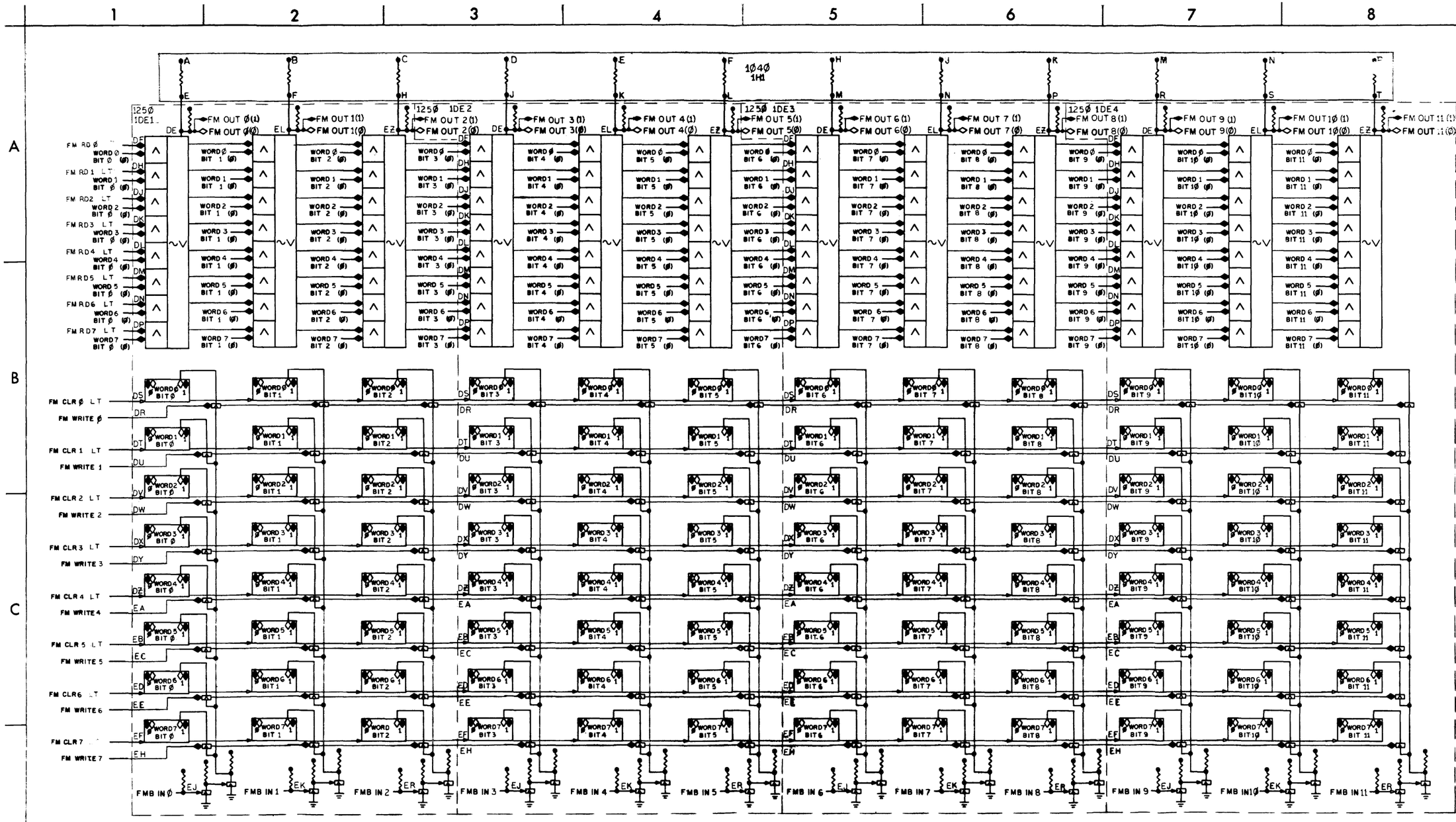


Figure 2-9 FMC - Fast Memory Control (BS-D-162-0-FMC1)



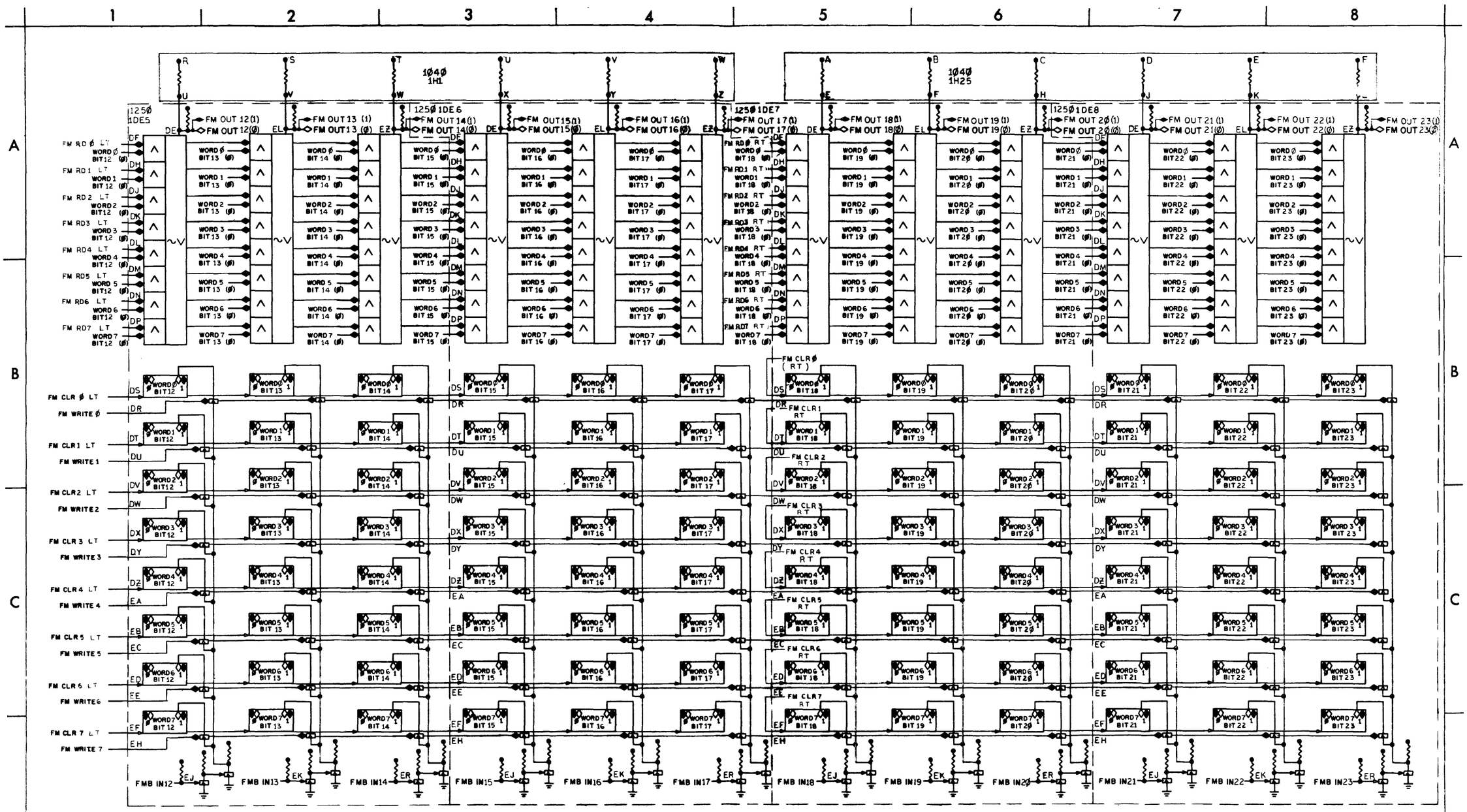
M FIGURE 2-10

Figure 2-10 FMC - Fast Memory Control (BS-D-162-0-FMC2)



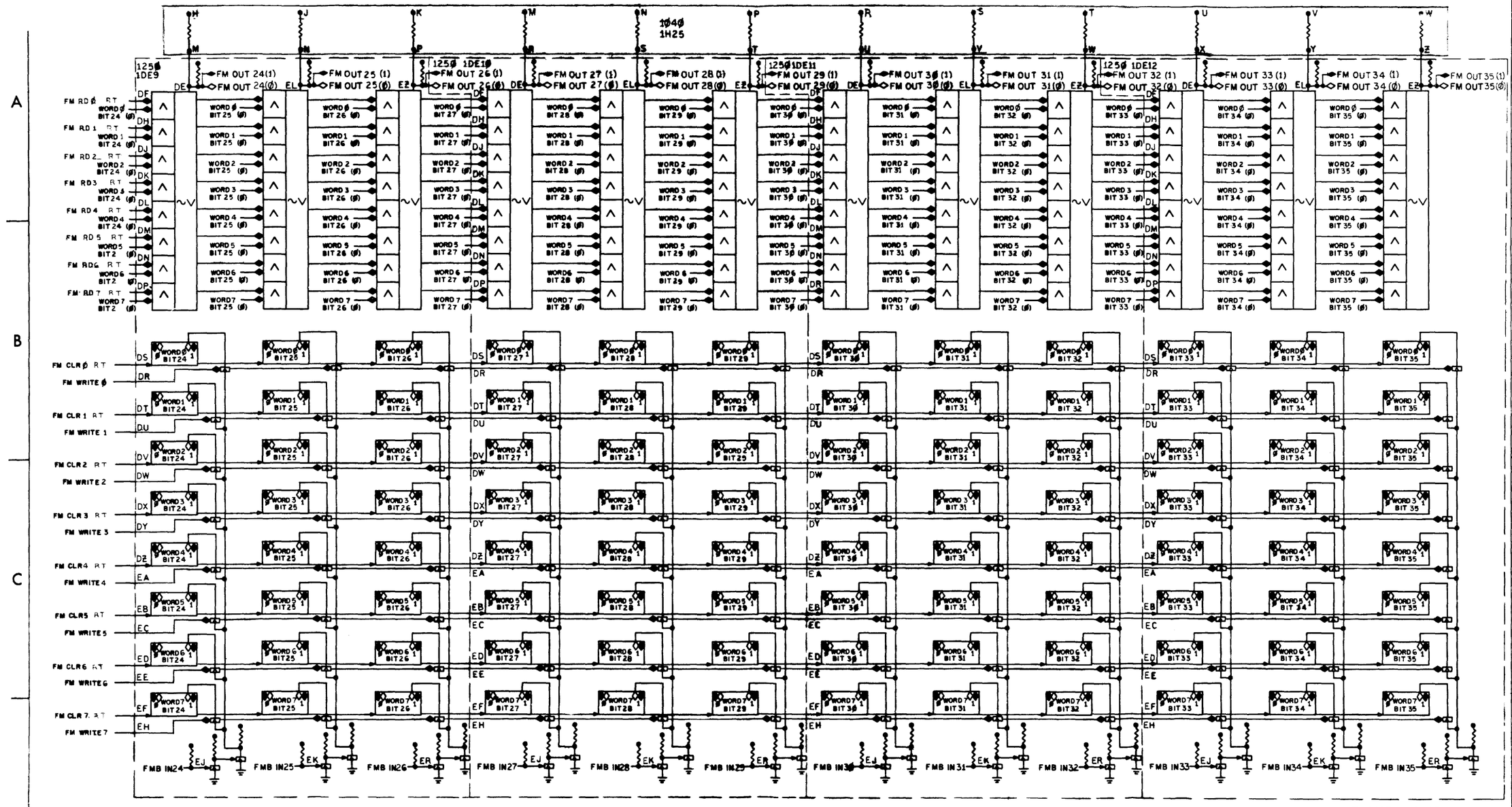
M FIGURE 2-11

Figure 2-11 FM - Fast Memory Registers (BS-D-162-0-FMR, Sheet 1)



M FIGURE 2-12

Figure 2-12 FM - Fast Memory Registers (BS-D-162-0-FMR, Sheet 2)



M FIGURE 2-13

Figure 2-13 FM - Fast Memory Registers (BS-D-162-0-FMR, Sheet 3)

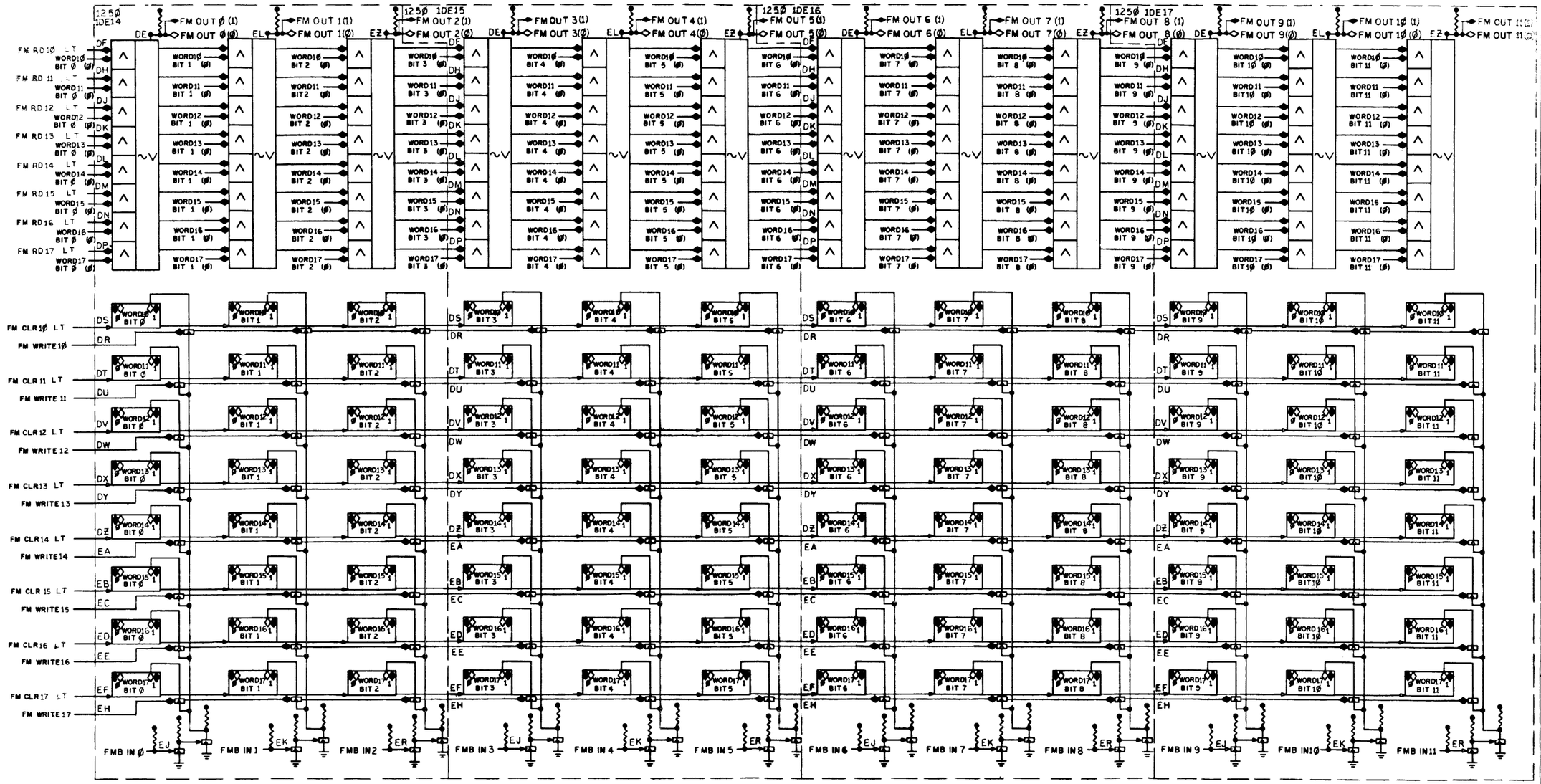


Figure 2-14 FM - Fast Memory Registers (BS-D-162-0-FMR, Sheet 4)

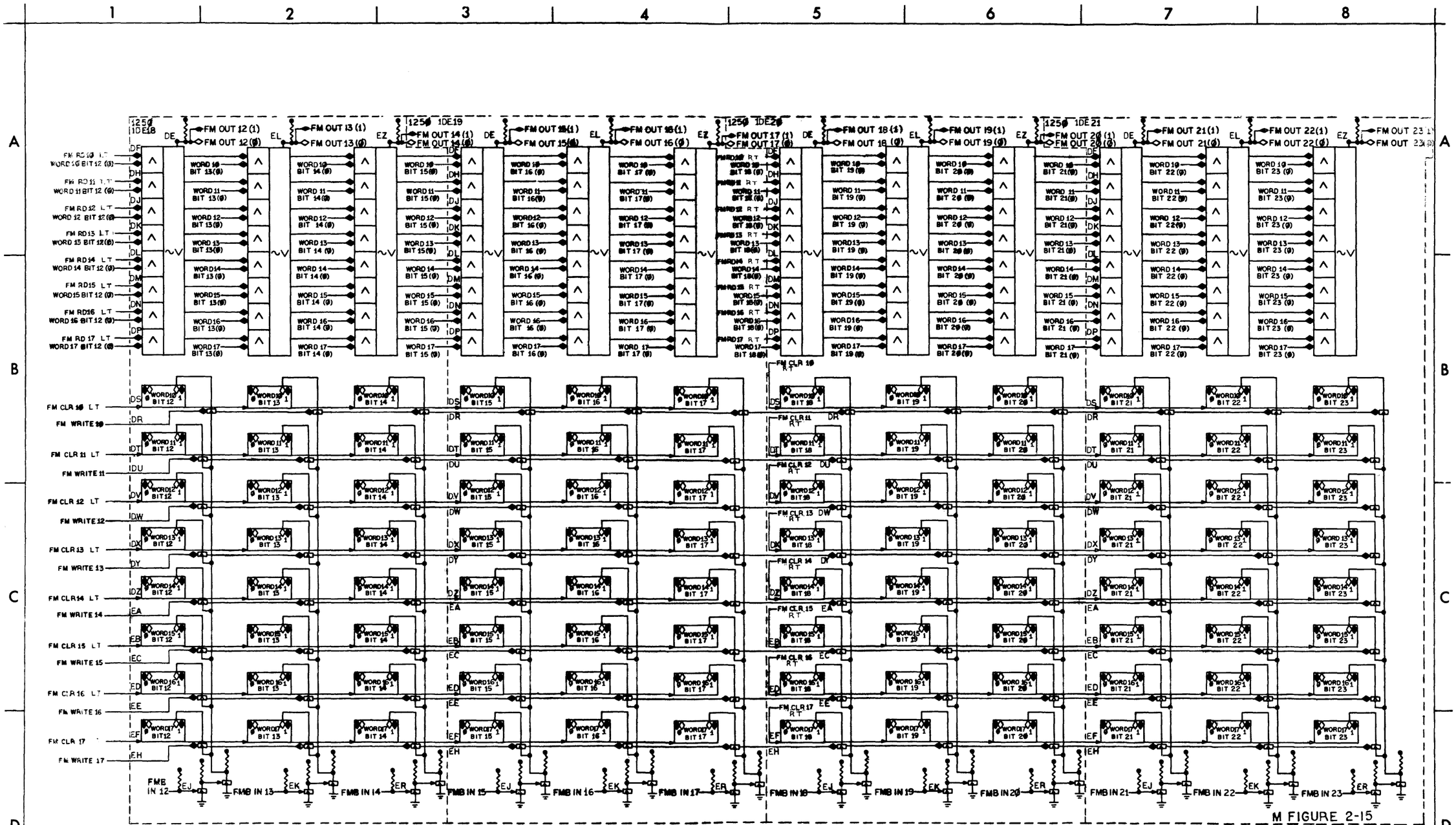
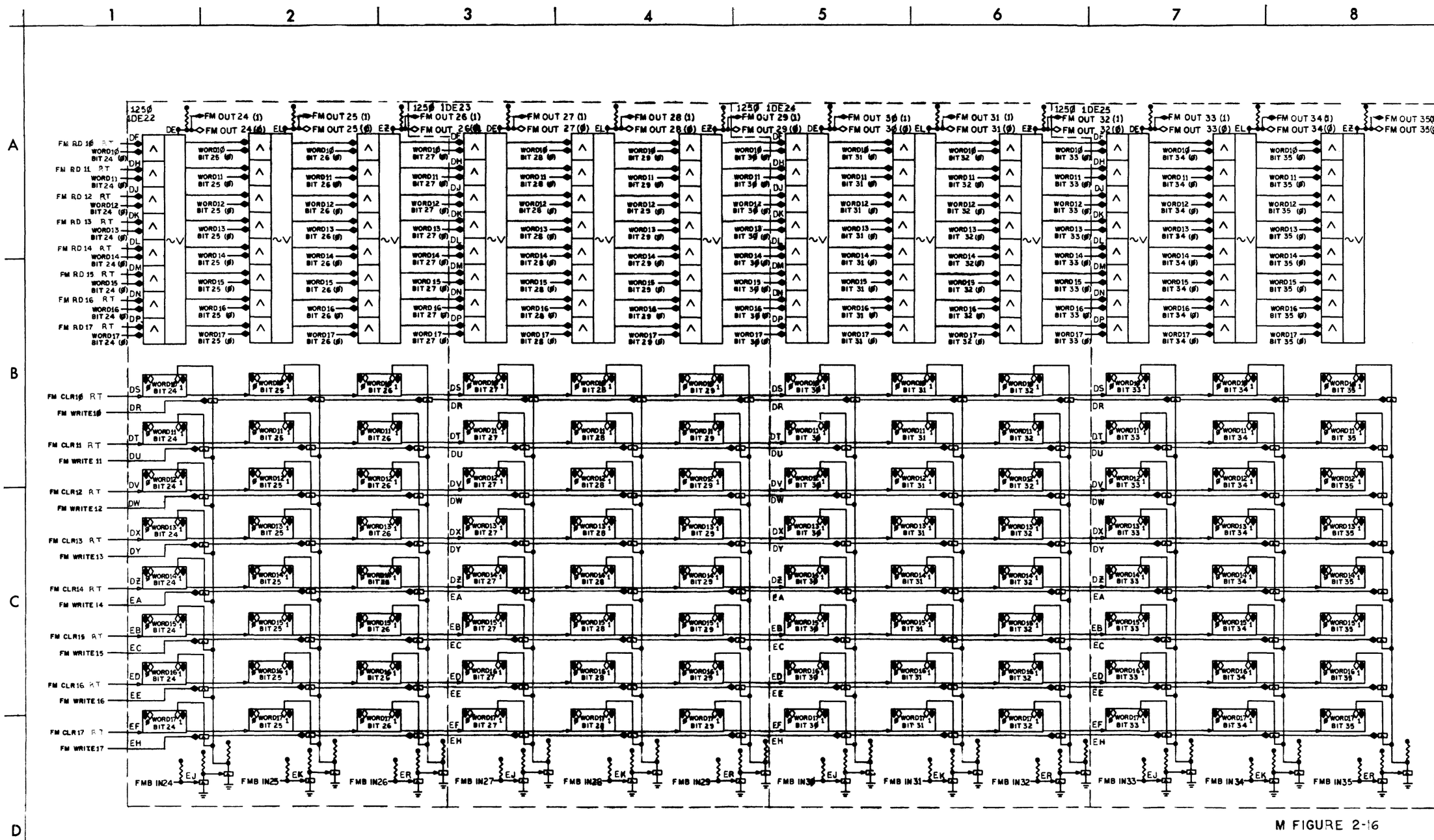
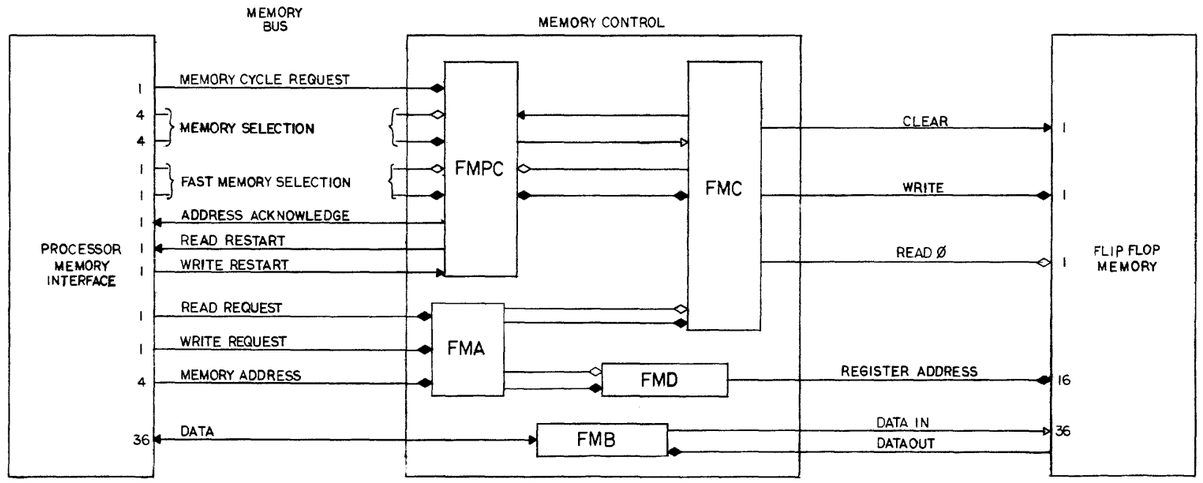


Figure 2-15 FM – Fast Memory Registers (BS-D-162-0-FMR, Sheet 5)



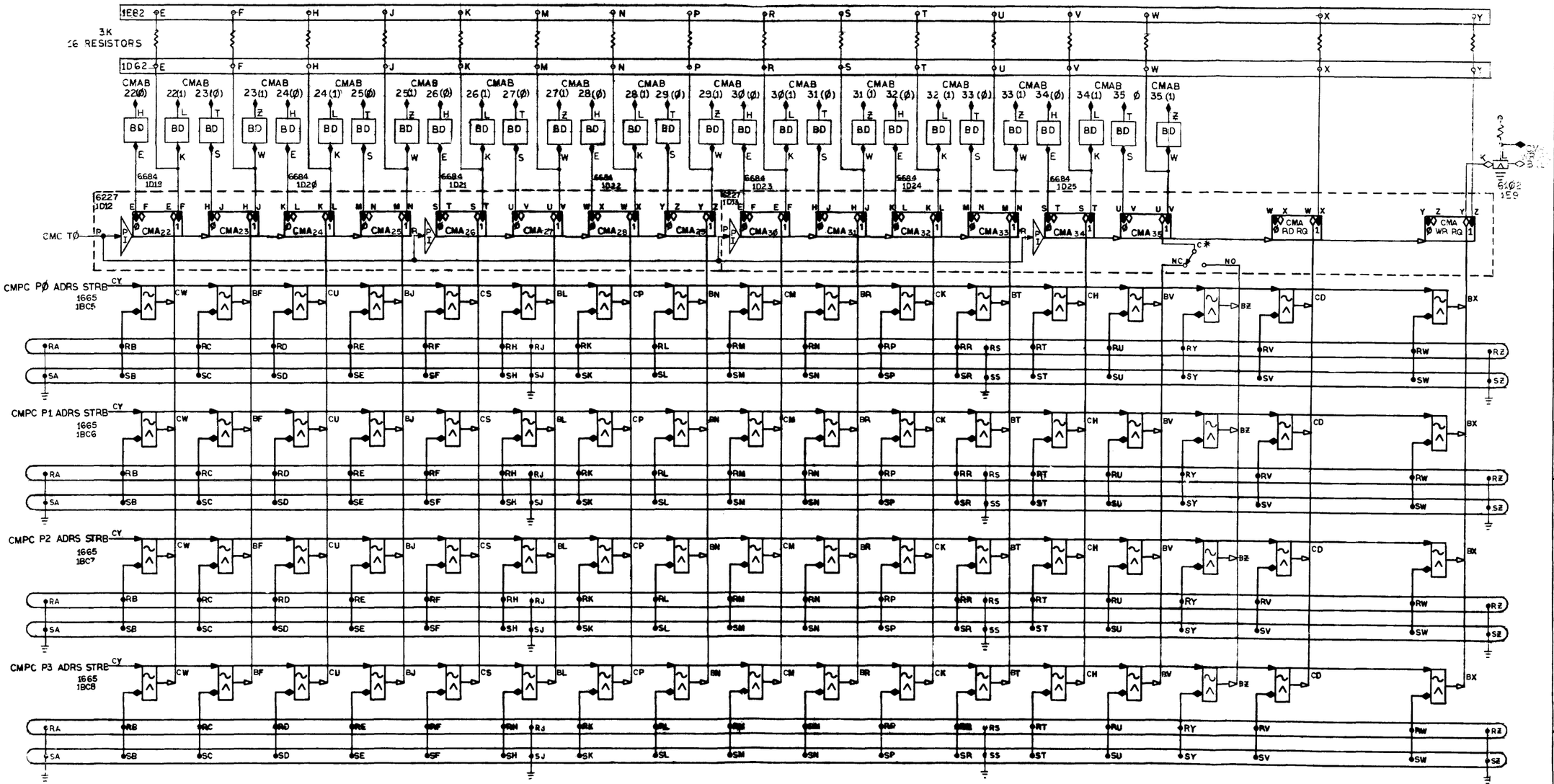
M FIGURE 2-16

Figure 2-16 FM - Fast Memory Registers (BS-D-162-0-FMR, Sheet 6)



PREFIX CODES
FM 2-7, 2-11 TO 2-16
FMA 2-2
FMB 2-4,5
FMC 2-9,10
FMD 2-3
FMPC 2-6

Figure 3-1 System Block Diagram, Core Memory Type 161C (BS-D-161C-0-SBD)



NOTE
1665 GND D, BE, BZ.

* INTERLEAVE SWITCH (MICRO TOGGLE) MOUNTED ON BRACKET BELOW 1D11 & 1D12

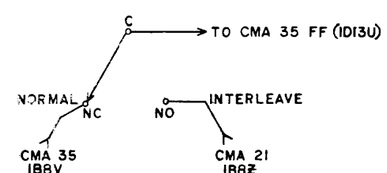


Figure 3-2 CMA - Core Memory Address, RD RQ, WR RQ (BS-D-161C-0-CMA)

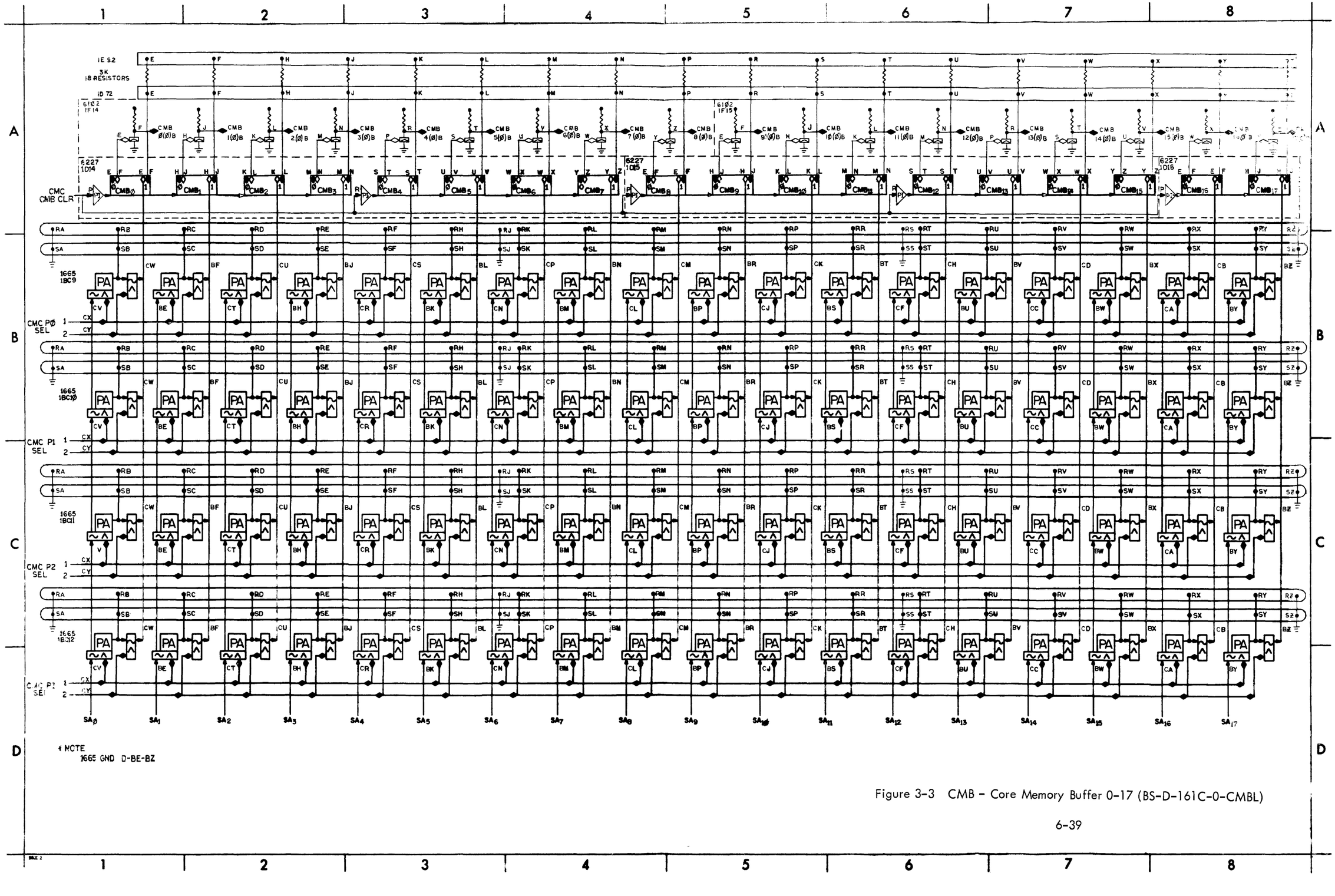


Figure 3-3 CMB - Core Memory Buffer 0-17 (BS-D-161C-0-CMBL)

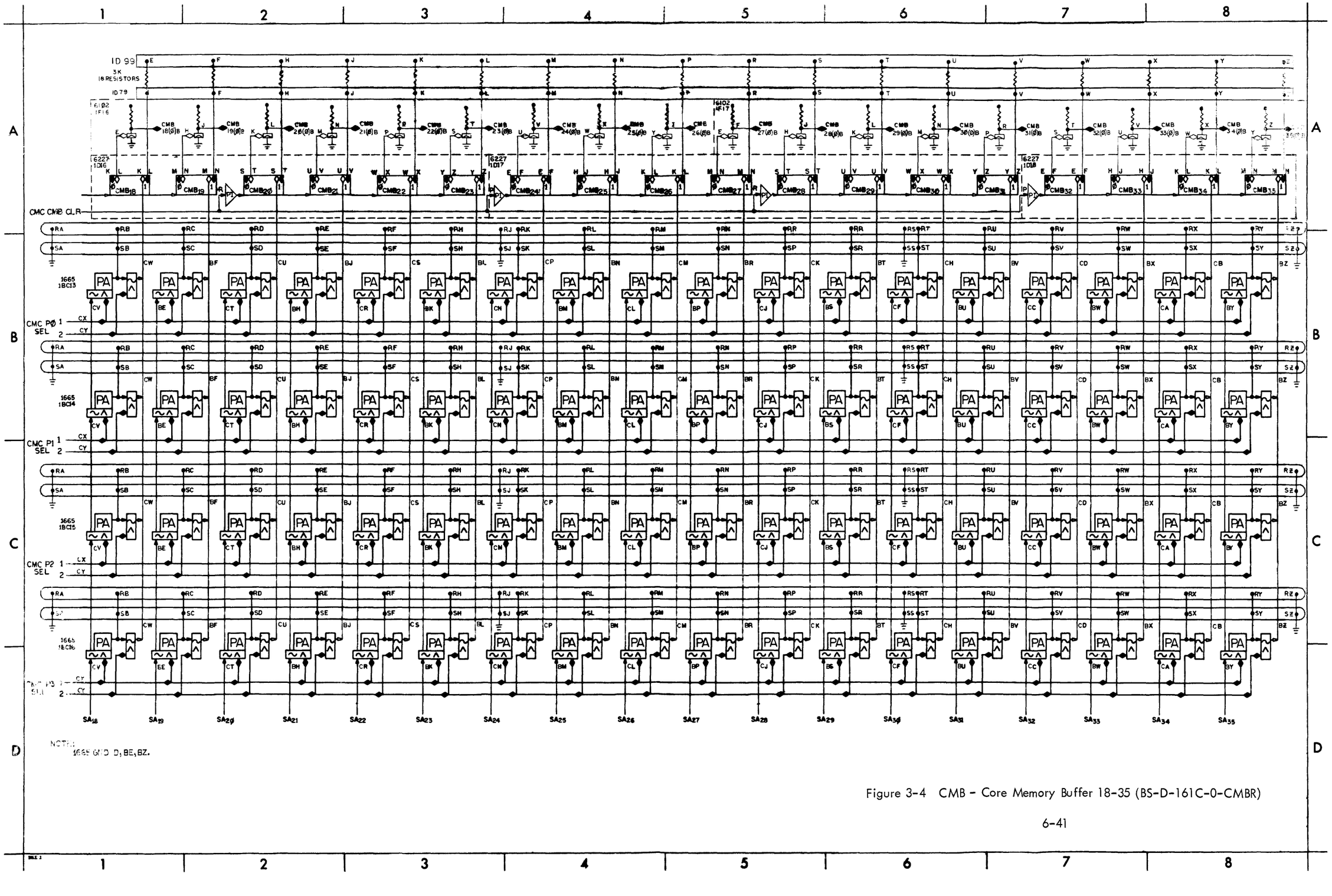


Figure 3-4 CMB - Core Memory Buffer 18-35 (BS-D-161C-0-CMBR)

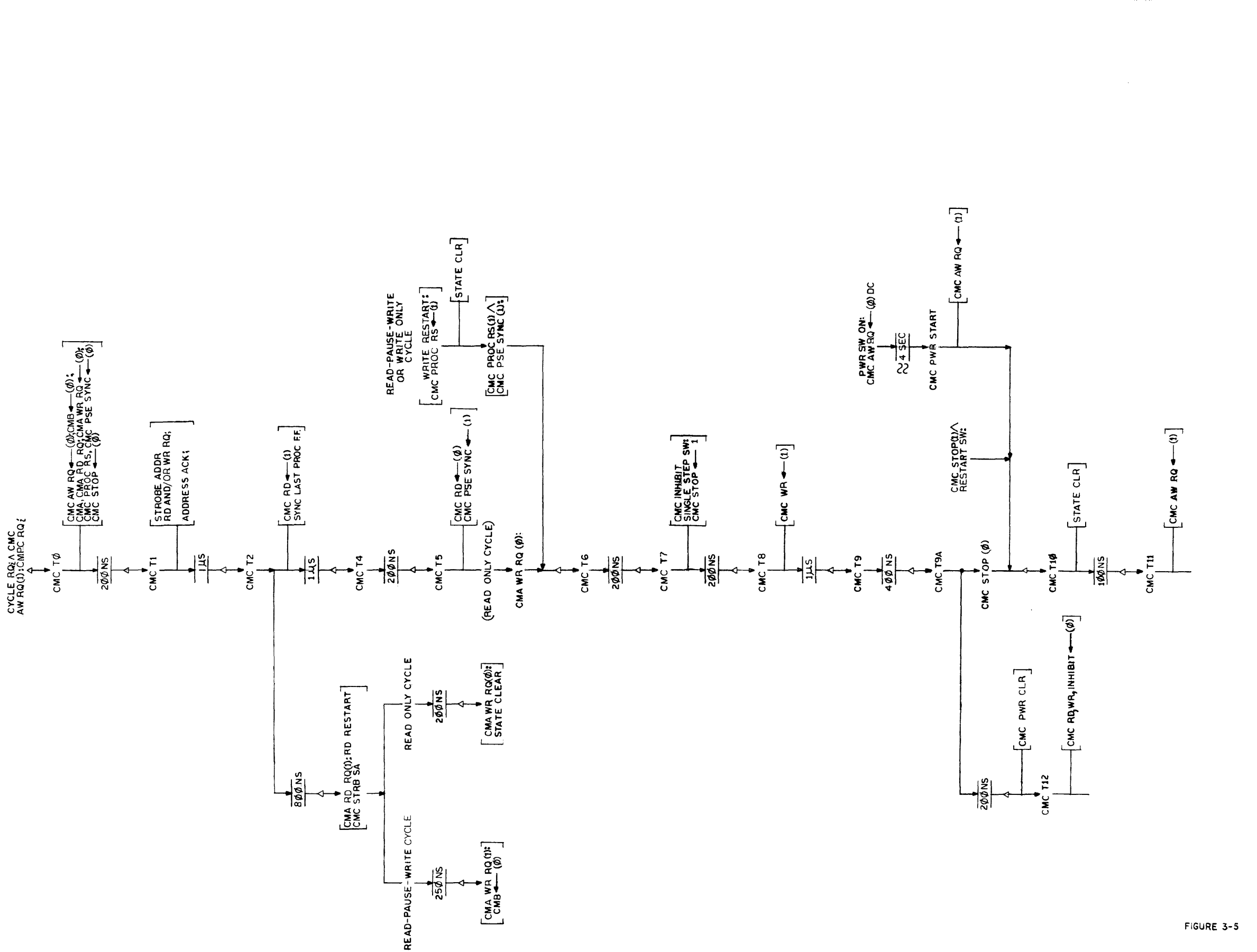
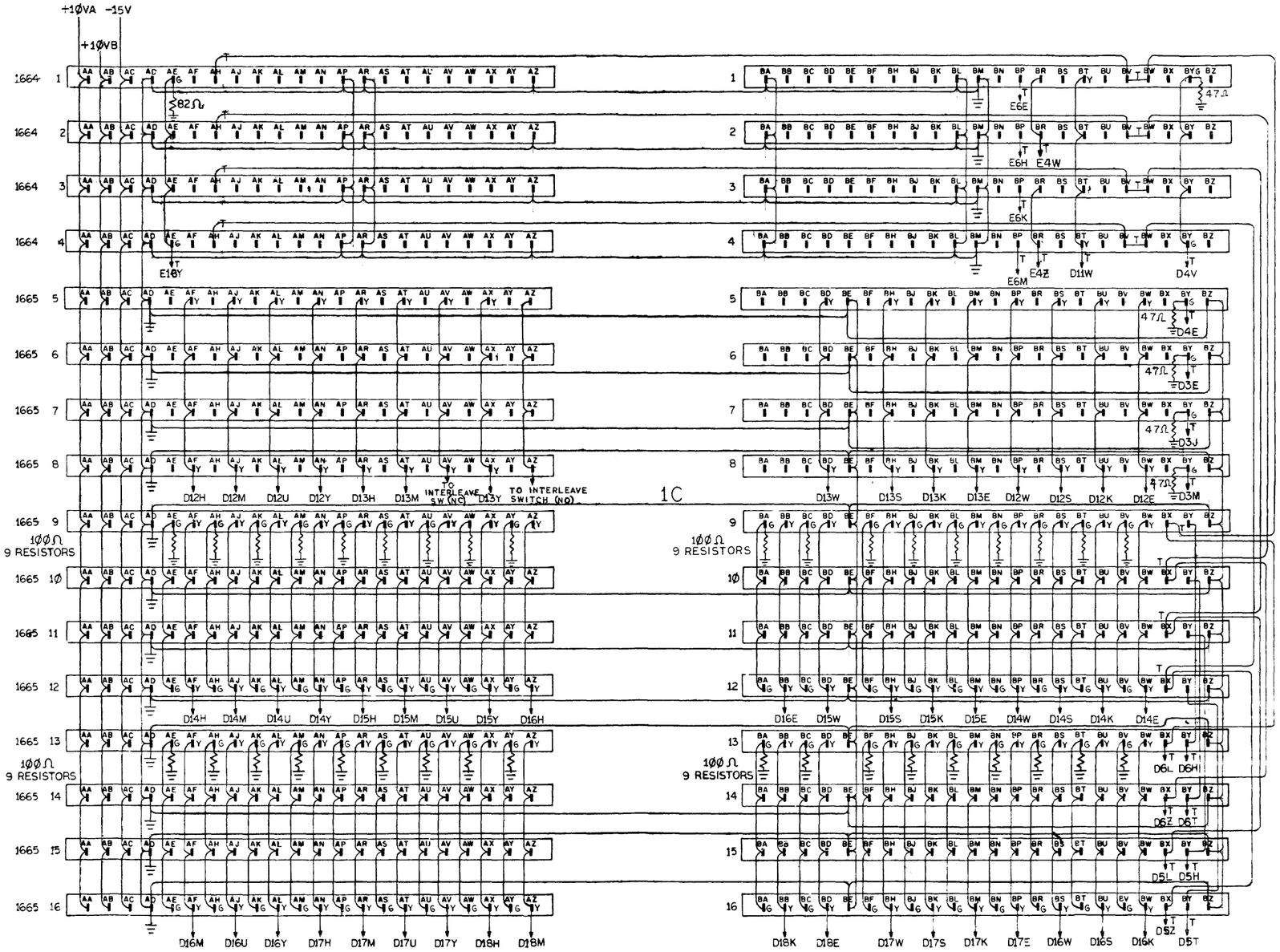


FIGURE 3-5

Figure 3-5 Core Memory Flow (FD-D-161C-0-CMF)



NOTES:

1. 1664 GND PINS AD-AP-AR-AZ-BA-BL-BM.
2. 1665 GND PINS AD-BE-BZ.
3. "T" DENOTES TWP.
4. CONNECTIONS MUST BE MADE IN ORDER SHOWN.

Figure 3-6 CMPC - Core Memory Processor Control (BS-D-161C-0-1BC)

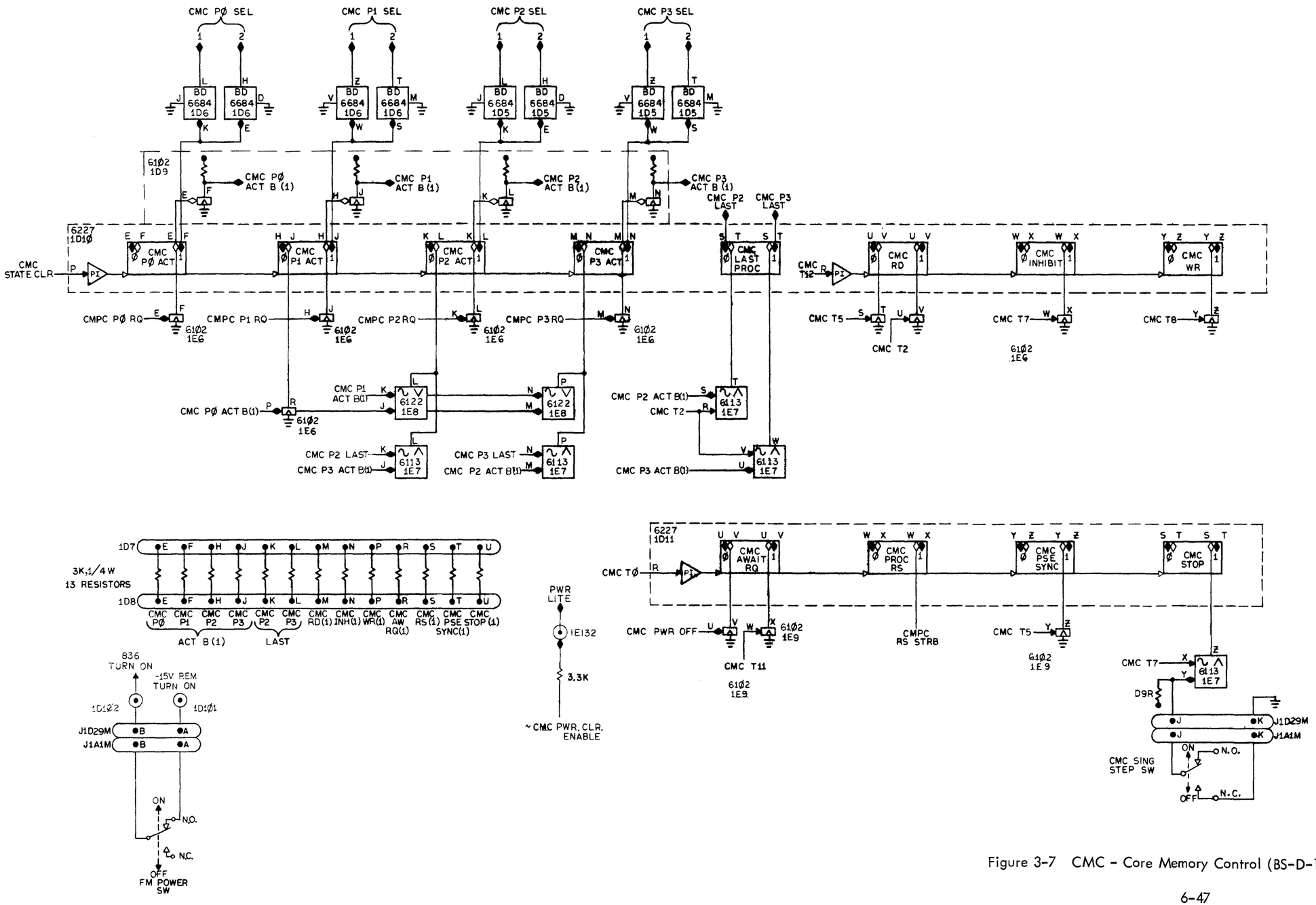


Figure 3-7 CMC - Core Memory Control (BS-D-161C-0-CMC2)

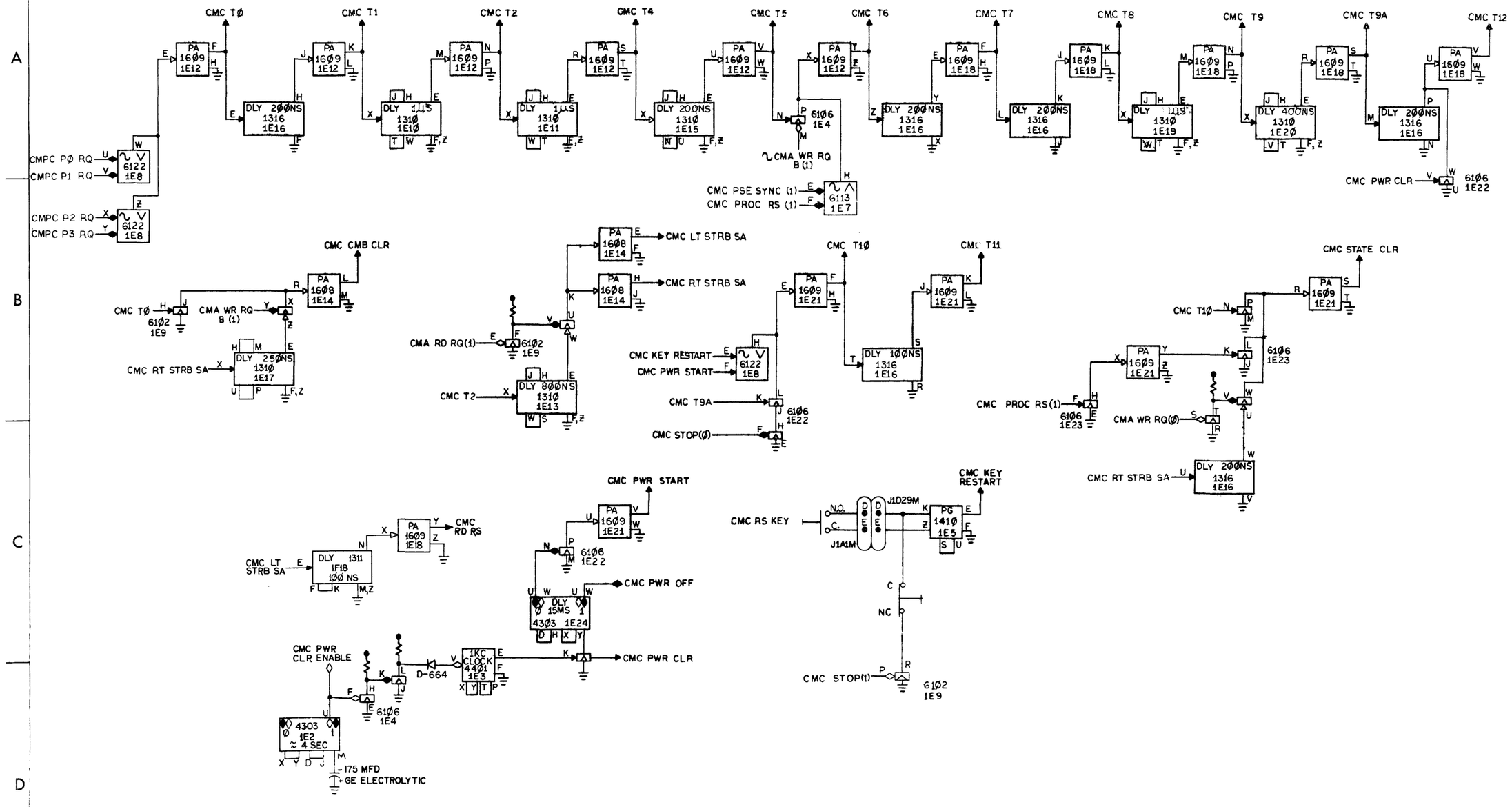


Figure 3-8 CMC - Core Memory Control (BS-D-161C-0-CMC1)

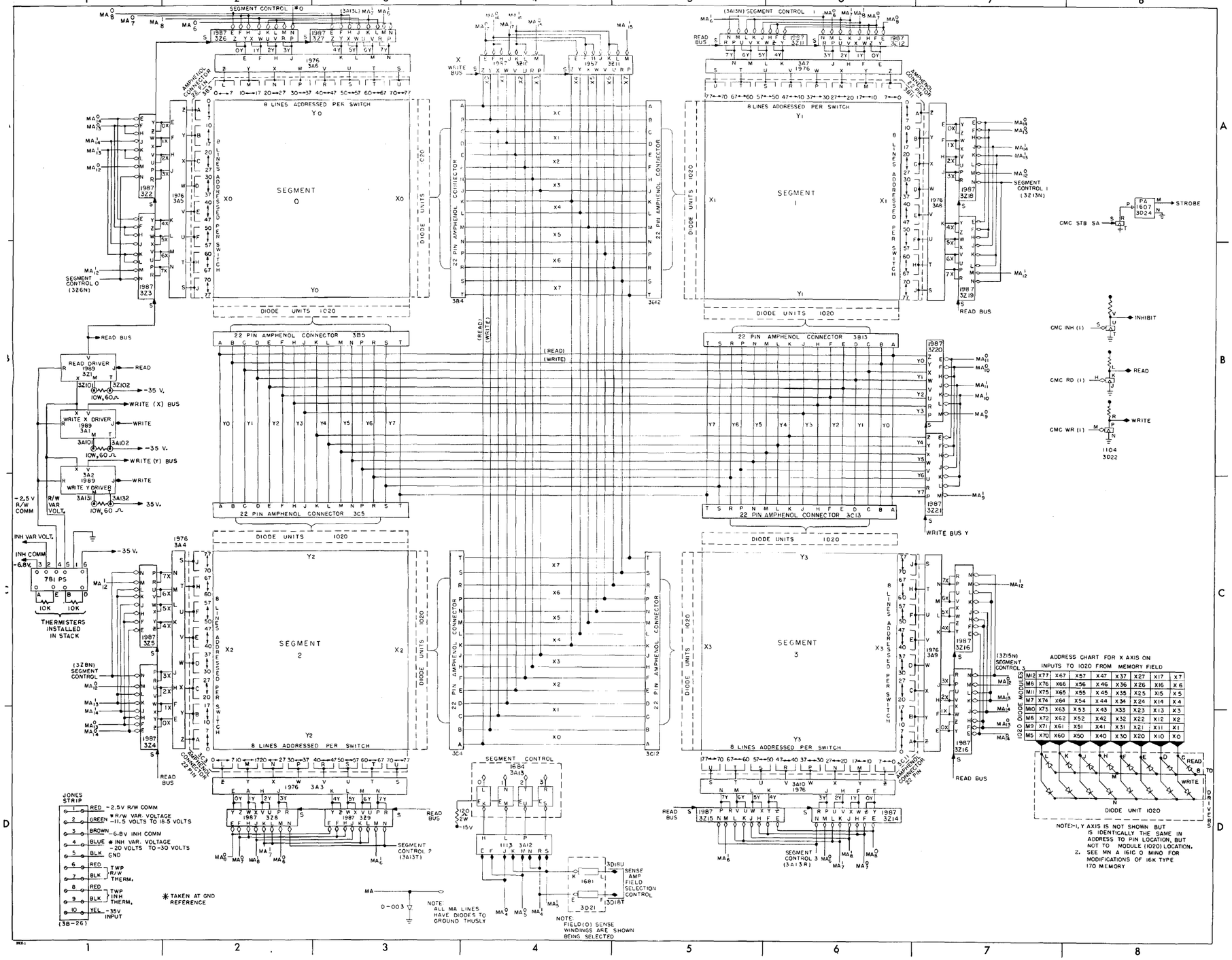


Figure 3-9 RW - Read-Write (BS-E-161C-0-RW)

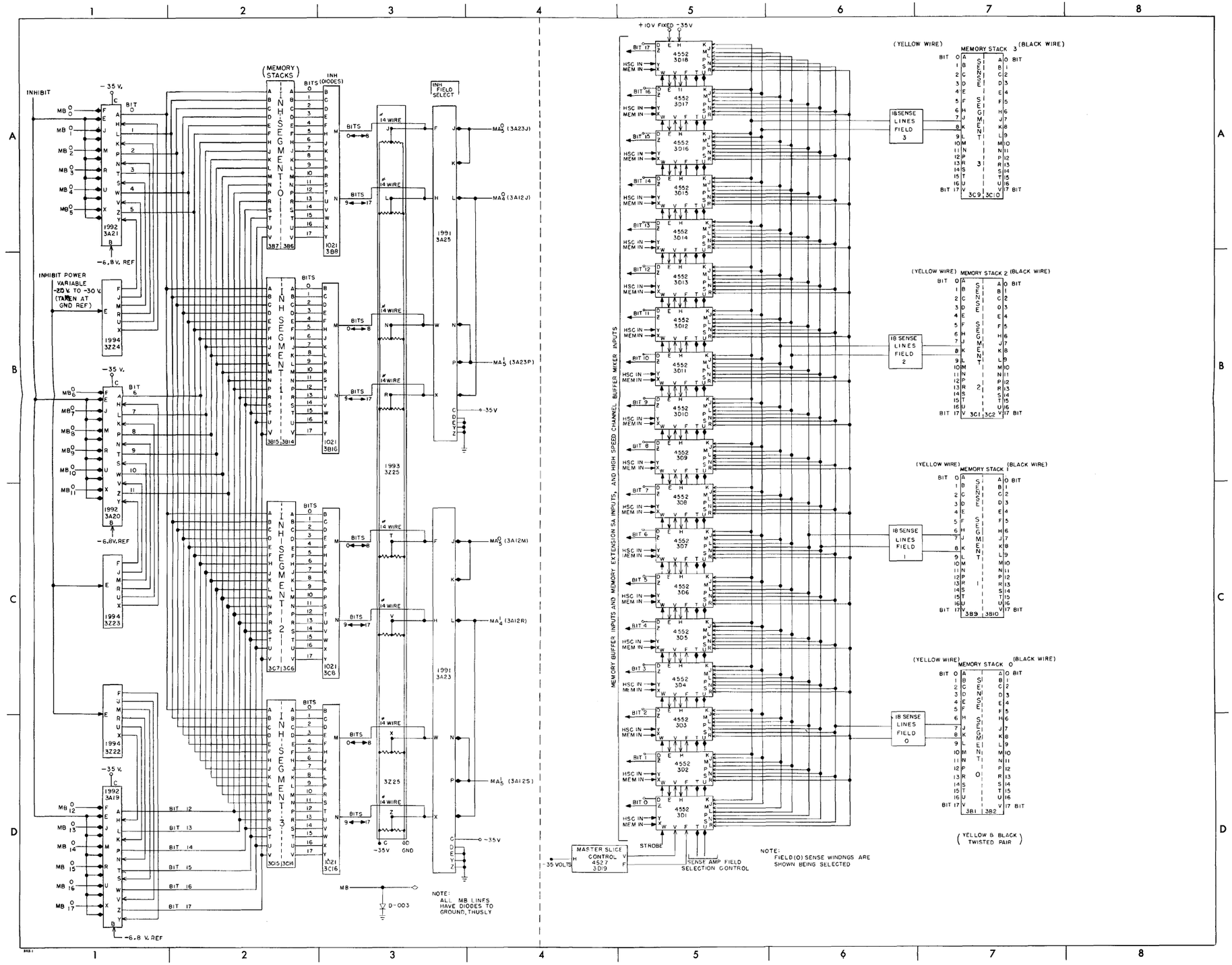
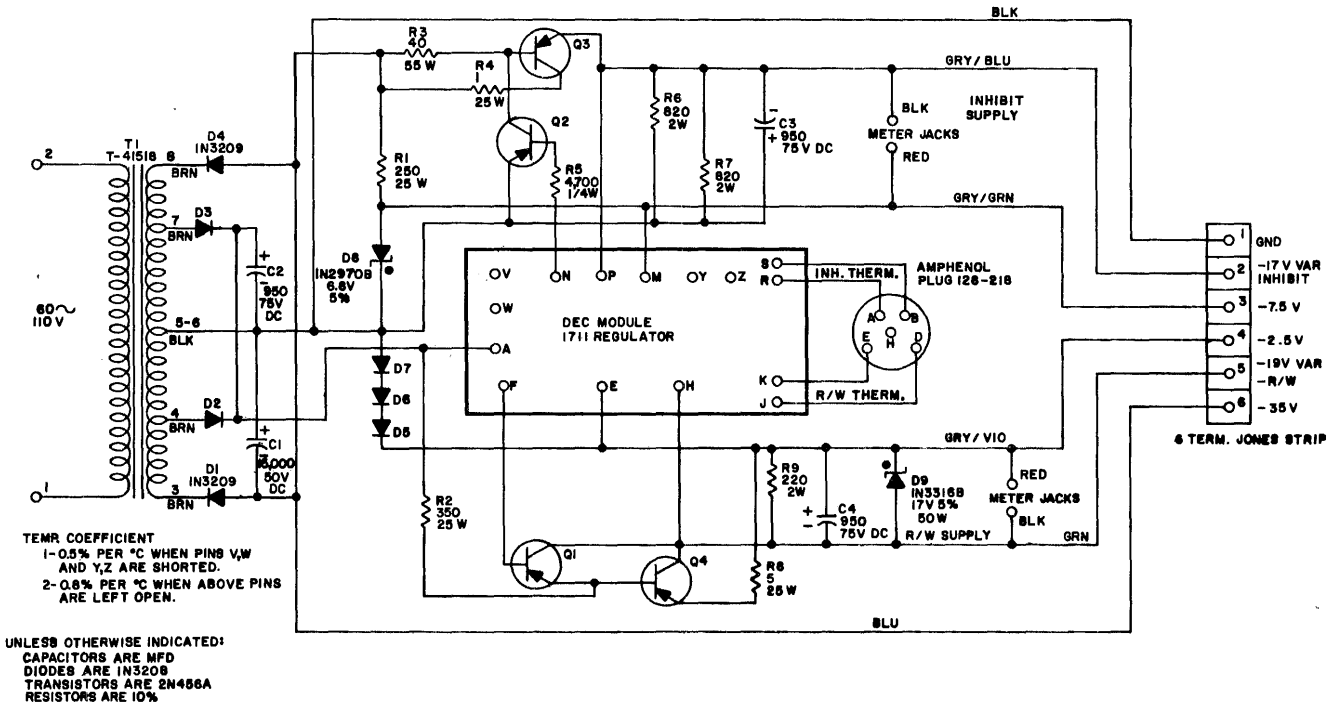
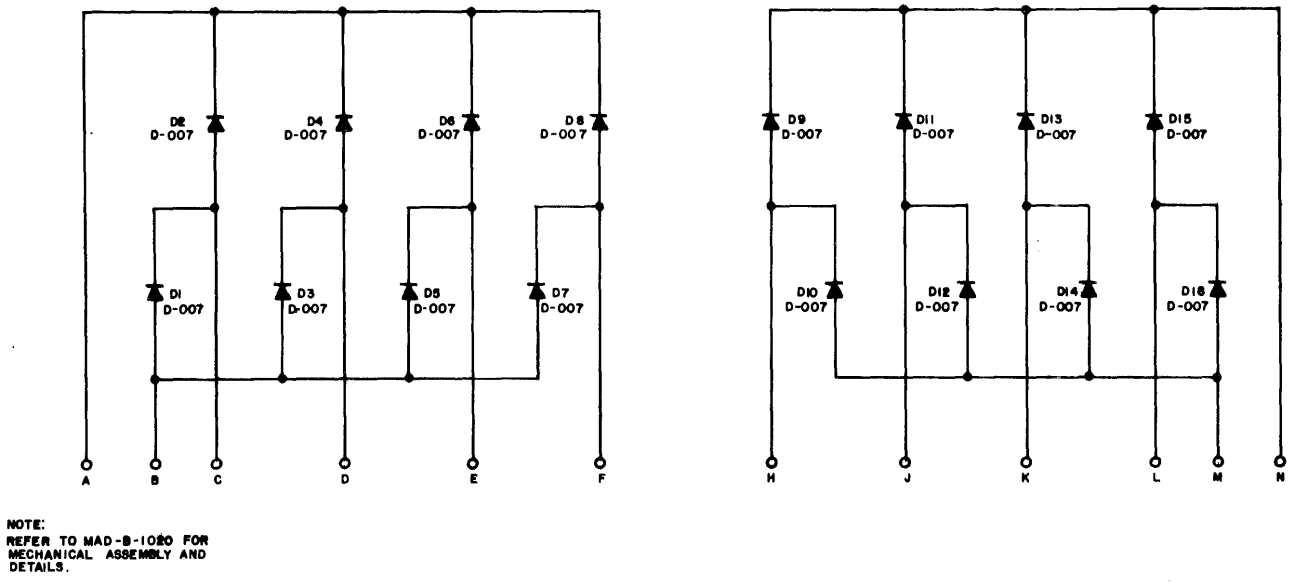


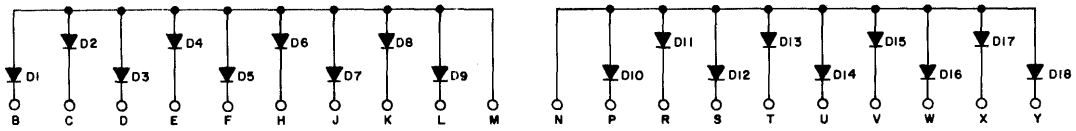
Figure 3-10 SA - Sense and Inhibit (BS-E-161C-0-SAI)



Memory Power Supply 781

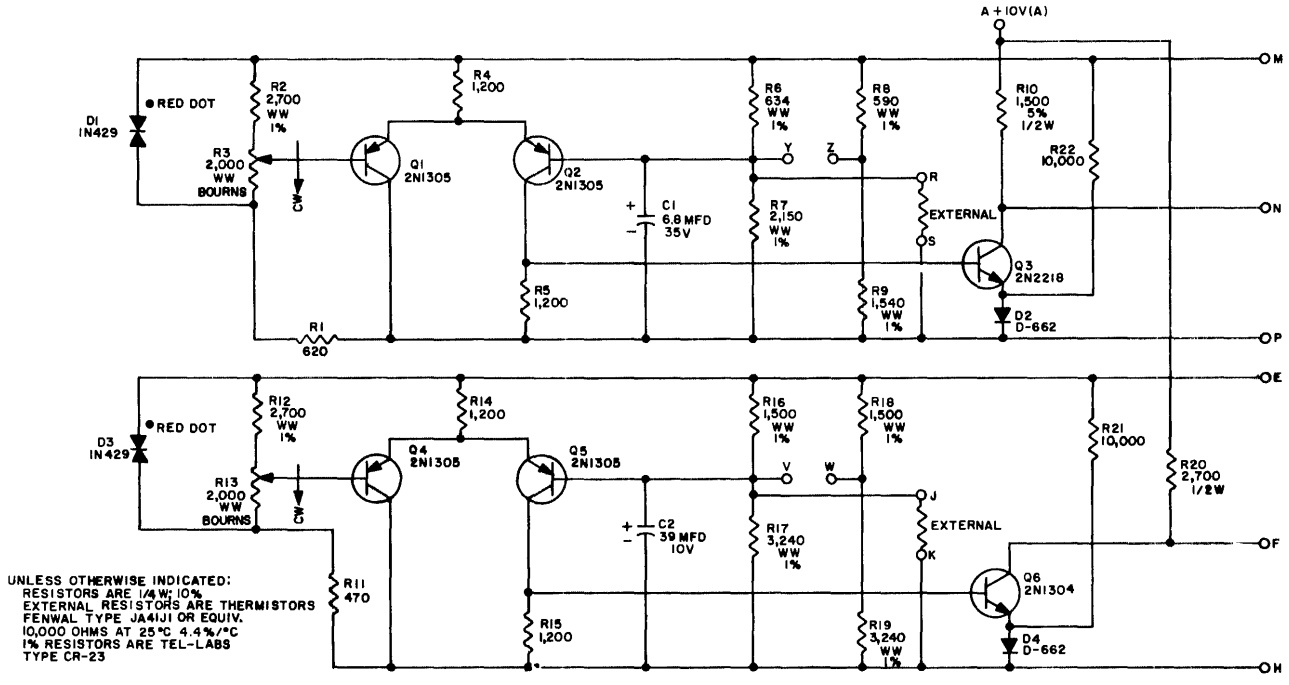


Memory Diode 1020

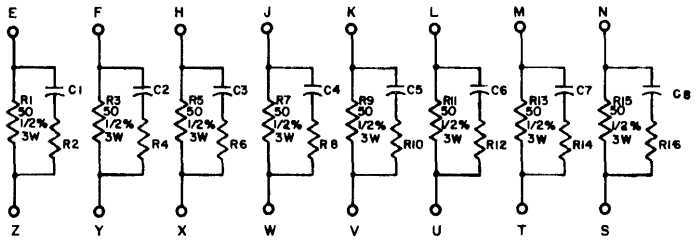


UNLESS OTHERWISE INDICATED:
DIODES ARE D-007

Inhibit Diode 1021

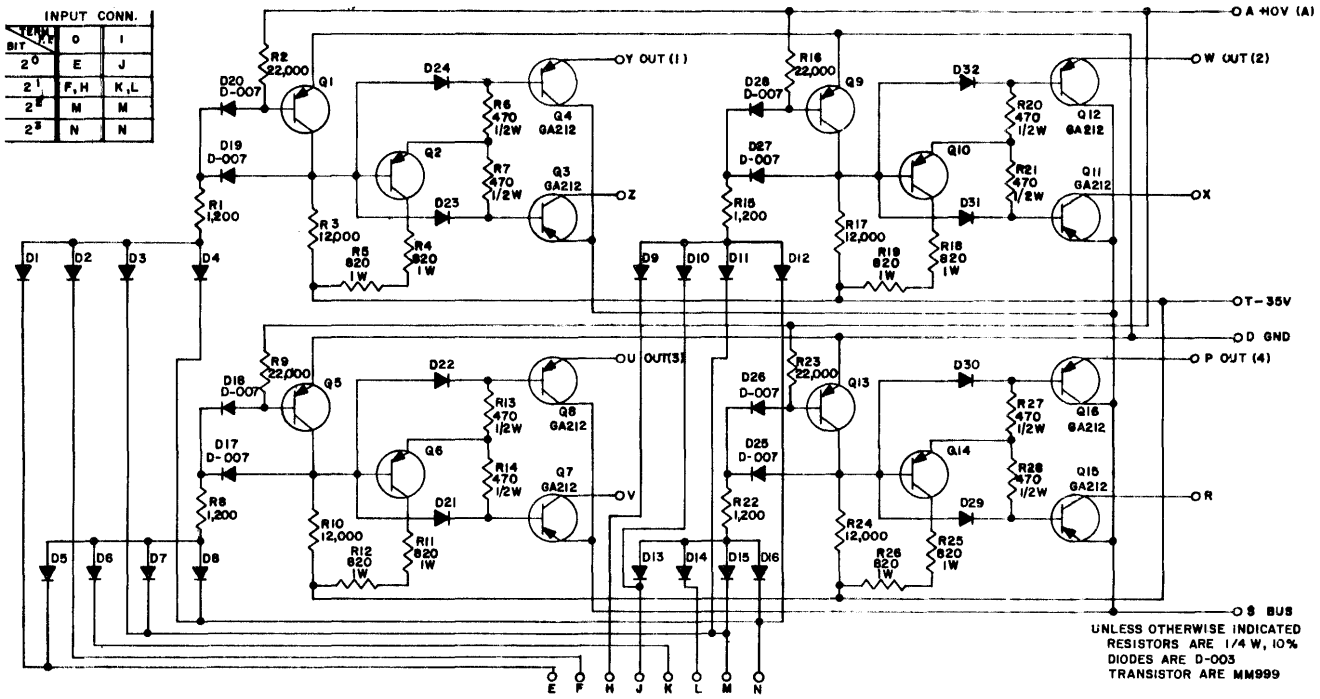


Power Supply Control 1711

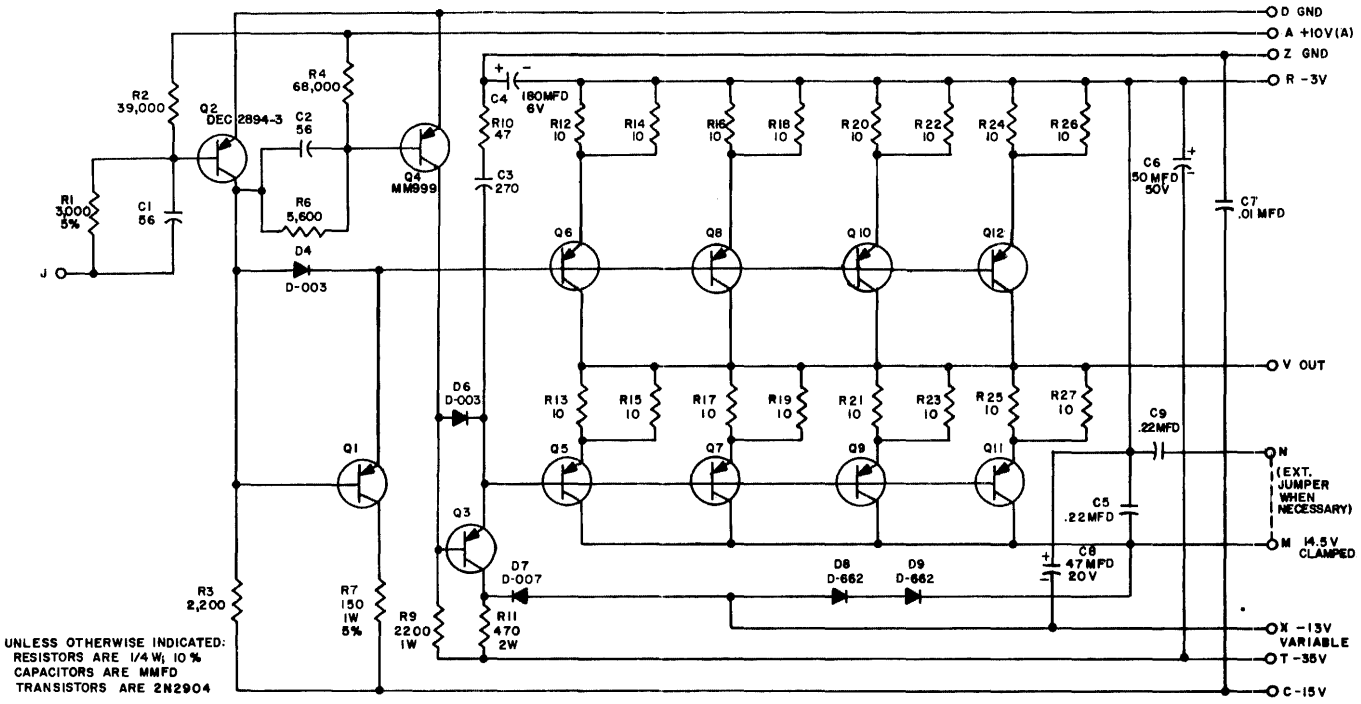


UNLESS OTHERWISE INDICATED:
RESISTORS ARE 47 OHMS 1%
CAPACITORS ARE 4700 MMFD 5%

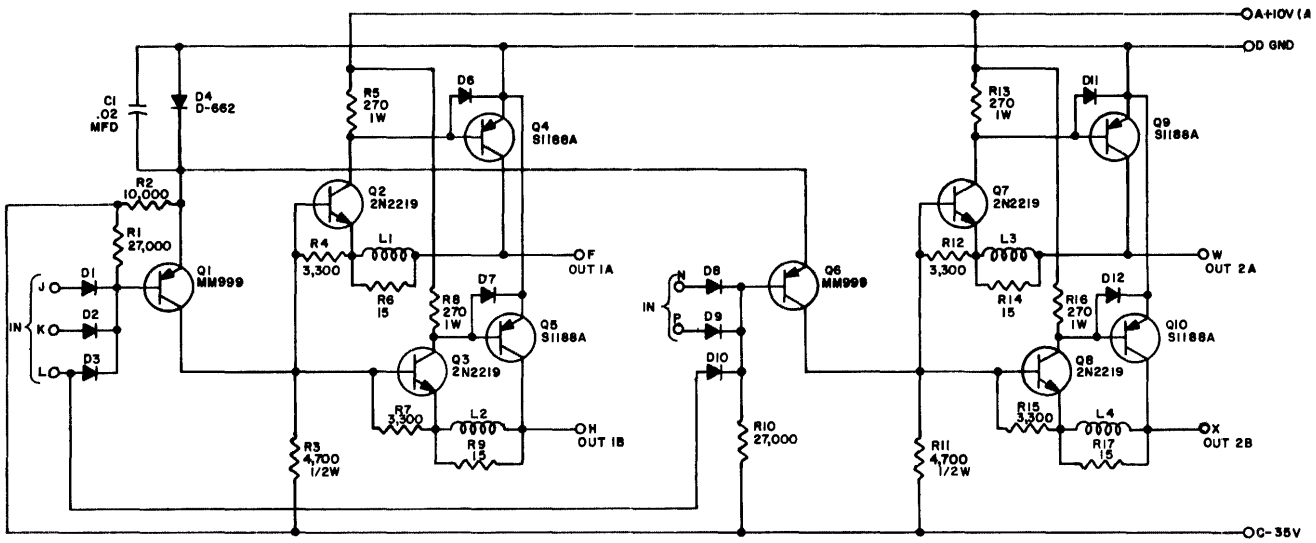
Resistor Board 1976



Read-Write Switch 1987

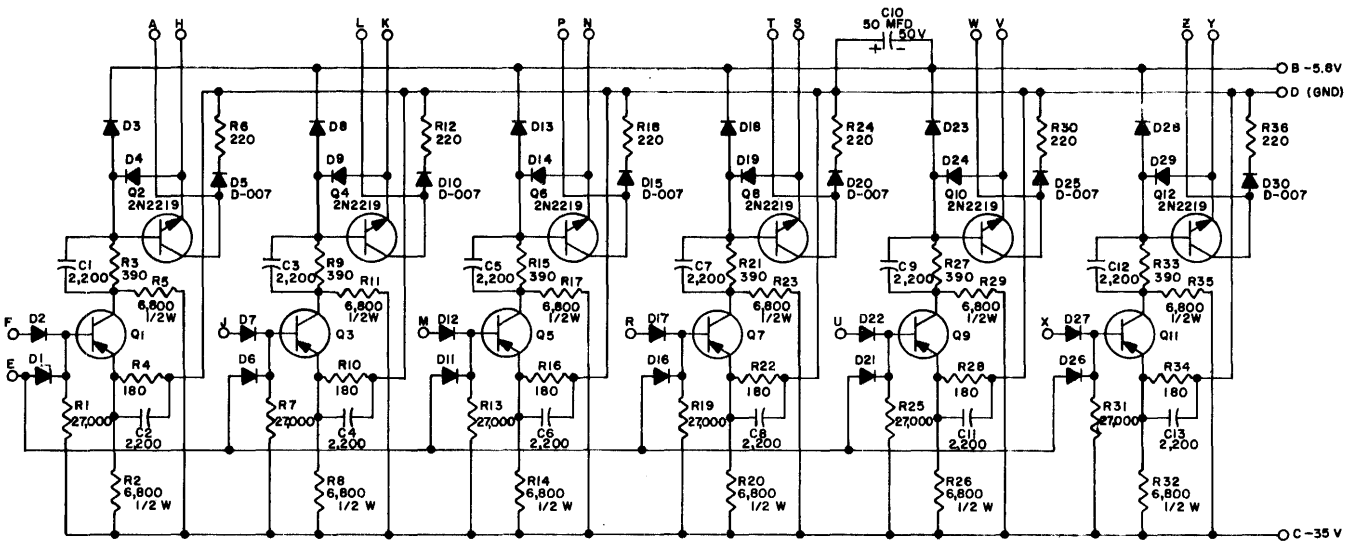


Memory Driver 1989

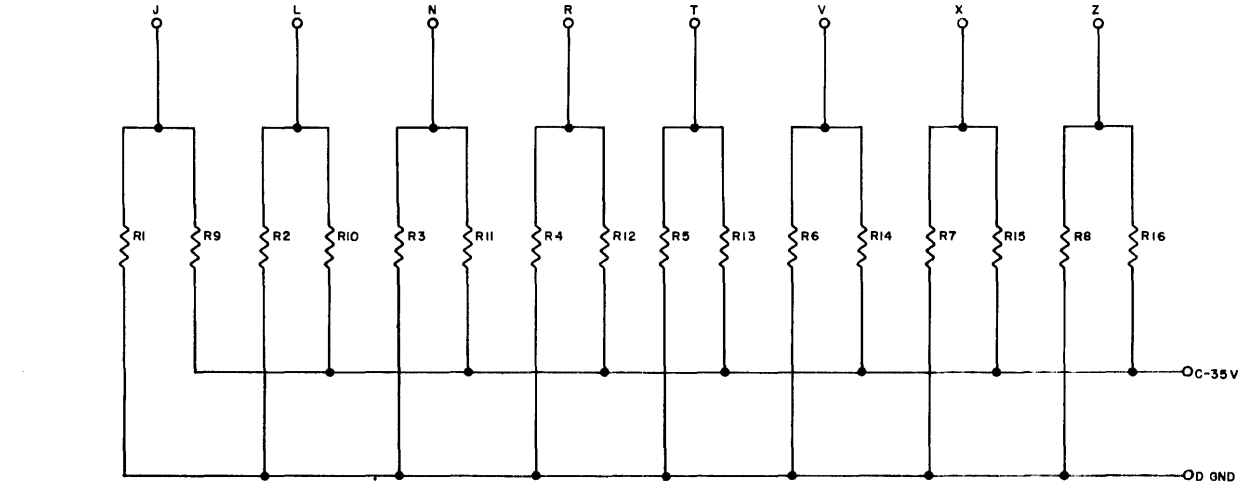


UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W, 10%
L1-L4 ARE ESSEX RFC-M 3.3 μH
DIODES ARE D-003

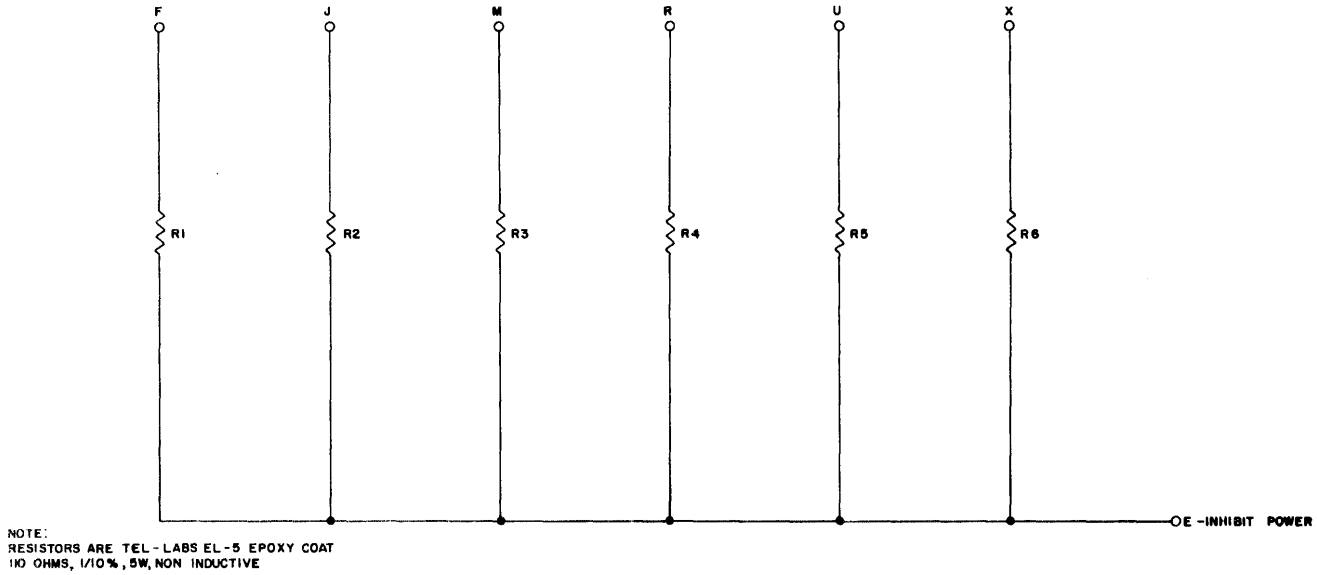
Inhibit Field Select 1991



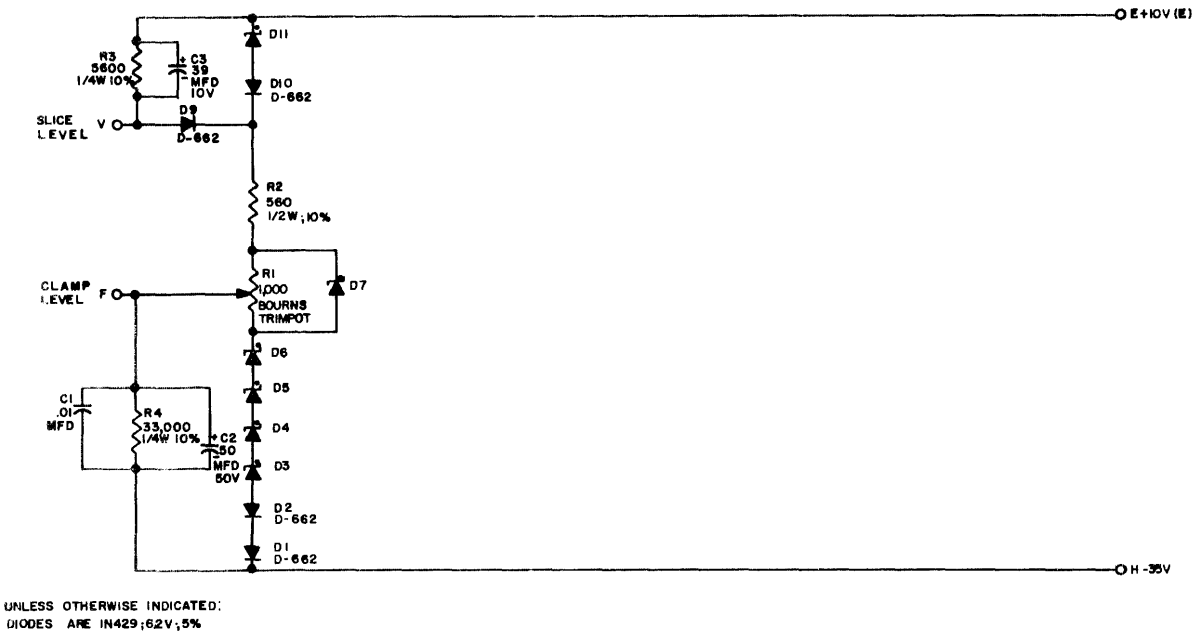
Inhibit Driver 1992



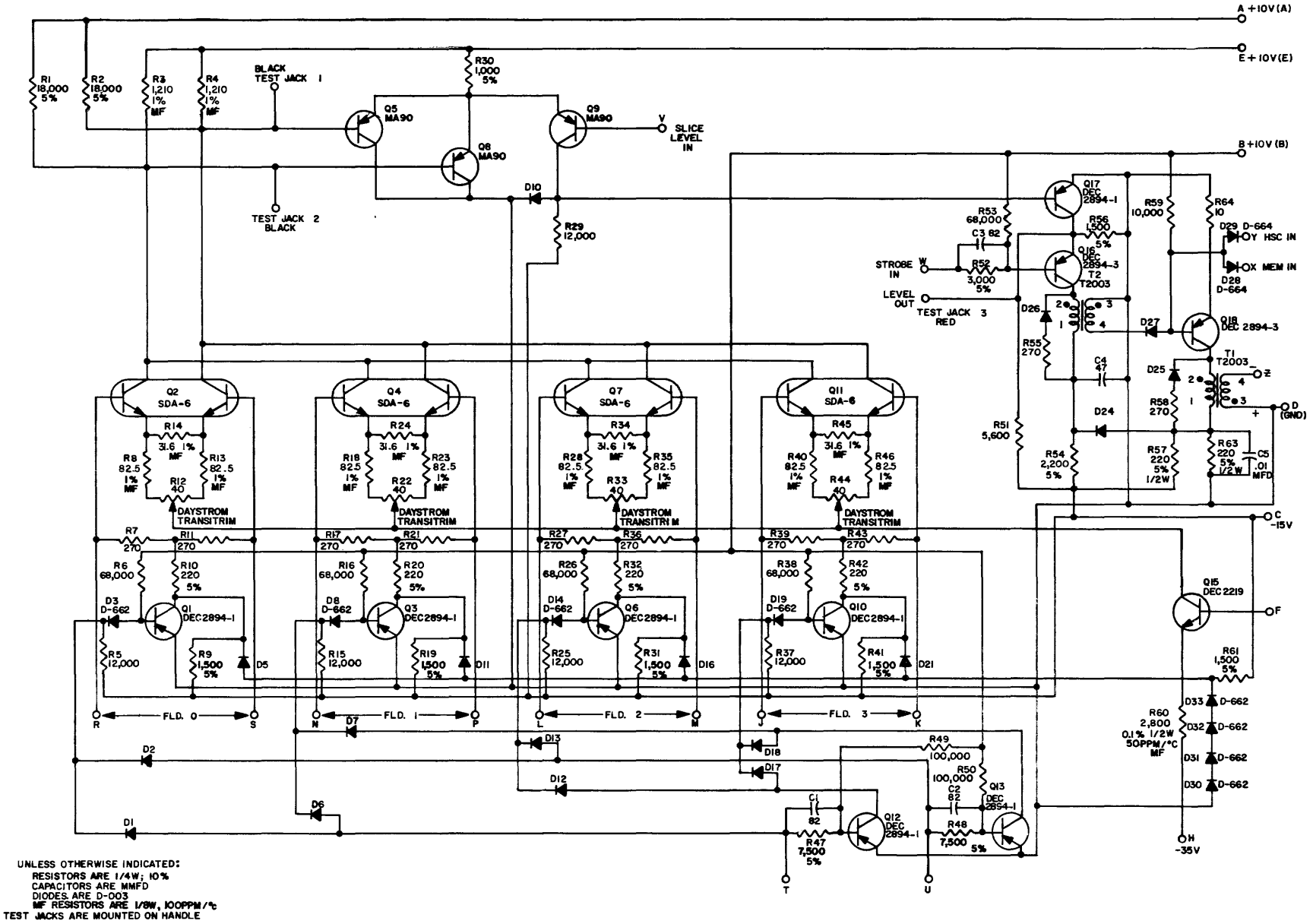
Inhibit Field Select Resistor 1993



Inhibit Driver Resistor 1994



Master Slice Control 4527


A +10V(A)
E +10V(E)
B +10V(B)
BLACK TEST JACK 1
TEST JACK 2 BLACK
SLICE LEVEL IN
STROBE IN
LEVEL OUT
TEST JACK 3 RED
DAYSTROM TRANSITRIM
FLD. 0
FLD. 1
FLD. 2
FLD. 3
Y HSC IN
X MEM IN
C -15V
D (GND)
F
H -35V
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%
CAPACITORS ARE MMFD
DIODES ARE D-003
MF RESISTORS ARE 1/8W, 100PPM/°C
TEST JACKS ARE MOUNTED ON HANDLE

GLOSSARY

Given below are the meanings of all prefix codes and all terms used in the signal names for the fast and core memories. With each prefix code are listed in parentheses the numbers of the figures to which the code applies. Complete signal names are omitted because they are composites of standard terms; their meanings are obvious to anyone who is familiar with the material on signal notation in Chapter 4 of the processor manual.

ACK	Acknowledgment
ACT	Active
ADDR	Address
ADRS	Address
AW	Await
B	Buffered
CLR	Clear
CMA (3-2)	Core memory address
CMB (3-3, 4)	Core memory buffer
CMC (3-7, 8)	Core memory control
CMPC (3-6)	Core memory processor control
COM	Common
CYC	Cycle
FM (2-7, 2-11 to 2-16)	Fast (flip-flop) memory
FMA (2-2)	Fast memory address
FMB (2-4, 5)	Fast memory bus data connections
FMC (2-9, 10)	Fast memory control
FMD (2-3)	Fast memory address decoder
FMPC (2-6)	Fast memory processor control
INH	Inhibit
LT	Left
MAB	Memory address input from the bus
MC	Memory control in the processor
P	Processor
PROC	Processor
PSE	Pause
PWR	Power
RD	Read
REF	Reference
RQ	Request
RS	Restart
RT	Right
RW (3-9)	Read-Write
SA (3-10)	Sense amplifier
SEL	Select
STRB	Strobe
VAR	Variable
WR	Write

EQUIPMENT CORPORATION
MAYNARD, MASSACHUSETTS

5593 PRINTED IN U.S.A. .5-4/66

F-67 (166)

ARITHMETIC PROCESSOR 166
INSTRUCTION MANUAL

VOLUME 1

PDP-6

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

F-67(166)

PDP-6 ARITHMETIC PROCESSOR 166
INSTRUCTION MANUAL

VOLUME 1

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

COPY NO.

PREFACE

This instruction manual is published in two volumes to aid personnel in the operation and maintenance of the Arithmetic Processor Type 166 and four of the more common PDP-6 input-output devices: Paper Tape Reader Type 760, Paper Tape Punch Type 761, Teletype Keyboard-Printer Type 626, and Card Reader Type 461. Maintenance information for the in-out devices is confined primarily to those portions of DEC manufacture; separate manuals for the devices themselves are furnished with the system.

The first three chapters present a general description of the system and its operation. Chapter 1 lists the operating specifications and describes the physical and electrical characteristics of the system. Chapter 2 provides a general description of system organization at the block diagram level, explaining what the system does rather than describing the circuit hardware involved in the various functions. This chapter also describes the number system and instruction formats used in the Type 166 Processor. Chapter 3 explains the use of all controls and indicators on the operator control panels and in-out devices, and outlines basic operating procedures.

The next five chapters present a complete, detailed description of the system logic. Chapter 4, Drawing Conventions and Flow Charts, discusses PDP-6 documentation and describes the symbols and terminology used in the logic drawings and flow charts. This chapter also escorts the reader through the flow charts in sequence, so that he may better understand the flow of operations in the processor, and discusses in detail some sequences that neither appear as coherent hardware units in the logic drawings nor are obvious from the flow charts. The next two chapters in this group describe the hardware for the main control sequence and logical and arithmetic processing; the final two describe the processor interfaces with the memory bus and the in-out bus. Also included in the last chapter are the control units for four common in-out devices. The reader is strongly advised not to embark upon any logic chapter in this or any other PDP-6 system manual without first gaining a thorough understanding of the material presented in Chapter 4.

Chapter 9 contains information useful in maintaining the system, including a discussion of maintenance operation, maintenance programs, and preventive and corrective maintenance.

Following Chapter 9 are appendixes on engineering drawings and spares, a glossary, and several convenient tables. All logic drawings and flow charts referred to in the text are in Volume 2; all other figures are interleaved with the text.

FOREWORD

PDP-6 is a general-purpose, digital computing system consisting of processors, memories, and input-output devices, each of which has independent internal timing. Processors in a system may share memories and input-output equipment; the memories themselves may have different speeds.

A central processor, usually the Type 166 Arithmetic Processor, performs arithmetic and logic operations and governs the movement of information between memory and peripheral devices. The Type 166 includes an executive system that allows a number of programs, each restricted to a definite area in core, to share processor time. The central processor uses two busses for system intercommunication, one to the memory system, the other to its input-output devices. A system may contain any number of central processors, each with a memory bus and an in-out bus. The memory buses permit the memory complex to accommodate several processors—as many as four may address a single memory module. In order to deposit or retrieve information, the processor supplies an address and requests a memory cycle. Upon accepting the request, the addressed memory times its own cycle and furnishes the appropriate response to the processor.

Through the in-out bus the processor controls all information transfers to and from the peripheral equipment. A priority interrupt system in the processor allows a device to signal when it needs service so that the processor is free during the actual input-output time. One of the units that may be connected to the bus is the Type 167, an autonomous drum processor which supplies direct memory access for high-speed devices such as drums, discs, magnetic tape, and displays. In addition to its in-out bus connection to the central processor, the drum processor is itself connected to the memory system via its own memory bus, and has its own smaller scale in-out bus through which it may govern up to three input-output control units. Although the central processor must provide initial conditions and commands, the drum processor then operates independently, so large blocks of information may be transferred between an in-out device and a memory without reducing central processor efficiency.

All PDP-6 memories store words of 36 bits but may be of different sizes and speeds. Core memories usually have core banks of 8192 or 16,384 words. Cycle times for reading from and writing back into memory are typically 2 and 5 μsec, although in each case access time is much shorter: when reading, the processor need wait only until data is available; when writing, only until data is accepted. A fast flip-flop memory, with access time less than 1/2 μsec, is normally used instead of the bottom 16 locations in core.

The instruction format allows the basic instructions to address one of 262,144 locations in memory for an operand, one of 15 index registers for modifying the memory address, and one of 16 accumulators for a second operand. Instruction results may be stored in an accumulator, in memory, or in both. In-out instructions govern the transfer of data in both directions over the in-out bus, the transfer of control information, including priority interrupt channel assignments, to the peripheral equipment, and the gathering of status information from that equipment. In addition to addressing a memory location and an index register, an in-out instruction may address one of 128 devices, two of which are the priority interrupt system and the processor itself.

For further information on the overall system, refer to PDP-6 Programming (DEC publication K-06), which also describes system software and discusses programming for the processor and most in-out devices. Maintenance documentation for the system is provided by a series of manuals. This one discusses system maintenance for the Type 166 Arithmetic Processor and several common in-out devices, others cover the several types of memories that may be used in a PDP-6 memory system, and still others treat the drum equipment, magnetic tape equipment, DECtape, and other in-out devices. A separate circuit manual discusses circuit maintenance and describes most standard circuits including all those used in the equipment described in the present manual and all logic circuits used in the memories. Descriptions of specialized circuits, such as those associated with the core stack, reading and writing on magnetic tape, and the like, are included in the appropriate system manuals.

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

CONTENTS (continued)

CONTENTS (continued)

ILLUSTRATIONS

CHAPTER 1

INTRODUCTION

The Type 166 is a general-purpose central processor that performs all of the arithmetic, logical, executive, and internal data transmission operations in a PDP-6 system. It also controls all transfers of data between memory and peripheral equipment, although in many cases it may provide control merely by supplying system commands and initial conditions to an in-out processor. It contains two bus interfaces, one for connection to memory, the other to the input-output system.

Except for certain control information held permanently in the processor, the state of the processor resides entirely in memory. The only information carried over by the processor from one instruction to the next is the program count, flags, and information for a user mode which allows a number of programs, each restricted to a definite area in core, to share computer time. Besides operating on a stored program, the processor must retrieve all operands for every instruction, and all data and results of computations are stored at the completion of an instruction. Thus the arithmetic registers in the processor contain information only during actual processing and the registers used for address modification are the same as those used for computations within a single instruction. The accumulators, 15 of which double as index registers, actually occupy the bottom 16 memory locations and are usually contained in a fast memory. Most basic instructions have three addresses which select an accumulator, a memory location (which may be another accumulator), and an index register for memory address modification. All instructions may use multiple-level indirect addressing and some may use a single address to call two adjacent accumulators for processing double-length operands. With a single instruction, the processor is capable of performing a full-word or half-word transfer, a block transfer, or the manipulation of a character (byte) of variable size. The processor includes hardware for performing Boolean functions, shift operations, both fixed- and floating-point arithmetic, jumps, logical and arithmetic comparisons, and a variety of modification and testing instructions.

In addition to standard instruction operations, the hardware also includes a program-assignable priority interrupt system through which an external device or an internal condition can interrupt

the normal program sequence; a number of flags that allow checking of various conditions and facilitate double-precision arithmetic; and memory protection and relocation registers that allow an executive routine to assign a specific area in core to each user program.

1.1 OPERATING SPECIFICATIONS

All timing in the Type 166 is completely asynchronous, and all processing is done in parallel except for a few extremely fast serial functions, such as carry propagation in the main arithmetic register and in dc adders that relocate memory addresses. Information handled by the processor has the following characteristics:

Word Length	36 bits
Instruction Format	
Basic	Instruction code, 9 bits Accumulator address, 4 bits Indirect, 1 bit Index register address, 4 bits Memory address, 18 bits
Input-Output	Instruction code, 6 bits Device code, 7 bits Indirect, 1 bit Index register address, 4 bits Memory address, 18 bits
Internal Number System	Binary
Negative Representation	2's complement
Number Format	
Fixed Point	Sign, 1 bit; magnitude, 35 bits
Floating Point	Sign, 1 bit; exponent, 8 bits; fraction, 27 bits

The time required for execution of any particular instruction varies tremendously because of the completely asynchronous operation. The basic operations, such as addition in the arithmetic register or a sequence of shifts controlled by the shift counter, are performed by built-in

hardware subroutines. These are called whenever necessary either from the main instruction sequence or by special sequences such as byte manipulation, block transfer, floating add-subtract, divide, etc., which are in turn entered from the main instruction sequence. Even at the level of individual events, the execution time may vary; for example in the basic addition or subtraction subroutine, the complement function and the parital addition each require 100 nanoseconds but carry propagation, which is serial, depends upon the number of carries needed. Similarly on a larger scale, multiplication and division are performed by a series of additions and subtractions and the time required for such a major sequence depends upon the number of times it must call various subsequences. Most processor control functions involved in the retrieval and setup of instructions, and retrieval and storage of operands take a negligible amount of time when compared to memory access time. Exact instruction execution times may be determined from the flow charts included in Chapter 4. For each memory access, the processor must first check for memory protection and relocation and then wait until the addressed memory is free; the time required for access once the memory is free depends upon the type of memory.

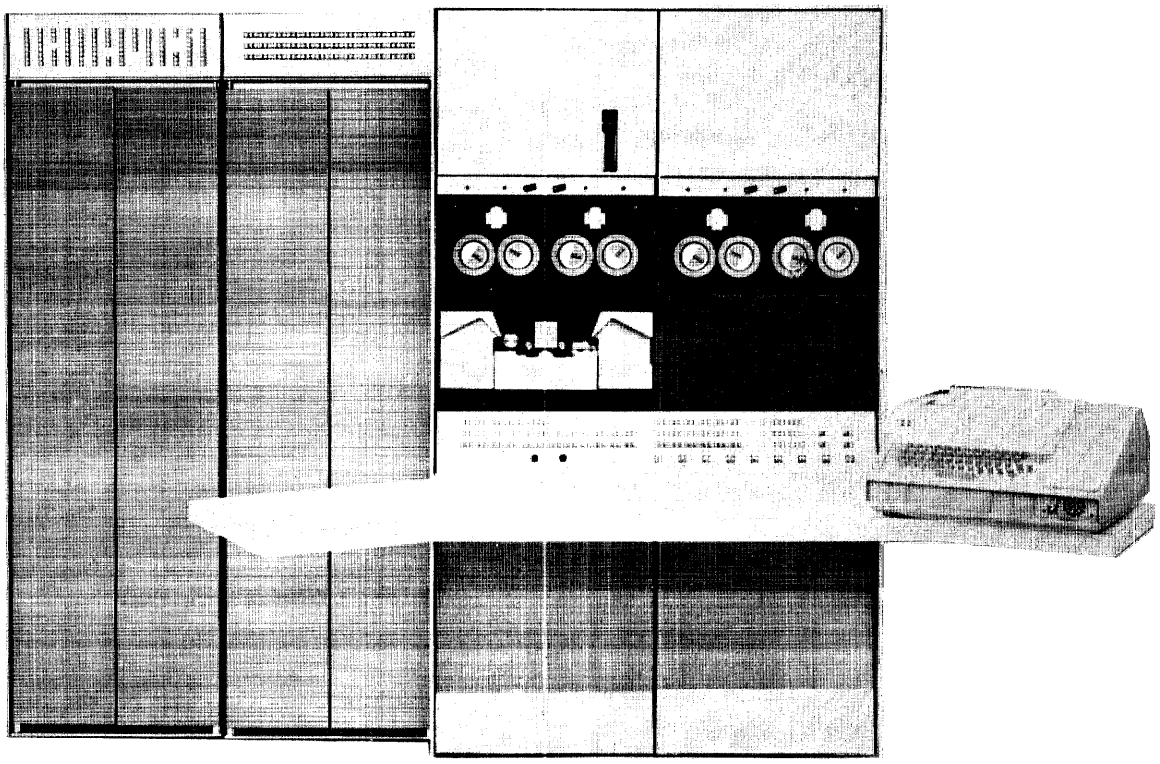
The processor must set up all transfers of control information and data to and from the peripheral equipment; but since a device can signal the processor by means of the priority interrupt system when it requires service, no processor time need be lost in waiting, and processor and peripheral equipment can operate in parallel. Every transfer over the I/O bus does, however, require 2.5 microseconds. The four I/O devices included in this manual have the following operating specifications.

Paper Tape Reader	400 8-bit characters per second
Paper Tape Punch	63.3 8-bit characters per second
Keyboard-Printer	10 8-bit characters per second
Card Reader	200 80-column cards per minute

1.2 PHYSICAL CHARACTERISTICS

Most DEC equipment is housed in steel bays with aluminum control panels. The arithmetic processor with its console uses four such bays bolted together. The front of each bay can accommodate up to twelve 19 inch by 5-1/4 inch panels lettered A to N from top to bottom

(skipping G and I). Bays 1 and 2, which house the bulk of the processor logic, each have an indicator panel at the top with the remainder of the bay occupied by eleven standard logic panels mounted behind double doors. Each mounting panel can hold up to 25 DEC system plug-in modules numbered from left to right when viewed from the front. At the center of bays 3 and 4, which hold the console, is the main operator control panel. Usually, a tape reader is mounted in the left console bay just above the control panel and a paper tape punch at the top (the front panel of the drawer containing the punch has an opening for removing fan-folded tape). At the top of bay 4 behind a metal cover are the marginal check controls and an indicator panel for the standard in-out equipment. The remaining space above the control panel may be used for DECtapes, displays, or other equipment. The space below the console table can hold up to eight logic mounting panels, two of which are used for the arithmetic processor; the remainder may be used to hold the control logic for some of the in-out equipment. Inside the double doors at the back of each bay is an inner plenum door, on which are mounted the required power supplies and power control panels.



Arithmetic Processor Type 166

Physical dimensions are as follows:

Arithmetic Processor

Height	69-1/2 inches
Width	100 inches
Depth	60 inches (75 inches with rear plenum doors open)
Weight	1300 pounds including tape reader and punch

Keyboard-Printer, Teletype Model 35 KSR

Height	38-1/2 inches
Width	20 inches
Depth	24 inches
Weight	151 pounds

Card Reader, Burroughs B122

Height	50 inches
Width	48 inches
Depth	29 inches
Weight	200 pounds

Intake fans at the bottom of every bay cool the logic modules by blowing air out between them. All equipment described in this manual can operate in an ambient temperature range from 50° to 100°F. However, if the installation includes temperature sensitive equipment such as magnetic tape, air conditioning is required. The floor should be capable of supporting approximately 150 pounds per square foot.

1.3 ELECTRICAL CHARACTERISTICS

All PDP-6 equipment uses standard line power at 105 to 125 vac, 60 cycles, single phase. All power cables use Hubbell Twist-Lok connectors; both cable and connector are rated at 30 amperes. The arithmetic processor, console and console-mounted standard in-out equipment together use two lines and two power controls. The main power control is usually a Type 829

or Type 835; it provides ac to all of the power supplies for the processor logic and any in-out control logic mounted below the console operator panel. The dc voltages required by the logic are +10 and -15 volts. Some power supplies provide both, others provide only the negative voltage. In some cases, two -15 volt supplies may be connected in series to provide -30 volts to solenoid drivers for in-out equipment. One -15 volt line turns on a secondary power control (Type 811 or 834) that provides ac to the motors for the reader, punch, and keyboard-printer. For the punch, ac is fed through a Type 823 Power Control (mounted directly on the punch) that allows the processor logic to control application of punch motor power. Another -15 volt signal is applied to external power controls (usually Type 811 or 834) via the in-out bus to turn on the peripheral equipment. Still another -15 volt signal turns on the power controls (usually Type 836) in the memories. This last dc turnon signal is not sent via the memory bus; instead it is included in a small bus that also carries marginal check voltages from a variable power supply located in the console.

Current consumption of the equipment described in this manual is as follows:

Arithmetic Processor, including console and console-mounted in-out logic	25 amperes, 1900 watts Turnon surge, 40 amperes
Tape Reader	1.8 amperes, 150 watts Turnon surge, 2.8 amperes
Tape Punch	1.85 amperes, 65 watts Turnon surge, 9 amperes
Keyboard-Printer	2.6 amperes, 140 watts Turnon surge, 7 amperes
Card Reader	1.5 amperes, 145 watts Turnon surge, 7 amperes

All PDP-6 logic is solid state; transistors and diodes operate on static logic levels of 0 and -3 vdc (tolerances are 0 to -.3 volts and -2.5 to -3.5 volts). Most logic modules include an internal supply to derive the negative logic level from the -15 volt input. PDP-6 logic uses pulse timing almost exclusively. Pulses are of either polarity depending upon gate input requirements. Pulse amplitude is 2.5 volts from ground with tolerances of +2.3 to +3.0 volts and -2.3 to -3.5 volts. Pulses at inverter outputs may be from ground to -3 volts or vice versa. Pulse widths may be 1 microsecond or 400, 70, 40, or 25 nanoseconds depending upon module type and application. Occasionally, an input may be triggered by a level transition instead of a pulse.

CHAPTER 2

SYSTEM FUNCTION

The logical configuration of the Arithmetic Processor Type 166 is shown in Figure 2-1. Large blocks at the top and bottom represent the buses that connect the processor to the input-output equipment and the memory; the figure shows all connections to these buses, both data and control (each connection is labeled with the number of physical lines required). Between the buses is a block diagram of the processor showing all registers, with transfers among them represented by lines connecting the register blocks. Each block is labeled with both the name of the register and the number of bits. The registers vary considerably in size. Data registers have 36 bits, those that handle only addresses have 18. Registers that handle floating-point exponents have 9 bits; those that control memory protection and relocation have 8; 7-bit registers govern the requesting and granting of program sequence breaks through the priority interrupt system. The figure does not show the control lines within the processor, but all control pulses for each register are written beside the corresponding block. For an explanation of signal names, refer to the discussion of signal notation in Chapter 4.

The heart of the arithmetic processor is a set of three full-size registers which handle all data transfers and in which are performed all logical and arithmetic operations. These are arithmetic register AR, multiplier-quotient register MQ, and memory buffer MB. All transfers between processor and memory are made through MB, transfers between processor and peripheral equipment are made via AR. At the console, the operator may communicate with the system through a register of data switches for sending information in via AR and a register of memory indicators for displaying memory words via MB. MB takes part in all data transfers, but in logical and arithmetic operations it usually plays a passive role by holding an operand which is combined with the operand in AR, the result appearing in AR. MQ serves primarily as an extension of AR for handling double-length operands.

The processor performs a program by executing instructions retrieved from consecutive memory locations as counted by the program counter PC, although the program may change its own sequence by changing the address in PC. To gain access to memory for retrieval or storage,

the processor requests a memory cycle and supplies an address over the bus from memory address register MA. This register also serves as a control link to the operator in that the system receives addresses via MA from the address switch register on the console. When a word is retrieved at MB as an instruction, its left half passes to the instruction register IR which controls further retrieval of index registers and accumulators, and which is decoded to govern the actual execution of the instruction. The right half in MB is the memory operand address, which may be modified in AR by the contents of an index register.

The operands are brought from memory to AR, MB, and sometimes MQ, for whatever operations are necessary for the execution of the instruction. In some of the more complicated operations, these full-size registers are aided by the shift counter SC and the floating-exponent register FE. SC is used for subsidiary computations such as the calculation of the exponent in floating-point arithmetic, and it also controls the count of any operation performed by the repetition of basic steps in the three main registers. FE is used only for temporary storage of preliminary results while SC is controlling the remainder of the operation.

Besides the registers that enter into the regular execution of the program and its instructions, the processor contains an executive system and a priority interrupt system. The first contains two registers for memory protection and relocation. When the processor is in executive mode, all instructions and all memory are available to the program. In user mode, a number of programs share processor time with each program restricted to a specific area in core, and certain instructions are illegal. All programs are written using the lower addresses but these are not supplied directly to memory from MA. In requests for memory access, the address in MA is compared with the contents of the memory protection register PR. The number in this register defines the size of the block available to the program and prevents it from addressing any location outside its assigned area. The address is then changed to one within the assigned area by adding a constant contained in the relocation register RLR to the address in MA.

The priority interrupt system allows peripheral devices and certain conditions internal to the processor to interrupt the normal program sequence. There are seven interrupt channels through which sequence breaks are allowed on a priority basis as governed by three control registers. The first register allows the program to turn individual channels on and off; the second synchronizes break requests to internal processor timing and assigns the break to the highest

priority channel that has been recognized; the third holds the break and prevents further interruption by lower priority channels. A break is executed by performing the instruction in a particular memory location associated with each channel. The assignment of channels to devices is entirely under program control; the program may assign several devices to a single channel or give a device no assignment. One of the devices to which the program may assign a channel is the processor itself. For this purpose the processor has an I/O interface containing a number of flags that allow internal conditions to interrupt the sequence; the flags may be sensed and controlled by the program. Through this interface, the processor may also bring information in from the console DATA switches or supply memory protection and relocation information to the executive system.

Timing for all operations in the processor is supplied by asynchronous pulse chains. Processor operation is initiated by means of a special key cycle that supplies timing for events associated with operator intervention at the console and provides entry into the main sequence. When the processor is running, timing is supplied by the main sequence which is repeated for each instruction. The main sequence uses a hierarchy of other sequences—built-in hardware subroutines—which can be called directly by the main sequence or by any sequence of higher rank. Thus the processor operates using many levels of nested sequences; each sequence stops upon calling a lower ranked sequence and restarts upon return from it (although the restart need not be at the point of departure). For example, in a block transfer, the main sequence calls the block transfer subroutine which in turn calls others to perform the necessary arithmetic and obtain memory access.

2.1 PROGRAMMING

The first 16 locations in memory function as accumulators, index registers, or ordinary memory locations. Their particular functions are determined entirely by the processor under program control, but they differ from the remainder of the memory system only in that they are usually contained in a fast memory. All 16 locations may be used as accumulators or ordinary memory, but only locations 1 to 17 may be addressed as index registers because a zero index register address specifies no indexing. Since ordinary memory addresses are 18 bits, only the information contained in the right half of an index register is actually used for address modification.

In systems that include a fast memory, it replaces the first 16 core locations (which normally hold a readin loader) for normal processor operations—operations in the reading area can be initiated only from the console; and once an instruction has been taken from outside this area, it again becomes inaccessible to the program. In all systems locations 40 and 41 are used for programmed operators, 42 to 57 are used by the priority interrupt system—a programmer should be wary of using these locations for other purposes.

The logic descriptions contained in this manual assume that the reader is completely familiar with the processor instructions, all of which are described in detail in PDP-6 Programming (K-06). That manual describes the instructions in terms of elements available to the program, i.e., by their effect on accumulators, memory locations, flags, and control registers. For convenience Table 4-1 lists the mnemonic and octal codes for all instructions. The remainder of this section describes the Type 166 number system, instruction format, and flags.

a Number System

For arithmetic computations the hardware is capable of handling numbers in two formats, fixed point and floating point. Both formats use the full 36-bit word; bits are numbered 0 to 35 from left to right. In all numbers, bit 0 represents the sign, 0 for positive, 1 for negative. In floating point, bits 1-8 represent an exponent, bits 9-35 represent a fraction. In fixed point, bits 1-35 represent magnitude, which is usually interpreted as a full fraction with the binary point between sign and magnitude or as an integer with the binary point to the right of bit 35. Of course, the assumed position of the point has no effect on the processor and a program may adopt any consistent point convention. However, the fixed-point hardware does include special provisions to facilitate handling of integers in multiplication and division because these operations make use of double-length numbers.

In ordinary arithmetic, the negative of a number is usually formed merely by changing the sign. This notation is inconvenient for a machine so hardware arithmetic represents negatives by 1's and 2's complements. If x is an n-digit binary number, its 2's complement is $2^n - x$ and its 1's complement is $(2^n - 1) - x$ or equivalently $(2^n - x) - 1$. Subtracting a number from $2^n - 1$ (i.e., from all ones) is precisely equivalent to performing the logical complement, i.e., changing all zeros to ones and all ones to zeros. Therefore, to form the 1's complement,

the processor uses the logical complement—usually referred to merely as the complement—and to form the 2's complement it complements and adds one to the result.

In 1's complement notation, one can read a negative number by attaching significance to the zeros instead of the ones. For 2's complement notation, this simple interpretation is not possible because adding one to the logical complement changes at least the final bit and in fact changes bits as far as the carry propagates. Thus in 2's complement notation, one can read a negative number by attaching significance to the rightmost 1 and attaching significance to the zeros to the left of it. A 2's complement system has the following characteristics. A number all ones represents -1. All even numbers both positive and negative end in 0. In a negative integer, ones may be discarded at the left. In a negative fraction, zeros may be discarded at the right; as long as only zeros are discarded, the number remains in 2's complement form because it still has a 1 which possesses significance. However if a portion including the rightmost 1 is discarded, the remaining part of the fraction is now a 1's complement.

In the Type 166, the 2's complement is used to represent negatives for both fixed and floating numbers. In a positive fixed-point number, the sign bit is 0 and bits 1-35 represent magnitude in normal binary fashion. In a negative, the sign is 1 and the remainder of the word contains the 2's complement of the magnitude of the corresponding positive number. Since 0 is considered one of the positive numbers, the magnitude of the largest positive number is one less than the magnitude of the largest negative number. Fixed-point integers thus have a range from -2^{35} to $2^{35} - 1$; for fractions, the range is -1 to $1 - 2^{-35}$.

The floating-point hardware interprets a computer word as containing an 8-bit exponent and a 27-bit fraction. For a positive number, the sign is 0, as before; but the contents of bits 9-35 are now interpreted only as a binary fraction and the contents of bits 1-8 are interpreted as an integral exponent in excess 128 (200_8) code, i.e., exponents from -128 to +127 are represented by the binary equivalents of 0 to 255. Floating-point zero and negatives are represented in exactly the same way as fixed point: zero by a word containing all zeros, a negative by the 2's complement. The negative thus has 1 for its sign and the 2's complement of the fraction, but since every fraction must contain a 1 unless the entire number is 0 (see below), it has the 1's complement of the exponent code in bits 1-8. Since the exponent is in excess 128 code, an actual exponent x is represented in a positive number by x + 128, in

a negative number by 127 - x. The program, however, need not concern itself with these representations because the hardware compensates automatically. For example, for the instruction that scales the exponent without affecting the fraction, the hardware interprets the scale factor in standard 2's complement form but produces the correct 1's complement result for the exponent.

In all floating-point operations, the hardware assumes that all nonzero operands are normalized and always normalizes a nonzero result. Floating-point numbers are considered to be normalized if the magnitude of the fraction is greater than or equal to 1/2 and less than 1. The test for normalization is thus that either the sign bit differs from bit 9 or bits 9-35 contain $400\,000\,000_8$, the latter being required for the special case of the fraction -1/2, in which bits 9 and 0 are equal. Floating-point numbers thus have a fractional range in magnitude from 1/2 to $1-2^{-27}$ and an exponent range of -128 to +127. Note that the signed fractional part -1 (i.e., a 1 in bit 0 and all zeros in bits 9-35) satisfies the test for normalization but the hardware always changes it to -1/2 and adjusts the exponent appropriately. The hardware may not give the correct result if the program supplies an operand that is not normalized or that has a zero fraction with a nonzero exponent.

The characteristics of 2's complement notation require additional precautions in floating-point operations and fixed-point fractional multiplication because these have double-length results. The programmer must remember that discarding the low-order part of a double-length negative leaves the high-order part in correct 2's complement form only if the low-order part is null. In floating point, the programmer may request rounding, which automatically restores the high-order part to 2's complement form if it is negative.

b Instruction Format

All but the input-output instructions and programmed operators use a basic format with bit assignment as follows:

0-8		Instruction code
9-12		Accumulator address
13	I	Indirect bit
14-17	X	Index register address
18-35	Y	Memory address

Bits 0-8 determine which operations are executed for the instruction. Bits 9-12 and 14-17 each address the first 16 memory locations which serve as accumulators and index registers. On some occasions, bits 9-12 are used for control purposes instead of addressing an accumulator, for example to address flags. The effective address E of an instruction depends on the values of I, X, and Y. The contents of index register X (zero X specifies no indexing) are added to Y to produce an address. If I is 0, this address is used as the effective address; if I is 1, this address is indirect and is used to retrieve another word. The new word is processed in exactly the same manner as above, i.e., X and Y are used to determine the effective address if I is 0; otherwise, they are used to retrieve another word. The process continues until a word is found in which I is 0. This calculation using I, X, and Y is carried out for all instructions even when E is to be used as an operand or control information instead of a memory address.

IOT instructions (designated by three ones in bits 0-2) have the following bit assignment:

0-2		111
3-9		Device code
10-12		Instruction code
13	I	Indirect bit
14-17	X	Index register address
18-35	Y	Memory address

Bits 3-9 address an I/O device out of a possible 128, bits 10-12 specify one of eight IOT instructions; the processor and the priority interrupt system are considered devices. As in the basic format, I, X, and Y are used to calculate E, which is used as an address in some cases, as control information in others.

A programmed operator is designated by three zeros in bits 0-2. Whenever such an operator appears in the program, the processor calculates an effective address from bits 13-35 of the instruction word in the usual manner but it does no further decoding; instead it stores the contents of the instruction register in the left half of location 40 and the calculated effective address in the right half, and then executes the instruction contained in location 41 (which is usually a JSR to an appropriate subroutine).

c Program Flags

The processor contains a number of flags that may be sensed by the program. Flags that are set automatically, e.g., by error conditions, usually cannot be set by the program; whereas, flags that allow the program to enable specific operations can always be both set and cleared. Some flags are governed primarily by jump instructions but most are contained in the processor I/O interface and are governed by IOT instructions. Any flag listed as being able to cause a priority interrupt does so on the channel assigned to the processor provided the priority interrupt system is active.

AR CRY0, AR CRY1	These flags are set by carries from the corresponding bits in AR. They are useful primarily for double-precision arithmetic and in correcting a result that has overflowed.
AROV AROV ENABLE	The overflow flag may be set by arithmetic operations in a variety of instructions. It indicates a loss of information, an incorrect result of a computation, or failure of the processor to perform a computation. Setting OV causes a priority interrupt if the enable flag has been set by the program.
PC CHG PC CHG ENABLE	The PC change flag is set when the program sequence is changed by a skip or jump instruction. Setting PC CHG causes a priority interrupt if the enable flag has been set by the program.
PDL OV	The flag is set and triggers a priority interrupt when a pushdown or pullout instruction has gone outside of the core area assigned to the pushdown list.
NON EXIST MEM	This flag is set and triggers a priority interrupt when the memory system fails to respond to a memory request.
CLOCK CLOCK ENABLE	The clock flag is set every sixtieth of a second by a signal from the main power control. It causes a priority interrupt if the enable flag has been set by the program.

USER

As a flag, the sole function of USER is to indicate to the executive routine whether a user program was interrupted, either by a priority interrupt or UUO (the executive routine must service all priority interrupts) or by the trapping of an illegal instruction (instructions that are illegal during a user program are a JRST that attempts to dismiss an interrupt or halt the computer, and any IOT). The execution of a JSR during a PI cycle or following a UUO or the trapping of an illegal instruction clears USER.

As a control flip-flop, USER implements the restrictions on user programs. Thus in order to restrict the operation of a user program, the executive routine must set USER when it transfers control to the program. If the sole purpose of an interrupt is to service a block IOT and there is no overflow, USER stays set and control automatically reverts to the user program after the IOT.

ILLEG OP

An attempt by a user program to address a location outside of its restricted area in core sets this flag causing a priority interrupt. At the time of the interrupt, PC may point either to the location of the instruction which tried to use the address or to the location following this instruction.

Some subroutine-calling jump instructions store what is referred to in the logic as "miscellaneous bits" in the left half of the location that receives the program count. In returning from the subroutine, the program may use a jump that restores the bits to their original states. Included in the miscellaneous bits are the two carry flags, overflow, PC change, user, and a control bit that is used in a special case for returning from a priority interrupt. The four byte manipulation instructions that load or deposit a character require two main sequences for their execution, and a priority interrupt can occur between them. The first part fetches and, if necessary increments the pointer; the second operates on the byte. If the program jumps to a subroutine for an interrupt that occurs between the two parts, bit 4 is set in the PC store location. Then in the subroutine, the program may determine whether a character operation was interrupted; and upon the return, the stored bit ensures that the interrupted instruction, which must be restarted, will not reincrement the pointer.

2.2 MAIN SEQUENCE

This section is devoted primarily to the manner in which the processor main sequence executes an instruction and sequences the program, but it also treats the control elements that allow entry into the main sequence from the console and the executive system which controls the sharing of processor time by user programs.

a Console Control

Operator control over the processor is exercised through two types of logical inputs associated with the keys and switches. Inputs from the switches are control levels that may provide data or addresses for use by the processor or gates to govern specific processor events. The keys are momentary contact switches that trigger specific events or initiate sequences although the level output of a key may also be used as a gate for events associated with the key action. The complete effect upon the computer of all keys and switches is described in detail in Chapter 3, Operation; we are concerned here only with the way in which the keys affect processor operation, in particular the main sequence.

The logic associated with the console keys consists primarily of a key cycle time chain and a control flip-flop RUN. Normal processor operation is initiated by triggering the main sequence and setting RUN—the 1 state of this flip-flop allows the completion of each main sequence to trigger the next so that the processor executes one instruction after another. Whenever RUN is cleared either from the console or by the program, operation ceases at the end of the current main sequence. The stop keys can, of course, affect the computer while it is running, but most keys that initiate events cannot; only the initiating keys trigger the key cycle. For those key functions that make use of the main sequence, the key cycle performs the necessary preliminary operations, such as transferring information in from the console data and address switches; but for those functions that do not use the main sequence, the key cycle controls the entire operation.

The operator may place the processor in normal operation by means of the START, READ IN, and INSTRUCTION CONTINUE keys. For these functions, the key cycle sets RUN and triggers the main sequence. The INSTRUCTION STOP key halts the processor at the end of the current main sequence by clearing RUN. The processor may also be stopped at the end of any memory access by means

of the MEMORY STOP key, which disables the return from the memory subroutine to the waiting sequence. In this case, the processor is still "running" and normal operation may be resumed through the MEMORY CONTINUE key which simulate a memory subroutine return. The operator may also deposit information in the memory location addressed by the ADDRESS switches or examine the contents of that location while the processor is running. For these two console functions, a single key cycle is merely inserted between two main sequences. For the remaining functions, the processor cannot be running, i.e., RUN must be 0. The EXECUTE key causes the processor to execute as an instruction the word contained in the DATA switches. For this instruction, the key cycle triggers the main sequence but does not set RUN, so the processor stops when the instruction is complete. There are also two keys that allow the operator to examine or deposit information into a sequence of consecutive memory locations without addressing them individually. Each such examine or deposit requires a key cycle and these functions cannot be performed unless the processor is stopped.

For maintenance purposes, the console has a REPEAT switch. When this switch is on, any key function can be repeated at a rate determined by a pair of speed controls. The logic enables this by having the key cycle retrigger itself through a delay whose interval is determined by the speed setting.

b Instruction Execution

Most instructions are executed by the five cycles that comprise the main sequence: instruction, address, fetch, execute, and store. Each main sequence begins when the instruction cycle requests memory access to retrieve an instruction from the location specified by the program counter. Upon receiving the instruction, the processor enters the address cycle and performs the effective address calculation as outlined in 2.1b. If an address is indirect, a new address word is retrieved from memory and the cycle begins again. After repeating the cycle as many times as is necessary to produce the effective address, the processor goes on to the fetch cycle to retrieve the necessary operands. If an accumulator is specified, it is retrieved first and sent to AR. If the instruction uses a double-length operand, a second word is fetched from the next consecutive accumulator (with location 0 being taken as following location 17) and sent to MQ. In some instructions, an extra word must be retrieved from the memory location addressed by either the right or left half of the addressed accumulator. This type of operand is also sent

to MQ. Finally the processor fetches the memory operand as specified by the effective address and leaves it in MB. This last fetch is skipped if E is to be used as control information, an operand, or a jump address.

After fetching the operands, the processor enters the execute cycle in which it performs whatever logical, arithmetic, or control functions are necessary to carry out the instruction. This cycle also increments the program counter by one so that it points to the next instruction in normal sequence. If a jump or skip is being performed, PC is changed following the count. Finally the processor enters the store cycle to deposit the result, which is usually contained in AR. For most instructions, the result may be deposited in an accumulator, in memory or in both as specified by the instruction; for a double-length result, AR and MQ may be stored in consecutive accumulators. The processor then returns to a new instruction cycle.

Although most instructions are performed by the sequence outlined above, there are many that are performed by variations of it. The more complicated instructions are performed by special sequences that are entered from the execute cycle and usually return to the store cycle. Sometimes a special sequence handles the storage itself and returns directly to the instruction cycle. Other instructions must first fetch and operate on a pointer that provides information necessary for the retrieval of the true operand; such instructions require in effect two main sequences. A block transfer repeats the fetch and execute cycles once for every word in the block. Whenever the execute cycle occurs more than once for a single instruction, the incrementing of the program counter is inhibited in all but the final occurrence. In this way, PC points to the next instruction only when the current one is bound to be completed before any interruption can occur.

The actual form of the sequence and the operations carried out in it are determined entirely by the instruction code as decoded from the instruction register. The codes are divided into eightclasses according to the configuration of bits 0-2. If these bits contain 111, the instruction is in the special IOT format and IOT control decodes bits 10-12 to determine which of eight instructions is specified. If bits 0-2 are 000, the instruction is taken to be a programmed operator—there is no further decoding and the processor enters a special sequence from which a subroutine must properly interpret the remainder of the code (and of the instruction word for that matter). In the other six classes the remaining six bits are decoded by the hardware,

primarily by the logic associated with IR. They may be decoded in a variety of ways depending upon the instruction class. Occasionally, single bits are used to represent specific operations, such as specifying the left or right half in a half-word transfer or whether fixed-point multiply is to interpret the operand as an integer or a fraction. In other cases, groups of bits are decoded; for example in the Boolean class four bits determine which of 16 Boolean functions is specified, the other two determine the mode of execution. In some cases, all six bits are decoded to a single control level for an individual instruction that has no modes.

There are some instruction codes that are not used and are executed as no-ops; the unused octal codes are those for which no mnemonic is listed in Table A4-1. Since most instruction codes are divided into sets of bits that are decoded in different ways, it is possible for some combinations of mode and instruction to have no effect on the state of the computer and these may be considered as no-ops. An obvious example is a full-word transfer that does not change the operand and is performed in the self mode.

The way in which instructions are executed is also influenced by the requirements of the priority interrupt system and the executive system. The interrupt channels are strobed at the beginning of every instruction and address cycle; and if a request is discovered, the processor honors it by entering a special PI cycle in which it executes the instruction in the location corresponding to the channel being serviced. For a PI cycle, the processor starts a new main sequence and executes it in the normal fashion except that the address supplied to MA for instruction retrieval comes from a channel address encoder in the PI system rather than from PC, and the strobe is disabled so that the PI cycle cannot itself be interrupted. The instruction executed in a PI cycle must either do an I/O data transfer or transfer control to a subroutine for further service. If the data transfer requires no further service, the processor automatically returns to the interrupted instruction; if further action is required, a second PI cycle is executed so that control can be transferred to a subroutine before honoring any other interrupt. If control is transferred to a subroutine, the interrupt is "held" so that the processor may again be interrupted but only on a channel of priority higher than the one being held; the subroutine is responsible for releasing the interrupt upon completion.

The executive system restricts processor operation in order to permit time sharing by several programs. When running restricted (user mode), each program must operate within the area of

core assigned to it; an attempt to use an address outside of the assigned area causes a flag to be set and immediately initiates an interrupt on the processor channel (the location to which PC points depends upon the time within the main sequence that the illegal memory request was made). The execution of a programmed operator (UUO) is unrestricted, but the locations used by UUOs (40 and 41) are inaccessible to user programs; UUOs executed by user programs always transfer control to the (unrestricted) routine responsible for overall system operation. Besides restricting addresses, the user mode traps (as if they were UUOs) attempt to halt the processor, dismiss an interrupt channel, or operate an I/O device. Instructions executed in PI cycles are unrestricted even if the interrupted program was running in user mode.

c Executive System

The executive system includes the 8-bit memory protection and relocation registers PR and RLR, nets that monitor user instructions, and the user flag. These logic elements allow the processor to be run in a restricted mode to permit time sharing of several user programs. A program that runs unrestricted (the executive routine) must be responsible for overall system operation. The executive routine is responsible for scheduling user programs (assigning core areas, entering user mode, and transferring control to the current program, interrupting when its time is up), for servicing all interrupts and UUOs, for servicing all I/O needs of user programs, and for taking action when it receives control because a user program attempted to use an illegal address or instruction or gave up control through a UUO.

Following power turnon, the processor is automatically in executive (unrestricted) mode, and when it is not running, the operator may place it in executive mode by pressing the I/O RESET key (this action also clears the I/O equipment). During a priority interrupt cycle, the processor runs unrestricted; but if a user program is interrupted, the user flag (which normally implements the user restrictions) remains set. Thus unless one instruction suffices to service an interrupt, the executive routine must within a PI cycle clear the user flag to return the processor to the executive mode and transfer control to one of its subroutines. Similarly, since all UUOs are under executive routine control, the instruction in location 41 must be a JSR, which stores and clears the user flag. The executive routine enters a user program by means of a jump which sets the user flag (JRST with a 1 in bit 12). The return to a user program after an interrupt or UUO may be made by means of a restoring JRST (a 1 in bit 11). This instruction restores all

other flags to their original states and can set the flag but can never clear it. This prevents a user program from leaving user mode as a result of an incorrect restoring JRST.

Each user program is assigned a block in core whose first location is an integral multiple of 2000 octal (since the executive routine must use locations 40 to 57 to service UUOs and interrupts, 2000 is the lowest first address available for a user block); the block size is also an integral multiple of 2000. A user program is restricted to addresses from zero to one less than its block size; if it attempts to use an address equal to or greater than its block size, the illegal operation flag is set and an interrupt occurs immediately on the processor channel. To assign a core area to a program, the executive routine uses a processor DATAO, which loads PR and RLR, respectively, from bits 0-7 and 18-25 of the data word. Each time the memory subroutine is called during a user program, the executive system tests for an illegal address by checking that the address does not exceed C(PR) x 2000 + 1777; the size of the block is equal to [C(PR) + 1] x 2000. At the same time the user address is relocated by adding the block starting address to it; i.e., the address sent out on the memory bus is equal to C(RLR) x 2000 + C(MA). Addresses 0 to 17 are never relocated, so all programs have access to fast memory (note that this means that no user program ever uses the first 16 core locations in its assigned block).

The user flag implements the restrictions on a user program by enabling the relocation and protection circuits and enabling the nets that monitor user instructions. A user program may not use a JRST with a 1 in bit 9 or 10 (an attempt to dismiss an interrupt channel or halt the processor) nor any IOT. These instructions are trapped by having their IR decoder outputs drive the UUO command line when the processor is in user mode. As mentioned above, UUOs are unrestricted, i.e., unrelocated location 41 is executed. Thus in user mode, an illegal instruction is executed as if it were a UUO and thereby returns control to the executive routine.

2.3 ARITHMETIC LOGIC

The arithmetic part of the processor includes the three full-size registers AR, MB, and MQ, the two 9-bit registers SC and FE, the time chains that execute the special sequence instructions and subroutines, and a subroutine interface through which connections are made from the special time chains to the gating for the three main registers. Included in the AR part of

the logic are four flags, AROV, AR CRY0, AR CRY1, and PC CHG. The states of these flags are stored as miscellaneous bits and may be restored by a JRST; they may also be sensed for a jump and cleared by a JFCL.

Transfers of full words or half words may be made between MB and AR, transfers of full words between MB and MQ. MB may also receive PC, IR, or the miscellaneous bits for storage in a UUO and in certain jumps. The two halves of a word can also be interchanged (swapped) in MB. Although the AND function of MB and AR can be formed in MB, it usually plays a passive roll in logical and arithmetic operations by holding an operand which is combined with an operand in AR. Associated with AR is a myriad of gates that implement the clearing or setting of individual bits in a word according to a mask, the formation of the complement, OR, AND, and exclusive OR logic functions, and the shifting of bits left or right. There is also a carry function which can be triggered at any point in the register and produces an arithmetic carry to the left; i.e., it complements the first bit, complements the second if the first changes from 1 to 0, and ripples to the left in this manner until it complements a 0 bit. If this carry chain is triggered only at the register LSB, it adds 1 to the number represented by the contents of the register. Some instructions use the left and right halves of a word to hold a word count and an address; in order to allow indexing of both half words simultaneously, the carry chain can be triggered at AR17 and AR35. Although this is used as two simultaneous index functions, there is no break in the carry chain and an overflow from the right half can carry into the left: hence the pair of index functions effectively adds 1000001 to AR. The above listed functions are the only ones that can be performed directly—all others are executed by combinations of them. If following an exclusive OR, the carry function is triggered at a number of places in the register (the particular places being determined according to the addition algorithm by the previous configuration of the words in MB and AR), it generates the algebraic sum in AR of the numbers originally in MB and AR. Negation (which always means arithmetic negation) is performed by complementing and adding 1. In subtraction, the number in MB must be subtracted from that in AR: for fixed point, the processor performs subtraction by complementing AR, then adding and complementing the result; for floating point, MB and AR are switched and the subtrahend in AR is then negated so the result can be produced merely by adding. Multiplication or division is a sequence of shifts with additions or subtractions interspersed.

The third register, MQ, is used occasionally for temporary storage and there is a special case in character operations where AR and MQ are shifted in parallel for control purposes, but MQ serves primarily as a right extension of AR for handling double-length operands. For actions on a pair of accumulators, the two registers are joined end to end and the double-length operand may be shifted in either direction. Moreover, the opposite ends of both registers may be joined to form a ring and the contents rotated in either direction. In multiplication, the multiplicand comes from AC and the multiplier is either C(E) or E, but when performing the actual arithmetic operation, MB holds the multiplicand and the multiplier in MQ controls the formation of partial products in AR. As bits of the multiplier are used and shifted out of MQ, the low-order bits of the double-length product are shifted in. In division, MQ holds the low-order half of the double-length dividend and as bits are shifted out to AR for use by the division steps, bits of the quotient are shifted in at the least significant end. At the completion of the computation, MQ contains the quotient and AR the remainder, but the divide subroutine then switches their positions so the quotient can be stored in AC.

In floating-point operations, the exponent is first calculated in SC, whose gating provides addition and indexing. In floating scale, the only operations performed are on the exponents. For other floating-point instructions, the exponent is calculated in SC and then stored in FE while SC is used to count the steps in the fixed-point part. Following computations, the exponent is transferred back to SC in case it must be changed while normalizing the result, and finally from SC it is inserted in the exponent part of AR. SC is also used to calculate the position portion of a pointer for a character operation that increments, and from SC the new position is inserted in the pointer in AR.

In addition to the registers, the arithmetic logic also includes the time chains and many control nets for executing the special sequences and subroutines. The basic subroutines, which can be called from any higher level, are the AR subroutine group (which includes fixed-point addition, subtraction, negation, and indexing in either direction), the SC addition subroutine, and the SC shift-count subroutine which simultaneously counts SC and shifts AR and/or MQ (for shift instructions both registers are shifted even though for a single operand only AR contains information). For fixed add and subtract, the execute cycle calls the AR subroutines directly. For other instructions, the processor switches from the execute cycle to a special sequence which calls the lower rank subroutines and which usually returns to the store cycle.

The sequences for character operations, block transfer, shift operations, and floating scale call only the basic subroutines (including the memory subroutine). Floating multiply and divide begin by calling the exponent calculate subroutine, then the multiply or divide subroutine whichever is appropriate, and both terminate by entering the normalize return subroutine which also follows the floating add-subtract sequence. The fixed multiply sequence calls only the multiply subroutine; fixed divide does not make use of an intermediate special sequence but instead enters directly into the divide subroutine.

2.4 MEMORY INTERFACE

The interface that connects the processor to the memory bus includes the memory address register MA, memory buffer MB, user mode registers PR and RLR, memory indicator register MI, and the control logic for the memory subroutine. A processor cycle or special sequence gains access to memory by triggering the memory subroutine, which has entries for read, write, and read-pause-write which must later be followed by a read-write restart. The calling sequence must also supply an address to MA, and if information is to be written, a word to MB. If the processor is in executive mode, the subroutine places the appropriate request levels on the bus immediately, but for user mode there is a delay while the address in MA is compared with PR. An illegal address causes the processor to go to the end of the current main sequence and sets the illegal operation flag requesting an interrupt on the processor channel.

While the comparison against PR is being made, the outputs of RLR and the more significant MA bits are applied to a set of dc adders whose outputs represent the sum of the two registers. If the address in MA is legal, memory control puts the relocated address on the bus (low-order bits are supplied directly from MA, high-order bits from the relocation adders). In the address as received by memory, MA34 supplies the least-significant bit of the address within a single memory and bit 35 is used as the LSB to select the bank. In this way, consecutive addresses are interleaved—all odd addresses in one bank, all even in another. A switch at the memory allows the operator to disable this feature when using a 16K bank (with 8K banks, addresses must be interleaved).

The processor memory subroutine requests a memory cycle by calling memory as a subroutine, and it must wait until the addressed memory accepts the request, which does not occur until

the memory is free and this processor has priority. The processor restarts upon receipt of an acknowledgement signal from memory. If the request is for a write cycle, the processor need wait only until the memory accepts the word in its own buffer; but for a read cycle, it must wait until it receives the information read from the memory location. If the request is made to fast memory, the write takes slightly longer than the read because there is no buffer. If the request is not acknowledged within a considerable time compared to a memory cycle, the nonexistent memory flag is set, requesting an interrupt on the processor channel. Following the acknowledgement signal, the memory subroutine sends a restart pulse to the waiting sequence unless the MEMORY STOP key is on. To restart the processor after a memory stop, the operator must simulate the return to the waiting sequence by pressing the MEMORY CONTINUE key.

If the address in MA is the same as that in the console ADDRESS switches or the operator is examining or depositing information from the console, the contents of the memory buffer are displayed in the memory indicators. On a read MI displays the information read, on a write it displays the information to be written.

2.5 INPUT-OUTPUT SYSTEM

At the processor end of the I/O bus is the in-out transfer control logic that times the transfer of data, initial conditions, and status over the bus by sending command signals (also over the bus) to the device control units. Two of the devices on the bus are the priority interrupt system located in the processor, and the processor itself whose I/O interface contains a number of flags through which internal processor conditions can request priority interrupts and which allow the processor to check its own internal status with IOT instructions.

When the code 111 appears in bits 0-2 of the instruction register, the processor IOT control decodes bits 10-12 to determine the specific IOT instruction. Upon reaching the execute cycle, the processor switches to a special IOT sequence that times the instruction operations and generates the necessary command signals. Only four types of command signals are sent out on the bus; these are for DATAI, DATAO, CONO, and STATUS, of which the first three correspond to individual IOT instructions. BLKI or BLKO requires signals on the bus only

after conversion to a DATAI or DATAO. CONI, CONSZ, and CONSO bring conditions in and the latter two then perform tests; all three generate the STATUS command and affect the peripheral equipment in exactly the same way.

While IOT control is generating the command signals, the device code from IR bits 3-9 is supplied over the bus to enable a gate in the device with that assigned code; signals are sent to all devices but only the selected device can respond. Data or initial conditions are supplied from AR over the bus to the selected device; data or status is supplied from the device over the bus to AR. Among the initial conditions that CONO may supply to a device is a priority interrupt assignment; CONO assigns a channel from 1 to 7 (zero is no assignment); and whenever the device requires service, it requests an interrupt by sending a request signal to the PI system on the bus line corresponding to its assigned channel. Every device except the PI system itself can receive at least one PI assignment.

The PI lines go to the priority interrupt system which contains three 7-bit registers, PIO, PIR, and PIH, to control the seven channels. A given channel is governed by one flip-flop from each register. The PIO flip-flop turns the channel on or off. The PIR flip-flop synchronizes the request to the processor main sequence and in conjunction with the remaining PIR flip-flops and a priority chain, generates an internal request signal for the channel that has priority. The PIH flip-flop holds a break on the selected channel. There are also three control flip-flops for the interrupt system, one that activates it, another that places the processor in a PI cycle, and a third that detects overflow from a block IOT performed in a PI cycle. By checking status, the program can determine whether the system is active and which channels are on. The processor strobes the PI lines at the beginning of every instruction and address cycle, and synchronizes a request signal from any PI line provided that the corresponding channel is on. If a PIR is on (a CONO can set the PIR for a channel even if the channel is off), the processor enters a PI cycle and starts a new main sequence which honors the request by performing the instruction in a particular memory location associated with the channel (if several PIR flip-flops are set, the lowest numbered channel has priority). To retrieve the instruction, MA receives the address from an encoder in the PI system. The encoder outputs are connected to MA in such a way that the channel number is doubled and added to 40. Thus for channel n, the processor executes the instruction in location 40 + 2n (the PI system uses locations 42 to 57, two for each channel). This instruction should be either a JSR to an appropriate subroutine or a block IOT to handle

a data transfer. If it is a BLKI or BLKO and there is no overflow, the processor returns immediately to the interrupted program (another priority interrupt can occur before any instruction in the interrupted program is actually executed). If there is overflow, the processor goes into a second PI cycle in which it performs the instruction in location 41 + 2n, which should be a JSR to an appropriate subroutine. An instruction in a PI cycle should be either a BLKI, BLKO, or JSR; other instructions can be executed but they usually would have unfortunate consequences for the program and could even hang up the processor. If there is a jump to a subroutine, the break is held by setting the PIH flip-flop for the channel. This disables part of the priority chain so that the break routine can be interrupted only by a channel of higher priority. At the completion of the subroutine, the program should dismiss the channel so as to reenable all lower priority channels as well as the channel on which the break occurred.

Since a BLT may require considerable execution time, the PI request lines are also strobed following each word processed in the block. Whenever a request is discovered, the current source and destination addresses are stored in the accumulator and the partial block is terminated. The processor then begins a new main sequence as if to restart the block transfer, but is interrupted instead.

The I/O interface for the processor contains the flags discussed in 2.1c except for the four flags associated with AR and the user flag in the executive system. However two of the AR flags, overflow and PC change, can be set by the CONO that controls the interface flags and supplies it with a PI assignment, and these AR flags plus the user flag can be sensed as processor status (although the user flag is meaningless as status since it is 0 by definition whenever an IOT can be executed). The flag set by an illegal user address, a pushdown list overflow, or a request made to a nonexistent memory automatically requests an interrupt on the processor channel. Setting the clock, overflow, or PC change flag can cause an interrupt only if it has been enabled by a CONO. The program may also use data instructions for the processor: DATAO loads PR and RLR for a user program; DATAI brings in information from the console data switches.

Control units for other devices each contain a data buffer for transfers between the I/O bus and the device, an interface for control connections to the device, and an interface for control connections to the bus. The size of the buffer depends upon the device. It is 36 bits if full

words can be transferred, but smaller if the transfers must be single characters. For an output device, the buffer can be loaded by DATAO; for an input device, DATAI gates the buffer onto the bus. The interface between the control unit and the bus includes a control register and a status register, which usually overlap and may be identical. For initial conditions, CONO can provide at least one PI assignment, place the device in operation, clear whatever error flags there may be, and often provide additional information such as determining the mode of operation or selecting an individual device from several that are connected to the same control unit. Usually all control bits can be examined as status and often additional status signals are supplied by the device. In most cases, the data instructions also perform certain control functions. For input, the loading of the buffer with information from the device usually sets a flag causing a priority interrupt. The processor responds with a DATAI that not only gates the buffer onto the bus but also clears the flag and initiates the retrieval of more information. For output, the transfer of information from the buffer to the device sets a flag causing an interrupt, and the processor responds with a DATAO that not only supplies new information but clears the flag and initiates the next transfer from buffer to device.

Included in this manual are the control units for four I/O devices. The paper tape reader has a 36-bit buffer but information may be retrieved in two modes. In alphanumeric mode, only one 8-bit line is read from the tape; in binary mode, the control unit accepts data from holes 1-6 only in lines in which hole 8 is punched, but it assembles six such characters into a 36-bit word. The punch handles only one character at a time but it still has two modes. In alphanumeric, it punches an 8-bit character; in binary, it punches a 6-bit character in holes 1-6, never punches hole 7, and always punches hole 8. The keyboard-printer is actually two independent devices with one PI assignment. For output, the processor prints single characters; for input, each character typed by the operator is placed separately on the bus. Characters typed at the keyboard are not printed unless the program sends them back out. With the card reader, only a CONO can initiate operations but only one CONO is required per card because once a card is started all 80 columns are read. The program can specify whether an interrupt shall be requested following each column or only when the buffer is full. In binary mode, all twelve holes of each column are read and three columns are assembled into a word. In alphanumeric mode, the Hollerith character in a column is translated into a 6-bit character and the control unit assembles six into a word. If an interrupt is requested but is not serviced before a new column is read, an error flag is set.

CHAPTER 3

OPERATION

This chapter discusses the normal operation of the arithmetic processor, reader, punch, Teletype, and card reader; some maintenance information is included, but the detailed discussion of operation for maintenance purposes is in Chapter 9. Although this chapter is relatively self-contained, it is recommended that the reader first familiarize himself with the functional organization of the equipment as presented in Chapter 2.

3.1 CONTROL PANELS

This section describes the function of the controls and indicators that are readily accessible to the operator; those mounted behind the doors of the bays are described in 9.1. All controls for normal operation of the processor, reader, and punch are on the main operator panel at the center of the console; this panel also contains most of the processor indicators. The panels at the top of bays 1 and 2 contain only indicators, most of which are for maintenance purposes. Indicators for the four in-out devices are on the upper part of the panel located behind the metal cover at the top of the right console bay (the lower part of this panel contains the marginal check controls, which are described in 9.1).

The name used in the logic drawings for a register or control level is listed in parentheses whenever it differs from the name engraved on the panel. When any indicator is lit, the associated flip-flop is in the 1 state or the associated function is asserted. Indicators for logic elements that retain their states over a considerable number of main sequences display useful information while the processor is running, but most indicators change too frequently and are therefore discussed in terms of the information they display when the processor has stopped. For maintenance purposes, the processor may be stopped from the console after every memory subroutine. Switches located inside the bay doors allow stopping after AR subroutines and single stepping through a shift-count. However, the discussion here is limited to stops at the end of a main sequence, i.e., at the completion of an instruction. This includes all programmed halts as well as the situation in which the operator latches down the INSTRUCTION STOP key to run a program at slow speed stopping after every instruction.

a Console Operator Panel

This panel contains indicators for most of the registers and control flip-flops that are of concern to the operator and contains all of the operating keys and switches. The switches supply continuous levels and all but the rotary speed controls and the console lock are 2-position toggles for which up is 1 or on. The keys are momentary contact levers that initiate or terminate operations, or produce an action only while held on.

Indicator Registers

INSTRUCTION (IR0-8) - Bits 0-8 of the instruction just completed. If the left three lights are all off, the instruction is a UUO and the remaining bits are defined by the program; if the left three are on, the instruction is an IOT and the remaining lights display the first six bits of the device code. Any other configuration of the first three bits indicates the basic format, for which the register contains the instruction code.

AC (IR9-12) - For instructions using the basic format, these four bits are usually an accumulator address, but for some instructions they are used for special purposes such as addressing flags. In an IOT instruction the left bit is the LSB of the device code; the remaining three bits specify one of the eight IOT instructions.

I (IR13) - This is the indirect bit, and it should always be off when the processor has stopped at the end of an instruction.

INDEX (IR14-17) - Contains the address of the last index register used in the instruction just completed. If the four lights are all off, there was no address modification in the final address cycle.

MEMORY (MI) - This 36-bit register displays the contents of the memory location associated with any console examine or deposit operation. The lights may also be used to display any desired location while the processor is running.

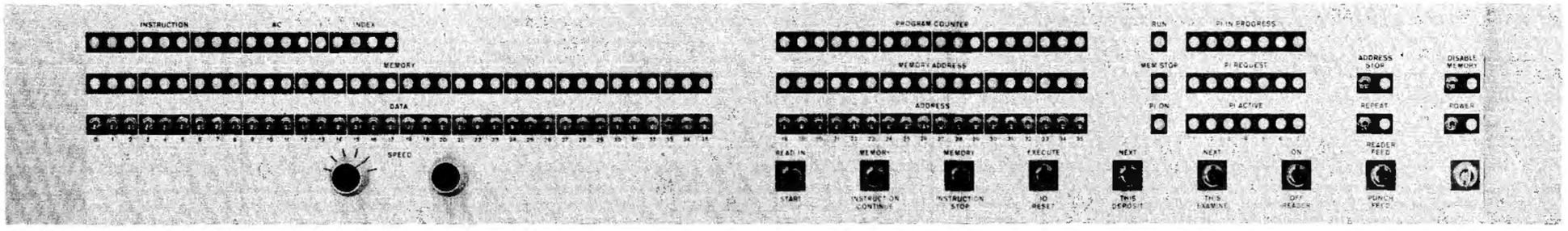
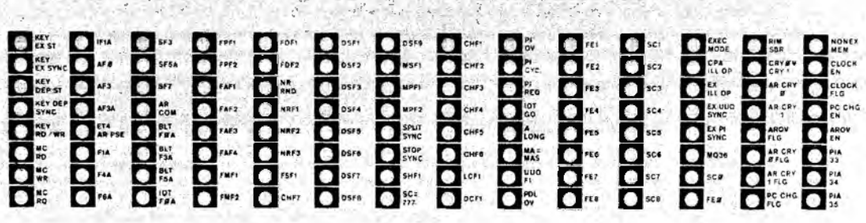


Figure 3-1 Console Operator Panel



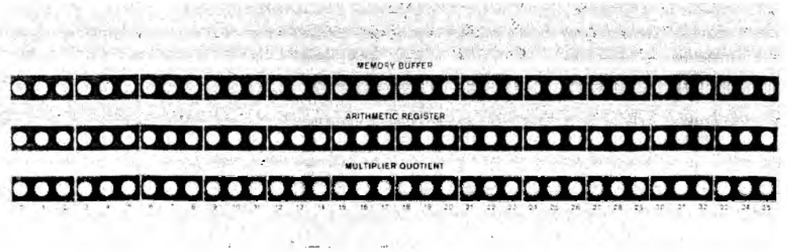


Figure 3-2 Bay Indicator Panels

PROGRAM COUNTER (PC) - This 18-bit register contains the address of the next instruction in the program.

MEMORY ADDRESS (MA) - On a programmed halt this 18-bit register indicates an address one greater than that of the location containing the halt instruction. On an instruction stop in slow speed operation, the register usually contains the address used for the last memory access. However, if there was no storage, either in the store cycle or in a subroutine, it contains the effective address, which may or may not be the address of the last memory access.

In addition to the above there are three processor registers located at the top of the in-out panel (c below).

Switch Registers

DATA (DS) - This register allows the operator to supply a 36-bit word to the processor. The operator may either deposit the word in memory or cause the processor to execute it as an instruction. The program may also read DATA with a DATAI for the processor.

ADDRESS (MAS) - By means of this 18-bit register the operator may specify address for use with the operating keys and switches. Whenever the memory subroutine gains access to the location specified by ADDRESS, the contents of that location are displayed by MEMORY. For a read request MEMORY displays the word read; for a write request the word written is displayed.

Control Indicators

RUN - Lit while the processor is running in normal operating mode, with each main sequence triggering the next. When the light goes off, the processor stops upon completion of the current instruction.

MEM STOP (MC STOP) - If this light goes on at the beginning of a memory subroutine, the processor stops after memory access is completed because the subroutine fails to send a restart

pulse to the calling sequence. If RUN is also on, the processor can be restarted only by lifting the MEMORY CONTINUE key. If RUN is off, other keys may be used but only MEMORY CONTINUE restarts the interrupt key function.

PI ON (PI ACTIVE) - Indicates that interrupt requests can be granted by the priority interrupt system.

PI ACTIVE (PIO1-7) - These lights indicate which PI channels are on. The numerals below the lights specify the channels for these and the following two sets of indicators.

PI REQUEST (PIR1-7) - These lights indicate the channels on which requests have been synchronized. The program can force a request even if a channel is not on; for a request from any other source the REQUEST light can go on only if the corresponding ACTIVE light is on.

PI IN PROGRESS (PIH1-7) - These lights indicate the channels on which breaks are currently being held. Several lights may be on simultaneously, but while a given light is on, no higher-numbered light may go on; a lower-numbered channel can interrupt following the PI cycle(s), and the channel that is actually being serviced is the lowest-numbered one whose light is on. When a PROGRESS light goes on (following a jump to a routine for the break), the corresponding REQUEST goes off and cannot go on again until PROGRESS goes off.

If a break is serviced by a block IOT without overflow, the PIH flip-flop is set and cleared within a single PI cycle so REQUEST goes off without PROGRESS going on. If there is overflow, two PI cycles are required; at the end of the first, PROGRESS will not yet be on and REQUEST will still be on even though the break is being serviced. However, in this case the PI OV, PI CYC, and PI REQ lights at the top of bay 1 will all be on. The lights act in this way because PROGRESS can go on and remain on in a PI cycle only if the instruction performed is not an IOT. Thus a faulty program can hang up the processor in a PI cycle, and the only visual indication that the break is being held is that PI CYC and PI REQ will be on. For example if the channel location contains a CONO, the processor will repeat the instruction indefinitely with REQUEST on and PROGRESS off; PC will be static and will point to the next instruction in the program.

Operating Keys

In the right half of the operator panel is a row of eight 3-position switch levers, each of which is two logical keys. The momentary contact, up and down positions of a given lever are the on positions for the keys whose names are written above and below; the stable center position is off for both keys. The two levers at the right end of the row control the reader and punch, and these may be used at any time whether the processor is running or not. The other twelve keys affect the processor, and of these, two are stop keys, the others are initiating keys (i.e., they trigger the key cycle). Although special considerations for individual keys are given below, it is assumed throughout the discussion that the executive system is not in use, i.e., that both operator and program have access to all of memory and no operations are illegal. In order to use the keys properly when the executive system is in use, the operator must be fully aware of the special conditions imposed (a complete discussion of the relation of the keys to the programmed operation of the system is presented in 3.3b).

START - This key functions only if RUN is off. It places the processor in normal operation (lighting RUN) and causes the first instruction cycle to retrieve an instruction from the location specified by the ADDRESS switches.

READ IN - This key is exactly the same as START except that it also causes the processor to enter the readin mode, lighting the RIM SBR indicator at the top of bay 1. In this mode the fast memory is disabled, and any memory call with an address 17 or less is given access to the readin area, the normally inaccessible bottom 16 core locations. Whenever an instruction is retrieved from any location above 17, the processor leaves the readin mode.

INSTRUCTION STOP (INST STOP) - Turns off RUN, causing the processor to stop at the completion of the current instruction. This key has a catch that allows it to be left in the on position for single step operation. The turnon of the key triggers events that facilitate emergency stops (for details see 3.3c).

INSTRUCTION CONTINUE (INST CONT) - This key functions only if RUN is off. It causes the processor to resume normal operation (lighting RUN) beginning with the instruction in the

location specified by PROGRAM COUNTER. By leaving INSTRUCTION STOP on, the operator can single step instructions by pressing INSTRUCTION CONTINUE. The latter key also has a catch so that by leaving both keys on and using the REPEAT switch, a program can be run at slow speed.

MEMORY STOP (MEM STOP) - This key has a catch that allows it to be left in the on position for single step operation. While the key is on, the MEM STOP light goes on at the beginning of every memory subroutine, causing the processor to stop at the completion of each memory access. During single step operation a call for read-pause-write in the fetch cycle generates only a read request so the processor does not hold memory during the stop. The subsequent restart then triggers a separate write cycle. This key is used only for maintenance purposes and the meaning of the lights depends upon where the stop occurs within the main sequence. However, MEMORY ADDRESS always displays the location to which access was made.

MEMORY CONTINUE (MEM CONT) - This key functions only if the MEM STOP light is on, and it then restarts whatever sequence was interrupted by the MEMORY STOP key (it also turns off the light). By leaving MEMORY STOP on, the operator can use MEMORY CONTINUE to single step by memory calls. The latter key also has a catch so that by leaving both keys on and using the REPEAT switch, a program can be run at slow speed from one memory call to the next.

EXECUTE (EXEC) - This key functions only if RUN is off. It causes the processor to execute the instruction contained in the DATA switches and stop immediately upon completing it. While the key is on, the normal program counting in the execute cycle is inhibited; thus PC cannot be affected unless a skip or jump is executed. A programmed skip always increments PC once for the normal program count and may increment it a second time for the skip; from the console a skip increments PC at most once. A programmed jump always increments PC before saving it so that it points to the next instruction in the program; when executed from the console, a jump loads PC normally but saves the count that is already in it.

IO RESET - This key functions only if RUN is off. It clears all flags, control flip-flops and control registers in the processor (placing it in executive mode) and in most equipment connected to the in-out bus.

DEPOSIT THIS (DEP) - Deposits the contents of DATA in the location specified by ADDRESS. The word deposited is displayed by MEMORY. If RUN is off during the deposit, the processor stops with the MA lights displaying the address of the affected location. This key should be used while the processor is running only if there is no chance of a program halt occurring (3.3b).

DEPOSIT NEXT (DEP NXT) - This key functions only if RUN is off, It deposits the contents of DATA in the location whose address is one greater than that specified by MEMORY ADDRESS, and the word deposited is displayed by MEMORY. At the completion of the operation MA contains the address of the affected location.

EXAMINE THIS (EX) - Causes MEMORY to display the contents of the location specified by ADDRESS. If RUN is off during the operation, the processor stops with the MA lights displaying the address of the examined location. This key should be used while the processor is running only if there is no chance of a program halt occurring (3.3b).

EXAMINE NEXT (EX NXT) - This key functions only if RUN is off. It causes MEMORY to display the contents of the location whose address is one greater than that specified by MEMORY ADDRESS. At the completion of the operation MA contains the address of the examined location.

READER ON - Turns on the reader motor, energizes the brake, and triggers a PI request on the reader channel.

READER OFF - Turns off the reader motor, releases the brake, and triggers a PI request on the reader channel.

READER FEED - Feeds tape through the reader while held on (provided the reader is on).

PUNCH FEED - While this key is held on, the punch generates blank tape, i.e., tape with only feedholes punched.

Operating Switches

The first four switches are toggles located at the right end of the operator panel, and associated with each is an indicator that lights while the switch is on.

POWER - This switch applies power to the processor and the control units for reader, punch, and Teletype, and makes power available to all external units (memories, peripheral equipment) whose local power controls are in remote. Almost every unit has its own power switch, which if left on, allows the unit to come on with system power. Exceptions include the reader, which must be turned on and off at the processor console, and the punch, which is turned on by the logic and goes off automatically whenever it is not called for 5 sec. After turning POWER on, wait a few seconds to allow the power clear to terminate and memory power to come on.

ADDRESS STOP (ADDR STOP) - While this switch is on, a memory stop occurs whenever access is made to the location specified by the ADDRESS switches. At the stop the MEM STOP light is on, and MEMORY displays the word read or written. Throughout the time that the switch remains on, any fetch cycle call for read-pause-write generates only a read request, so the processor does not hold memory following the stop. The subsequent restart then triggers a separate write cycle.

DISABLE MEMORY (MEM DISABLE) - While this switch is off, the failure of a memory to respond within 100 μsec to a request for access turns on the NONEX MEM light on the bay 1 indicator panel, causing a PI request on the processor channel. If the switch is on, such failure causes the processor to hang up in the memory subroutine. The operator can free the processor by pressing INSTRUCTION STOP and then I/O RESET.

REPEAT - Causes the sequence initiated by an operating key to be repeated as long as the key is held on. The sequence is iterated at a rate determined by the SPEED switches.

SPEED - These switches allow the operator to vary the repeat interval from 3.4 μsec to 8 sec in six overlapping ranges. They include a 5-position rotary range switch and a potentiometer knob for fine control within each range.

Console Disable - In the lower right corner of the panel is a key-locked switch. Turning the key clockwise disables all operating keys and switches on the panel (except those for the reader and punch) so no one can interfere with the operation of the processor.

b Bay Indicator Panels

Figure 3-2 shows the indicator panels at the tops of bays 1 and 2. Bay 2 displays the three main full-word registers: memory buffer MB, arithmetic register AR, and multiplier-quotient register MQ. Since the results of an instruction are stored in memory, these registers are useful primarily for single-step maintenance operation, and their contents at an instruction stop depend entirely upon the instruction just performed. If the instruction requires storage, MB always contains the last word stored. AR contains the word stored in an accumulator (if any), and for a double-length result MQ contains the word stored in a second accumulator.

On bay 1 are the indicators for flip-flops and control levels. Indicators for the flags described in 2.1c are at the right end of the panel. At the top of the third column from the right is the EXEC MODE light, which is driven from the 0 output of the user flag and is thus lit when the executive routine is running (or the executive system is not in use). The remaining flags are as listed in the text, although the names engraved on the panel are in many cases abbreviated from those used in the logic drawings. PDL OV is at the bottom of the sixth column from the right. The second light in the third column, CPA ILL OP, is the illegal operation flag, which indicates that a user program has attempted to address a location outside of its assigned core area and should not be confused with EX ILL OP just below. The latter flip-flop inhibits relocation when a UUO or an illegal user instruction is trapped, and the light is always off at an instruction stop. In the second column are the AR overflow and carry flags and the PC change flag. The carry flags should not be confused with the carry flip-flops above them: the flip-flops detect carries in AR and their states are transferred to the flags only in those instructions wherein the information is relevant to the program. The remaining flags are in the right column.

Besides the flags, the indicator panel also includes shift counter SC, floating-exponent register FE, several important control levels, and a multitude of flip-flops that govern the sequencing of the various processor cycles, special sequences, and subroutines. The following

indicators are of importance to the operator in normal operation (unless otherwise specified, the meaning of a light is given for an instruction stop):

KEY EX SYNC, KEY EX ST - If the operator presses EXAMINE THIS while the processor is running and it stops with both of these lights on, the desired key function has not been performed. If only the sync light is on, the operation may have been performed incorrectly (see 3.3b). If the key is pressed while RUN is off, the start light does not go on at all, but the sync light goes on and remains on until some other initiating key is pressed.

KEY DEP SYNC, KEY DEP ST - If the operator presses DEPOSIT THIS while the processor is running and it stops with both of these lights on, the desired key function has not been performed. If only the sync light is on, the operation may have been performed incorrectly (see 3.3b). If the key is pressed while RUN is off, the start light does not go on at all, but the sync light goes on and remains on until some other initiating key is pressed.

CHF7 - If this light (bottom, fifth column from left) is on following a PI cycle that executes a BLKI or BLKO, an interrupt has occurred between the two parts of a character operation. Following a JSR in a PI cycle, the light will be off even if a character operation was interrupted. If CHF7 is on following a JRST, the instruction is returning from a break and the processor is about to restart an interrupted character operation.

SPLIT SYNC - Indicates that if there was a read-pause-write call during the preceding fetch cycle, it triggered only a read request, and the subsequent restart triggered a separate write cycle.

STOP SYNC - Indicates that the preceding fetch cycle triggered a read-pause-write memory cycle.

PI OV - Indicates that a BLKI or BLKO performed in a PI cycle has overflowed.

PI CYC - This light goes on when a PI request is honored, but is still on at the completion of an instruction only if a second PI cycle is required for the interrupt (i.e., the instruction was a BLKI or BLKO that overflowed).

PI REQ - At the completion of an instruction this light is on whenever PI OV and PI CYC are on. If those two lights are not on, PI REQ indicates that the PI system is active and a request that has been synchronized has not yet been honored. This can occur if a previous instruction activated the system and some requests were already waiting. If the system was already active, either a request was made by the program or synchronized by a BLT, or several requests were synchronized simultaneously and the processor has just finished servicing one of those with higher priority.

A LONG - Indicates that the address cycle of the preceding instruction used an index register for address modification or used an indirect address.

MA = MAS - Indicates that the number displayed by the MEMORY ADDRESS lights is identical to that contained in the ADDRESS switches. This light is on whenever an address stop occurs but may be on at other times as well.

EX PI SYNC - Indicates that the main sequence just completed was a PI cycle. This light remains on even when PI CYC goes off before the instruction is completed.

RIM SBR - Goes on when the operator presses the READ IN key and remains on until an instruction is retrieved from a location above 17. The light is always on if the system includes no fast memory, or if it is not in use.

PIA 33, 34, 35 - Indicate the PI channel assigned to the processor. If all three lights are off, no channel has been assigned.

At the completion of an instruction, SC and FE may have any configuration. Besides the flags and the control indicators discussed above, the following lights may be on at an instruction stop: MC WR, NRF2, NRF3, DSF7, MPF2, SC = 777, UUOF1, EX UUO SYNC (always on),

MQ36, CRY0 ⊻ CRY1, AR CRY0, AR CRY1. None of the remaining lights should be on at an instruction stop. At a memory stop MC RD will be on if the memory cycle was used to retrieve information, and at least one other light will be on to indicate the point at which a time chain is stopped awaiting the return from the memory subroutine. For example if KEY RD/WR in the left column is on, the memory stop occurred in an examine or deposit operation initiated at the console. Refer to 9.1 for further information on the use of these indicators for maintenance purposes.

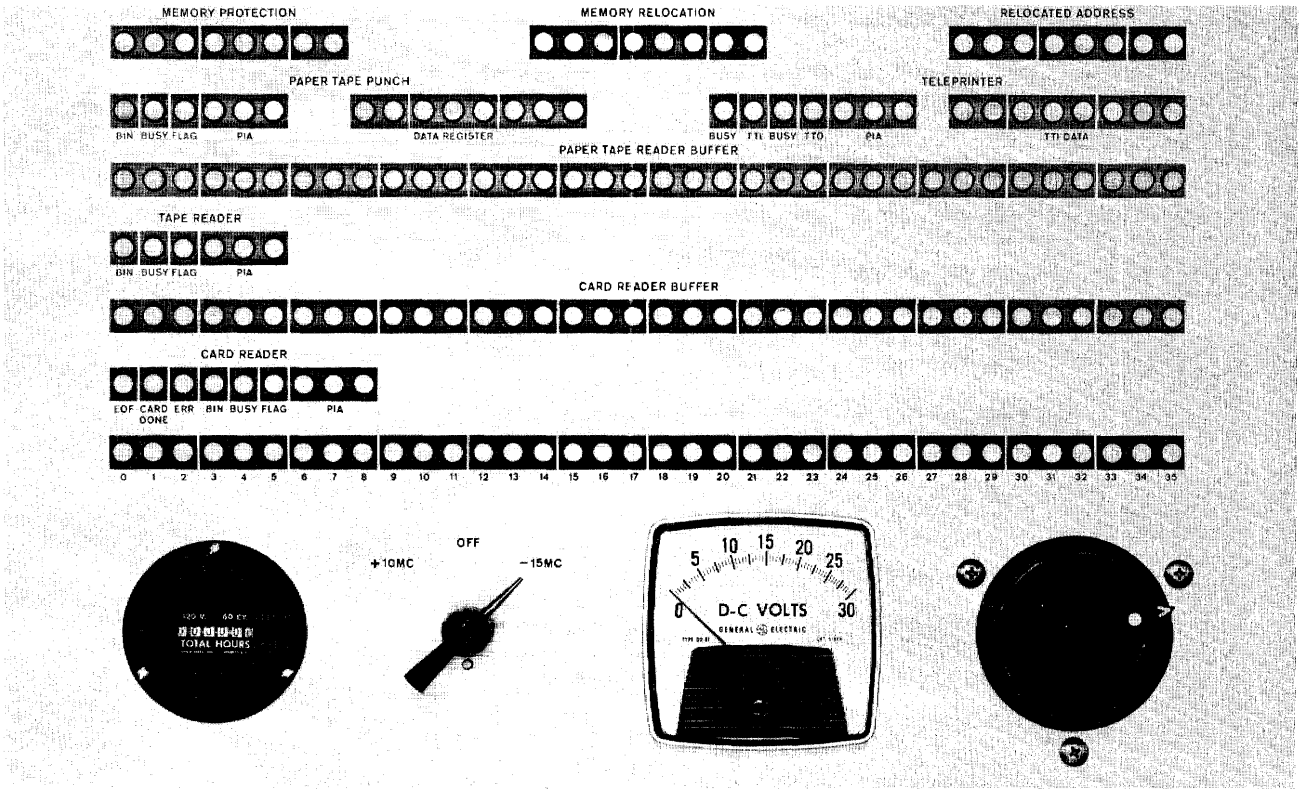


Figure 3-3 In-Out and Marginal Check Panel

c In-Out Indicator Panel

Figure 3-3 shows the panel that contains the in-out indicators and the marginal check controls (the latter are described in 9.1). At the top of the panel are three 8-bit indicator registers that are associated with the executive system rather than the in-out equipment. These are as follows:

MEMORY PROTECTION (PR) - Defines the size of the block in core available to a user program. The number of locations in the block is 2^{10} times the number one greater than that contained in the register. Each program must use addresses from zero to one less than the block size. If a user attempts access to an address greater than the number in PR followed by ten ones, i.e., to an octal address greater than C(PR) x 2000 + 1777, the processor skips the remainder of the current instruction, and the CPA ILL OP light at the top of bay 1 goes on causing a priority interrupt on the processor channel.

MEMORY RELOCATION (RLR) - Specifies the position of a user block in core. The address of the first location in the block is the number contained in RLR followed by ten zeros. Each user address other than for fast memory is relocated to the assigned block by adding C(RLR) x 2000 to the number displayed by MEMORY ADDRESS.

RELOCATED ADDRESS (RLA) - These lights display the most significant eight bits of a user address as it is placed on the memory bus (the least significant ten bits come directly from MA).

The bottom row of indicators on the panel displays the contents of the 36 data lines in the I/O bus. Since the bus is reset following every data transfer, the lights should always be off when the processor is stopped. The remaining lights are the buffers and some of the control and status bits for the reader, punch, keyboard-printer, and card reader. Each device has a 3-bit PIA register that contains the number of the PI channel assigned to it. Whenever the FLAG light for any device goes on, a PI request is made on the channel specified by the associated PIA (if all three PIA lights are off, there is no channel assignment). The lights labeled TELEPRINTER are actually for two distinct devices, keyboard and printer. Both share a common PI assignment but have duplicate control bits. The data buffer shown is actually for the keyboard; the printer buffer is not shown because it automatically clears as each character is being transmitted.

PAPER TAPE PUNCH -

DATA REGISTER (PTP1-8) - Contains the last character punched. The buffer bits are numbered 1 to 8 from right to left and correspond to a frame of

tape viewed with the feed hole near the right edge. The buffer receives information from bus lines 28-35, with line 35 supplying the information for bit 1.

BIN (B) - While this light is on, any punch operation always punches hole 8, never punches hole 7, and punches hole 6-1 according to the information on bus lines 30-35. While the light is off, the information on lines 28-35 is punched.

BUSY - Indicates that the punch is in operation.

FLAG - Causes a PI request upon completion of a punch operation. FLAG goes off when the program supplies another character.

TELEPRINTER -

TTI DATA (TTI1-8) - Contains the last character received from the keyboard. The buffer bits are numbered 1 to 8 from right to left so that when a character is shifted in at the left, the first bit received ends up in buffer bit 1. The character is transferred to the processor over bus lines 28-35 with bit 1 on line 35.

BUSY, TTI (BUSY, FLAG) - This pair of lights (at the left) is for the keyboard. BUSY goes on when a key is struck. When the entire character is assembled in TTI DATA, BUSY goes off and TTI goes on, requesting an interrupt. TTI goes off when the program retrieves the character.

BUSY, TTO (BUSY, FLAG) - In the right pair of lights, BUSY is on while a character is being transmitted to the printer. When transmission is complete, BUSY goes off and TTO goes on, requesting an interrupt. TTO goes off when the program supplies a new character.

PAPER TAPE READER -

BUFFER (PTR0-35) - Contains data read from tape but no yet retrieved by the program. The 36 buffer bits are numbered to correspond to the bus lines. Characters of six or eight holes from tape are brought in at the right end of the buffer, with bit 35 receiving hole 1.

BIN (B) - When this light is on, each reader operation reads hole 6-1 of only those characters in which hole 8 is punched and assembles six such characters into a 36-bit word. When each character is brought into the buffer at the right, the previously read characters are shifted left. When BIN is off, each read operation retrieves a single character, sensing all eight holes.

BUSY, FLAG - When BUSY goes on, the reader goes into operation retrieving information in the manner specified by BIN. When the required number of characters is retrieved, BUSY goes off and FLAG goes on, requesting an interrupt. FLAG goes off when the program retrieves the information from the buffer. FLAG is also set when the operator turns the reader motor on or off.

CARD READER -

BUFFER (CR0-35) - The 36 bits of this buffer are numbered left to right to correspond to the bus lines. Six-bit characters from cards are brought in at the right end of the buffer over reader signal lines 1, 2, 4, 8, A, B, with buffer bit 35 receiving information from line 1. The program specifies whether information retrieved from a card is to be placed on the bus in units of one, two, or six characters. If more than one character is to be read per bus transfer, previously read characters are shifted left in the buffer as new ones come in.

BIN (B) - While this light is on, a card is read in binary mode wherein each column is read as two characters. The first character is from the lower half of the column (holes 4-9) with hole 9 on reader line 1; the second character is the upper half (holes 12, 11, 0, 1, 2, 3) with hole 3 on line 1. If the light is off, reading is in alphanumeric mode in which the reader converts the Hollerith character in a column to the Burroughs 6-bit code, and six columns are required to fill the buffer.

BUSY - This light goes on when the program requests that the reader begin a card cycle, and the light remains on until the entire card is read.

FLAG - Each time the buffer contains the amount of information specified by the program, this light goes on, requesting an interrupt. Retrieving the data turns FLAG off and clears the buffer, which is cleared automatically if the program does not respond to the request before the next column is read. FLAG may light after each column or only when the buffer is full; depending on whether BIN is off or on respectively, a single column is one or two characters, and a full buffer contains six or only three columns. FLAG also signals that the reader has finished an entire card regardless of the number of characters in the buffer, or that a card jam has occurred.

CARD DONE - Lit from the time the reader completes one card cycle until it starts another. It is possible for CARD DONE and BUSY to be lit at the same time. The program turns on BUSY to cause the reader to begin a card cycle, but CARD DONE remains on until the cycle actually starts.

EOF - When the card hopper is empty, pushing the END OF FILE button on the reader turns on this light. It also turns on FLAG, requesting an interrupt.

ERR (CREL) - Indicates a validity check or read check error in the reader.

3.2 OPERATION OF IN-OUT EQUIPMENT

This section describes the normal operation of the photoelectric perforated tape reader, paper reader, paper tape punch, keyboard-printer, and card reader. Information for other devices is included in their maintenance manuals and in the operator manual, PDP-6 Operation. DEC also supplies manufacturer manuals for all devices included in a PDP-6 system.

a Tape Reader

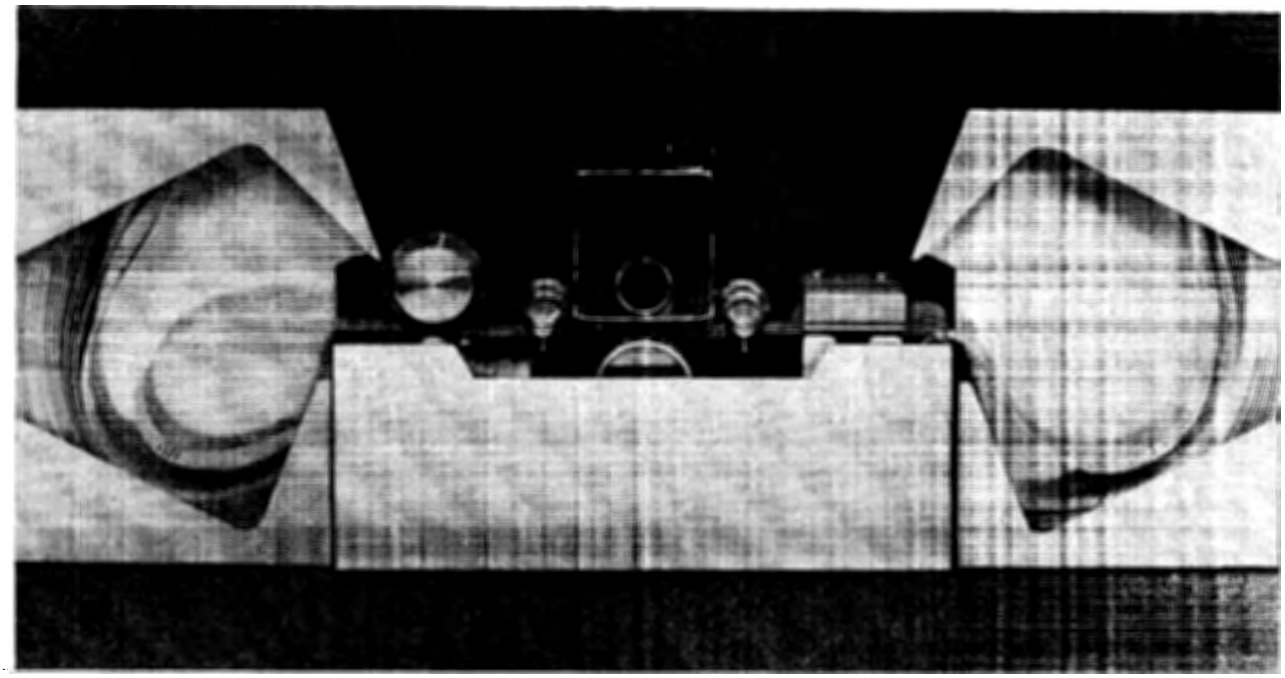


Figure 3-4 Paper Tape Reader Type 760

Before loading a tape in the reader, turn off the reader motor by pressing the READER OFF key. This releases the brake so that tape may be inserted, and it also requests an interrupt on the reader channel to inform the program that the reader is unavailable. Place the fanfold tape stack vertically in the bin at the right with the tape oriented so that the front end of the tape is nearest the read head and the feed holes are nearer the reader mounting panel, i.e., away from the operator. Take three or four folds of tape from the bin and slip the tape into the reader from the front so it is threaded as shown in Figure 3-4. Make sure that the part of the tape in the left bin is placed to correspond to the folds, otherwise it will not stack properly.

Once the tape is properly loaded lift the READER ON key to start the motor and energize the break. This also requests an interrupt to inform the program that the reader is on. When using the readin mode loader, always turn on the reader before starting the loader. The program makes use of the reader by sending signals to the clutch, which moves the tape past the sensing photocells. After the program has finished reading the tape, run out the remaining leader by lifting the READER FEED key, or turn the reader off so the tape may be slipped out directly.

b Tape Punch

The punch is located in a drawer at the top of the left console bay. The punch mechanism faces the right side of the drawer. Fanfold tape is fed from a box as shown in Figure 3-5. After punching, the tape moves into a storage bin from which the operator may remove it through a slot on the front of the drawer.

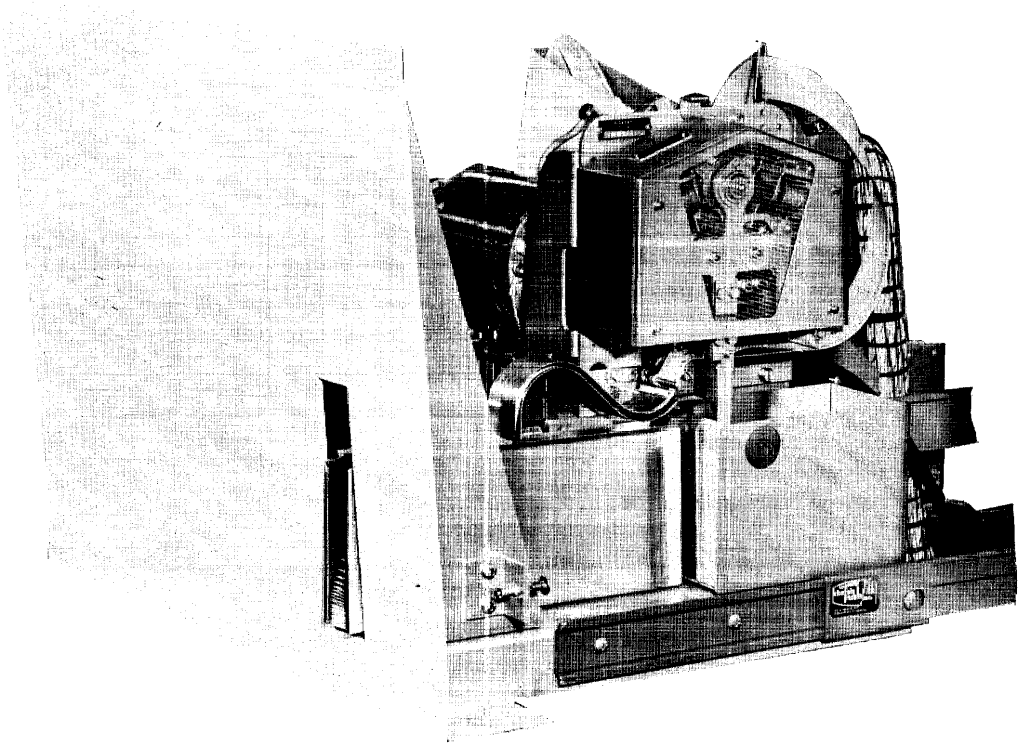


Figure 3-5 Paper Tape Punch Type 761

To load the punch, first empty the chad box below the punch mechanism. Then tear off the top of a box of fanfold tape (the top has a single flap; the bottom of the box has a small flap in the center as well as the flap that extends the full length of the box). Set the box in the frame at the right side of the punch and thread the tape through the mechanism as shown in Figure 3-5. The arrows on the tape should point in the direction of tape motion. If they point in the opposite direction, the box was opened at the wrong end; remove the box from the frame, seal up the bottom, open the top, and thread the tape correctly. After loading the tape, hold down the console PUNCH FEED key long enough to feed approximately 18 in. of leader. Make sure the tape is feeding and folding properly in the storage bin.

To remove a length of perforated tape from the bin, first hold down PUNCH FEED long enough to provide an adequate trailer at the end of the tape (and also leader at the beginning of the next length of tape). Remove the tape from the bin and tear it off at a fold within the area in which only feed holes are punched. Make sure that the tape left in the bin is stacked to correspond to the folds; otherwise, it will not stack properly as it is being punched. After removal, turn the tape stack over so the beginning of the tape is on top, and label it with both name and date.

c Teletype Keyboard-Printer

The teletypewriter (Figure 3-6) provides two-way communication between operator and computer. It is actually two independent devices, keyboard and printer, which may be operated simultaneously. The equipment operates at speeds up to ten characters per second, with 8-bit characters plus start and stop control signals transmitted serially. Located at the right front of the unit is a 2-position rotary switch, OFF/LINE. When this switch is set to LINE, the unit is on line and it goes on and off with system power.

The keyboard resembles that of a standard typewriter with four rows of keys and a space bar. Striking a key transmits a character to the Teletype control unit connected to the bus, but the character is printed or the function executed only if the processor sends it back to the printer. The line feed moves the carriage only vertically with a spacing of six lines to the inch. The return moves the carriage to the left margin but does not feed a line: to start a new line the operator must strike both return and line feed. Codes for the characters on the lower parts

of the key tops can be transmitted merely by striking the keys. Codes for printable characters on the upper parts (punctuation, ampersand, percent sign) are transmitted by holding down the shift key when striking the character key. Control codes are transmitted by holding down the control key, CTRL, when striking the appropriate character key. Codes for all characters listed on the keyboard and some that are not can be transmitted to the computer, but codes for some of the control functions have no effect on the printer when sent back. Table A4-2 lists all codes, their ASCII assignments, and the key combinations required to transmit them. Because of recent changes in the code, there may be slight differences in the printing characters associated with certain key positions. In such cases alternate characters are listed in parentheses.

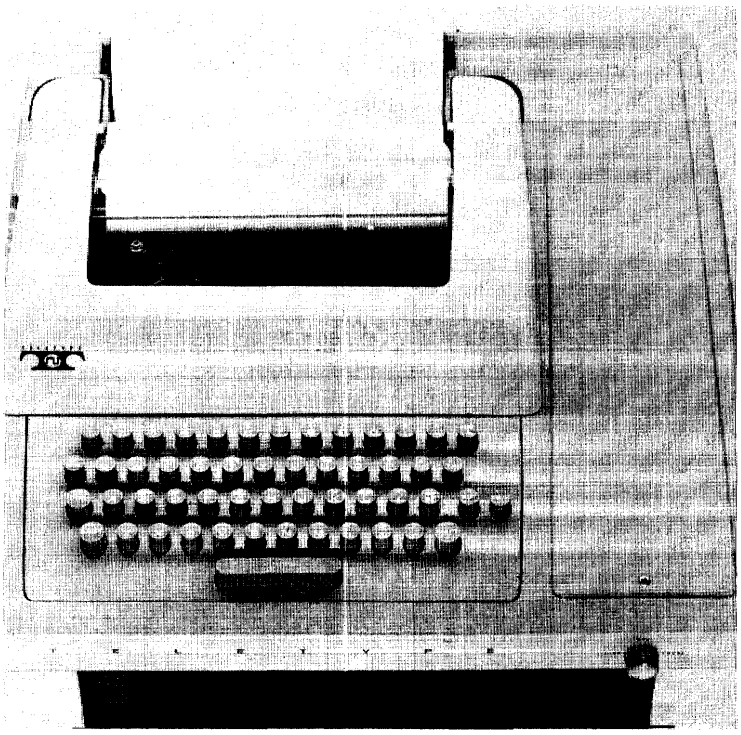


Figure 3-6 Keyboard-Printer Type 626

In line with the space bar below the keyboard are four red buttons. At the right end is the repeat button REPT. Pressing this button and striking any character key causes repeated transmission of the corresponding code so long as REPT is held down. Characters that require the shift key may also be repeated in this manner, but there is no repetition of control characters.

The red button on the left, BRK RLS, is not connected in the console teletypewriter. The remaining two buttons, LOC LF and LOC CR, are the local line feed and carriage return. These buttons affect the printer directly and do not transmit codes over the bus.

Paper installation, ribbon replacement, and the procedure for setting horizontal and vertical tabs are described below. All references are to figures in typing unit section 574-220-100 in Vol. 1 of Teletype Bulletin 281B (Technical Manual 35 Keyboard Send-Receive (KSR) and Receive-Only (RO) Teletypewriter Sets).

Paper

The unit has a sprocket feed and uses 8-1/2 x 11 fanfold form paper. The supply is held in a tray at the back of the unit, and printed forms can be torn off against the edge of the glass window in front of the platen. To replace the paper first remove the upper cover by pressing the cover release button on the right side. To free the remaining old paper for removal, lift the paper guides by pushing the handle marked PUSH at the right of the platen. To insert new paper from the tray, offer it up below the platen at the rear, lining up the holes at the edges of the paper with the sprockets, and press the local line feed button to draw the paper in under the platen.

Ribbon

Replace whenever it becomes worn or frayed or when the printing becomes too light. Disengage the old ribbon from the ribbon guides on either side of the type block, and remove the reels by lifting the spring clips on the reel spindles and pulling the reels off (the ribbon feed mechanism is called out in Figure 4). Remove the old tape from one of the reels and replace the empty reel on one side of the machine; install a new reel on the other side. Push down both reel spindle spring clips to secure the reels. Unwind the fresh ribbon from the inside of the supply reel, over the guide roller, through the two guides on either side of the type block, out around the other guide roller, and back onto the inside of the take-up reel. Engage the hook on the end of the ribbon over the point of the arrow in the hub. Wind a few turns of the ribbon and make sure that the reversing eyelet has been wound onto the spool. Make sure the ribbon is seated properly and feeds correctly in operation.

Tabs

The horizontal and vertical tabulator mechanisms are also called out in Figure 4. Each is a slotted wheel surrounded by a spring on which are mounted a number of tab stops. The horizontal tab mechanism is shown in detail in Figure 47. The slotted wheel is mounted on the spacing drum, and a tab can be set by inserting a tab stop in a groove where it catches the tabulator pawl when the type block carriage is in the desired position. With needle-nose pliers or equivalent, lift the tab stop out of the slot in the wheel against the spring tension. Slide the stop along the spring in the desired direction, and reinsert it into the slot at the new location. A stop may be removed from use by turning it so that it does not catch the pawl. Figure 49 shows the vertical tabulator mechanism. The slots in this disc allow vertical tabs at any desired line, but adjacent tabs must be at least 1 in. apart.

d Card Reader

The B122 Card Reader handles 200 cards per minute and has a hopper and stacker capacity of 500 cards. With a trivial change in the control unit logic, the processor can control the B124 Card Reader. Its operation is similar to that described here, but the maintenance information given in Chapter 9 applies only to the B122. The B124 handles 800 cards per minute and has a hopper and stacker capacity of 2000 cards. In both machines the cards are read lengthwise and sensed photoelectrically. In Figure 3-7, the hopper is at the right, the stacker at the left; in the center is a console that contains the operating buttons and indicators. Of the following four indicators, the first is white, the other three are red error indicators.

NOT READY Indicates one or more of the following:

START button has not been pushed

Hopper empty

Stacker full

Card jam (feed check)

Read check

Validity check (only when VALIDITY ON button is lit)

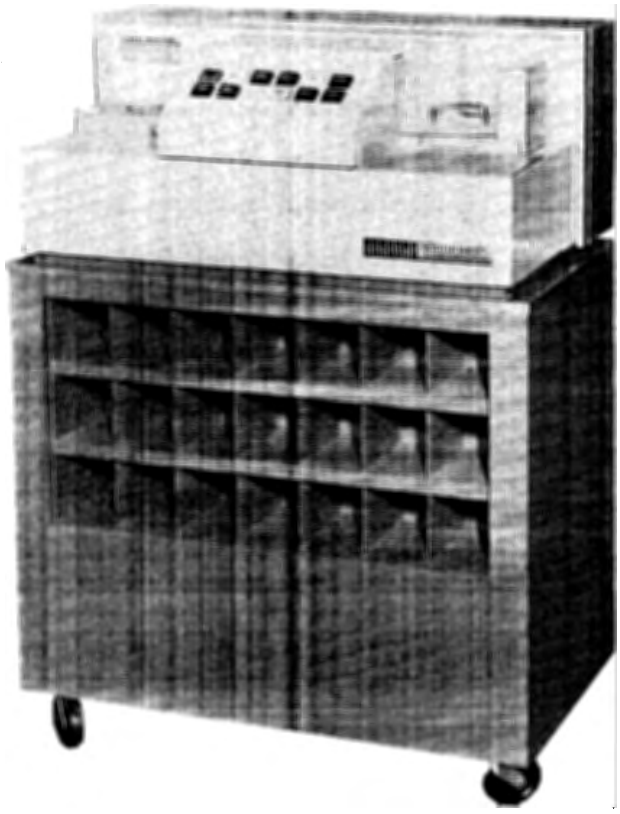


Figure 3-7 Card Reader Type 461

The reader cannot respond to the program while NOT READY is lit. The program may check a not-ready status bit to determine if the reader is available. This status bit is 1 on any of the above conditions and also when the reader power is off (the console cover is interlocked to turn power off when not in place).

READ CHECK - Indicates a NOT READY condition due to a malfunction in the read circuits, e.g., exciter lamps, solar cells, photo amplifiers. There is no read checking during any card cycle; otherwise, read checking is continuous. When this light is on, the ERR light on the processor in-out panel is also on and the reader error status bit is 1.

FEED CHECK - Indicates a NOT READY condition due to a jammed card or failure to select a card. The drive motors stop as soon as the light goes on. A signal to the processor turns on the FLAG light on the in-out panel, requesting a priority interrupt on the channel assigned to the reader; the signal also supplies a status bit to the program.

VALIDITY CHECK - This light functions only if the VALIDITY ON button is lit. It then indicates that an invalid punch combination has been read in alphanumeric mode. When this light is on, the ERR light on the processor in-out panel is also on and the reader error status bit is 1.

Located on the left side of the stacker is the main power switch that controls power to the reader auxiliary power supply. With this switch on, the reader may be turned on either at the reader console or by turning on system power at the processor console. The reader console contains seven buttons, three of which light; the first six below are momentary contact, the last is alternate action.

POWER ON - Green button which turns on the main power supply, the reader motors and, after a 3-sec delay, the reader control logic. This action is duplicated by turning on system power at the processor console. Button is illuminated when power is on.

POWER OFF - Turns off the main supply, but the auxiliary supply remains on.

START - Turns off the NOT READY indicator provided no other not-ready condition exists.

STOP - If the reader is in operation, this button turns on the NOT READY light and the reader stops when the current card runs out to the stacker.

RESET - Turns off the three red check lights: READ CHECK, FEED CHECK, VALIDITY CHECK.

END OF FILE - Pressing this white button when the hopper is empty, lights the button and turns on the EOF and FLAG lights on the processor in-out indicator panel, requesting a priority interrupt on the channel assigned to the reader; the button light goes out when cards are placed in the hopper. The signal generated by the button can be checked as a status bit by the program.

VALIDITY ON - This is an alternate-action yellow button that is lit when on. An invalid punch combination read in alphanumeric mode is sent to the reader control unit as an all-zero character; when the switch is on, and invalid punch also lights VALIDITY CHECK and stops the reader.

In addition to the above there are also interlocks in the hopper and stacker and a LOCAL RUN toggle switch under the cover; the interlocks generate the NOT READY condition when the hopper is empty or the stacker is full. Raising the toggle switch (local) causes the reader to feed cards continuously until the hopper is empty, and then stop with the FEED CHECK light on. With the switch in remote (down), card cycles can be started from the processor whenever the NOT READY light is off.

For operation off line, turn on the main power switch and press POWER ON; after 3 sec NOT READY should light. For normal operation on line, the main power switch is turned on at system power turnon and the reader is left on with NOT READY lit whenever it is not in use. Cards should be placed in the hopper face down with the 12 edge toward the operator. Place the plastic weight on top of the deck to prevent jamming as the last few cards are read. If any of the red check lights are on, push RESET. Push START to turn off NOT READY, and the reader is then available to the processor. STOP may be pushed at any time to generate a NOT READY condition, causing the reader to stop at the end of the current card. To continue, push START.

When reading is in alphanumeric mode, every column is checked for a valid Hollerith character. If an invalid punch is encountered and the VALIDITY ON button is lit, a VALIDITY CHECK error is indicated and the reader stops at the end of the current card. If a second attempt to read the card fails, check it for improper punches. Table A4-3 lists the Hollerith character codes and all invalid punch combinations (note that the Burroughs code is incompatible with that used by the IBM 029 Card Punch).

If the reader stops with READ CHECK lit, the self-checking circuits have detected a malfunction in the read circuitry. Usually this is either the failure of a lamp or solar cell and may be temporary. If both READ CHECK and FEED CHECK go on together, the reader is in need of adjustment. There is one section of the read circuitry in which a failure is not detected as a read check error. However, a failure in this particular part would cause incorrect timing resulting in a validity check error if reading is alphanumeric. If no invalid punches can be found on a card, but several attempts to read it result in a validity check error, it is likely to be a malfunction in this unchecked part of the read circuits. For particulars refer to the manual for the Burroughs B122 Card Reader.

When a card cycle begins, the card is first contacted by a knife at the bottom edge of the hopper and is pushed into the read station where the feed operation is taken over by rollers. The most probable point for a feed malfunction to occur is at the entry into the read station. If the card is bent, it may jam in the feedways; if the trailing edge had been damaged by frequent handling, the pickup knife may fail to move the card through the rollers. When a card fails to appear at the read station in the prescribed time, the FEED CHECK light goes on and the drive motors stop. Do not attempt to reread a worn or damaged card that has caused a feed check error, but put a duplicate in its place.

In the unlikely event that a card should jam inside the read station, no FEED CHECK is indicated, but no cards are processed either (if reading is in alphanumeric, a card stuck in the read station may produce a validity check error). To check for jammed cards, remove the head cover by lifting it at its base and pulling it out horizontally (an interlock removes power if it has not already been turned off). If necessary, the photocell head can be removed by moving the knurled-head sliding bolts to their vertical positions and squeezing them toward the center. The head can then be lifted straight up. Be sure when replacing the head that it is seated properly and the knurled-head bolts are fully engaged.

3.3 PROCESSOR OPERATING PROCEDURES

After turning on system power at the processor console, check the memories and peripheral equipment connected to the memory and in-out busses. In general all memories should go on with system power unless a single unit has been taken off line deliberately. Whether a particular peripheral device is on depends upon its own organization. Most in-out control units go on and off with system power; however, in some instances power supplies must be turned on and off independently of the processor.

a Read In

In order to allow initial information to be brought into memory, a readin loader is usually kept permanently in the part of core that is ordinarily inaccessible because of the fast memory. To use the readin loader, set the appropriate starting location in the ADDRESS switches and lift the READ IN key. This turns on the RIM SBR light at the top of bay 1, and while the light is on,

any memory address from 0 to 17 provides access to core instead of fast memory. The light goes out and the processor leaves readin mode whenever an instruction is retrieved from any location above 17. Read in can be single stepped using the CONTINUE and STOP keys, but any other key takes the processor out of readin mode.

To deposit a loader in the bottom of core, the operator must make use of a small toggle switch labeled RIM MAINT, which is mounted behind the double doors on bay 2, on a bracket at the left end between mounting panels 2L and 2M. Putting this switch up holds the processor in readin mode regardless of any action taken at the console, so the operator may deposit the loader. Place the first word in the DATA switches, set the ADDRESS switches to the first location (0 is most convenient), and lift DEPOSIT THIS. MEMORY displays the word deposited. If the remaining words are in consecutive locations, they may be deposited in order by setting them in the DATA switches and pressing DEPOSIT NEXT for each. Although all words are displayed when deposited, it is a good idea to check the entire loader by going through it first pressing EXAMINE THIS, then EXAMINE NEXT. After the loader has been deposited, turn RIM MAINT down.

b Operating Keys

The operator should check material accompanying each program for information on halts, tape requirements, and so forth. Every program is begun by either START or READ IN. On a halt the operator should make note of the console lights, particularly PROGRAM COUNTER, and do whatever is requested in the program operating instructions; the operator may restart by pressing INSTRUCTION CONTINUE.

To debug programs, INSTRUCTION STOP may be latched on and the program single stepped using INSTRUCTION CONTINUE. Or with the REPEAT switch on, and both INSTRUCTION STOP and INSTRUCTION CONTINUE latched down, the program speed can be varied by the SPEED controls. By similar use of the MEMORY STOP and MEMORY CONTINUE keys, a program may be single stepped from one memory call to the next; low-speed operation can be effected by using REPEAT.

The keys for the reader and punch may be operated at any time whether the processor is running or not. The stop keys may also be pressed at any time, but ordinarily these are used only for

single-step or low-speed operation (for special stop considerations see c below). The remaining ten keys use the key cycle to initiate some operation, if only a clear function as is the case with IO RESET. For seven keys, entry into the key cycle is gated by the 0 state of RUN so that inadvertent key manipulation can have no effect while a program is running. Even through RUN remains on throughout memory single stepping (unless of course a program halt should occur), the program can be restarted by pressing MEMORY CONTINUE; at a memory stop MEM STOP is on. Of the initiating keys, only EXAMINE THIS and DEPOSIT THIS have any effect while the program is actually running, i.e., there is neither an instruction stop nor a memory stop. Either of these keys inserts a key cycle between two instructions without stopping the processor. However, do not use these keys if there is any chance of a program halt occurring. The halt instruction stops the processor by clearing RUN at the beginning of the execute cycle, and it is thus possible for a key cycle to be triggered between the time RUN is cleared and the instruction is completed (of course the same caution holds for any wanton key manipulation while the processor is running). If a program halt should occur (RUN goes off) at the same time that EXAMINE THIS or DEPOSIT THIS is operated, check the corresponding pair of SYNC and START lights in the left column on the bay 1 indicator panel. If both lights are on, the corresponding key function was not performed; if neither light is on, the key function was performed prior to the halt instruction. If only a SYNC light is on, the key function was performed but there is no way of knowing whether it was executed during the halt or after, and thus the console lights are meaningless. If EXAMINE THIS or DEPOSIT THIS is pressed with RUN off, the SYNC light does not go out but the key function is performed.

There are also special precautions that must be observed while user programs are sharing processor time. While RUN is on, relocation and protection are inhibited during a key cycle so the operator may use EXAMINE THIS and DEPOSIT THIS with all of memory available to him. However, when the processor is stopped, as between instructions in single-step operation, the operator must make sure his actions at the console are compatible with the operating mode. In user mode any address supplied for a key function, including addresses in an instruction initiated by the EXECUTE key, must be smaller than the block size as indicated by the MEMORY PROTECTION lights at the top of the in-out panel, or a priority interrupt for an illegal address will occur. The operator should also understand that unless the address is for fast memory, it is relocated to the block specified by MEMORY RELOCATION (the RELOCATED ADDRESS is

displayed at the right). Furthermore, an illegal instruction executed from the console will take the processor out of user mode. The operator must observe the lights at the top of bay 1 to determine what he can do. All addresses and instructions are legal if EXEC MODE is on. If this light is off but PI CYC is on (this can happen only between two PI cycles required for the same interrupt), there is also no relocation or protection. However, in this circumstance not all instructions are legal; an IOT may be executed from the console, but any other illegal user instruction or a UUO will return the processor to executive mode. If both EXEC MODE and PI CYC are off, the operator must observe all user restrictions. The operator can switch from either mode to the other by executing the appropriate instruction from the console. The switch may be made from user to executive mode by pressing IO RESET, but this also clears most of the in-out equipment including all PI assignments.

Care should be exercised in the use of the EXECUTE key whenever priority interrupts are allowed while a program is being single stepped. In addition to observing the user-executive restrictions associated with priority interrupts, the operator must be aware of the following. An interrupt has priority over any instruction including one executed from the console. If an interrupt request is waiting when EXECUTE is pressed, the processor performs the PI instruction instead of the one in DATA switches. If PI CYC is on (indicating that the preceding instruction was a block IOT that overflowed), a non-IOT executed from the console will be taken by the processor to be the jump to the break routine. This will turn off PI CYC, and the JSR to the subroutine will be skipped when the program is continued. Furthermore, the processor will "hold" the break—i.e., PI REQUEST goes off, PI IN PROGRESS goes on, and both the channel on which the interrupt was requested and all lower priority channels will remain disabled as the program continues.

CAUTION

> Never under any circumstances press more than one initiating key at a time because the processor will try to perform both functions at once. Note that in low-speed operation one of the continue keys is always on, so although EXAMINE THIS or DEPOSIT THIS can be used while the processor is running, they cannot be used in low-speed operation.

c Emergency Stop

Ordinarily INSTRUCTION STOP is used for single step operation and maintenance procedures, but it can also be used for an emergency stop if the processor should get caught in a loop. For this purpose the turnon of the key triggers a pulse generator whose output triggers a one-shot that temporarily inhibits certain pulses in the instruction and address cycles; then the processor can stop at one of these points if a loop prevents its reaching the end of a main sequence. The pulse generator output also clears RUN in case a hardware malfunction should disable the normal clearing in the execute cycle. Both features apply only to the turnon of the key so that once it has been latched down it will not interfere with single step operation. Once RUN is clear, IO RESET can be used to clear the computer.

If the processor should hang up or be running without seeming to accomplish anything, do not call a DEC Technical Representative until certain routine checks have been made, as it is possible for an inept programmer to hang up the machine. Although it is recommended that only BLKI, BLKO, or JSR be used in a PI cycle, nonetheless the processor will perform a PI cycle correctly for any non-IOT instruction. But a condition IOT will cause the processor to hang up in the PI cycle. When this happens, the processor repeats the instruction over and over: PC is static, the PI REQUEST light for the channel is on but the PI IN PROGRESS light never goes on, and PI CYC and PI REQ at the top of bay 1 remain on indefinitely (the AC lights display the IOT instruction code).

If the program attempts to retrieve an instruction from a memory that is not connected to the bus (and the DISABLE MEMORY switch is off), the lack of any instruction retrieved is interpreted by the processor as a UUO. The attempt to address a nonexistent memory usually results in an interrupt on the processor channel, but if the JSR for the break should attempt to go to the same memory, the processor would go into a loop.

Both of the above loops include the complete main sequence, so pressing INSTRUCTION STOP will cause a stop at the completion of an instruction. There are other program failures, however, that never allow the processor to finish an instruction. If a program should include an XCT that executes itself or if a programmer puts a UUO in location 41, the processor goes into a loop that keeps jumping back to the instruction cycle without ever completing a main se-

quence. The key thus halts these in the address cycle. If DISABLE MEMORY is on and the program attempts any access to a memory that is not on line, the processor hangs up in the memory subroutine. Interrupting the time chain cannot affect this situation, but INSTRUCTION STOP clears RUN, and the machine can then be freed by IO RESET.

Hardware malfunctions can cause loops that the time chain inhibit cannot stop. For example if PI CYC fails to set, an interrupt request will cause a loop in which the processor keeps trying to honor the request without succeeding. If MQ0 refuses to clear in a block transfer, the processor will loop forever, returning on each step to the fetch cycle. For any loop that includes a memory call, do not free the processor merely by clearing RUN and press-IO RESET. If the processor is within the memory subroutine when the reset occurs, it will very likely hang up a memory while freeing the processor. For this situation it is preferable to press MEMORY STOP and then check the lights to be sure the processor has not stopped following the read part of a read-write access. If it has, hold on MEMORY STOP and press MEMORY CONTINUE, so that it will stop following the subsequent write. Once the processor has stopped with no chance of hanging up a memory, press INSTRUCTION STOP to clear RUN, and then press IO RESET.

CHAPTER 4

DRAWING CONVENTIONS AND FLOW CHARTS

Accompanying each PDP-6 is a complete set of drawings, consisting primarily of D-size flow charts, logic drawings (block schematics), and wiring diagrams. Every drawing is labeled with both a DEC drawing number and a type code. The drawing number is in four parts separated by dashes (e.g., D-166-4-EX): the first part is a letter indicating size; the second is the type number of the equipment (usually three digits); the third is the drawing serial number (see next paragraph); and the last is a number or a mnemonic letter code specifying the individual drawing (the code may end with a number, sometimes preceded by another dash, if more than one drawing is required to treat a section of the logic). If a drawing includes several sheets, both the sheet number and the number of sheets are written at the lower left of the drawing number. If a drawing is revised after being signed by the project engineer, a revision letter is written to the right. To the left of the number is a type code; some typical codes are block schematic BS, system diagram SD, flow diagram FD, timing diagram TD, interconnection diagram ID, cable diagram CD, wiring diagram WD, power wiring PW, module list ML, utilization module list UML, master drawing list MDL, cable list CL, wiring list WL (the last three are usually A size).

In general, the only drawings included in the manual for a given piece of equipment are the associated flow diagrams and logic drawings. The maintenance chapter of each manual does, however, describe the other types of engineering drawings and their use. Drawings in the manual are intended for instruction purposes only; personnel working at the machine should use the prints for the equipment rather than the figures for the manual. Drawings that are reduced to B size and printed in this and other manuals for the PDP-6 system are serial 0, corresponding to the standard production machine. Although every unit of a given type is assigned a different serial number, most of the prints accompanying the equipment have drawing serial 0. But if a particular unit differs in some way from standard, those drawings that reflect the difference have the same serial number as the lowest numbered machine that is so modified. Therefore, although each manual contains the drawings for that portion of the standard system that it describes, maintenance personnel should use the prints for work on the equipment because they show any variations peculiar to the installation.

All drawings included in a manual are assigned figure numbers by chapter. These numbers are also written on the prints in the lower right, above the drawing legend. To differentiate drawings associated with one manual from those of another, a code designating the manual appears in front of the figure number (the code may not be included on the figures reproduced in the manual). The letters "AP" indicate a figure for this manual, which includes not only the Arithmetic Processor Type 166 but also the control units for four in-out devices Types 461, 626, 760, and 761. Figures for the memory manual are prefixed "M" and show both the fast and core memories. Figure numbers on drawings for in-out devices described in separate publications are usually prefixed by the appropriate type number.

The complete system logic for the arithmetic processor and the common in-out devices is shown in a series of flow charts for Chapter 4 and a series of logic drawings for Chapters 5, 6, 7, and 8. This chapter describes the conventions and notation used in these drawings.

4.1 LOGIC DRAWINGS

The logic drawings are block diagrams that show the function of every logic element used in the computer. They also show the type of signal present at any module connector pin that carries a logic signal or some special voltage level. The standard power and ground pins (A to D on every module) are not shown. In addition to showing the function of every logic element, the drawings indentify every circuit by type and by physical location. Circuit type is always identified by the type number as given in the DEC module catalog. Below the type number is a location code made up of one digit, one letter, and one or two digits. For example, the location code 2F10 represents plugin unit connector 10 in mounting panel F in bay 2 (in the lettering on the logic drawings, each numeral "0" has a slash through it to distinguish it from letter "O"). Pin designations may be formed merely by adding the pin letter to the module location code, e.g., 2F10H.

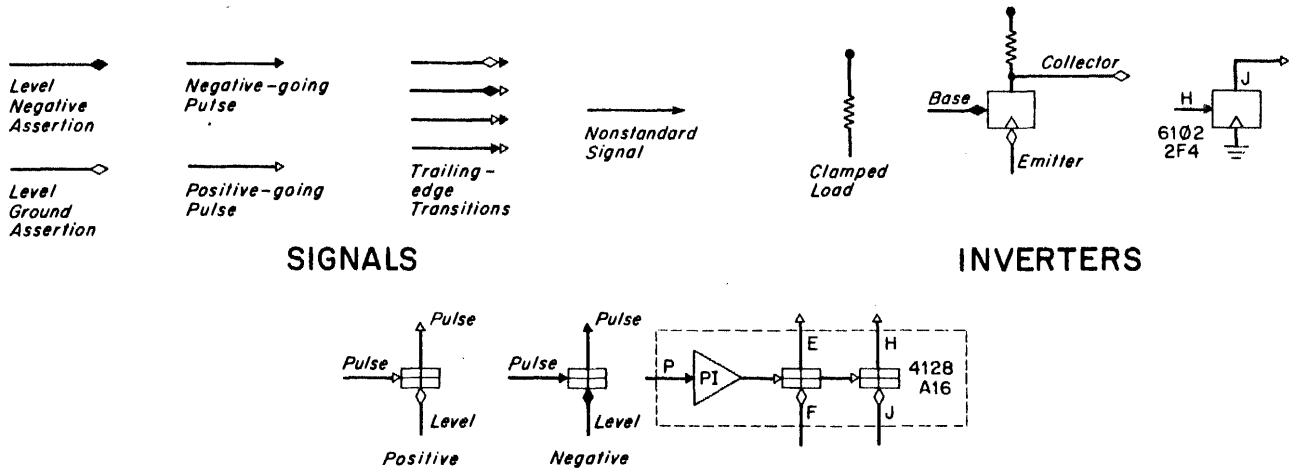
The frame containing the arithmetic processor and the console includes bays 1 to 4. Each memory, or peripheral device requiring a major portion of a bay, is designated as bay 1. If an in-out device control unit requires only a few logic mounting panels, it has no bay number; the panels are designated A, B, ..., even though they may be mounted in any position. For the smaller block symbols, such as those representing single inverters and capacitor-diode gates, the circuit type number and location code are written near the symbol, and the inputs and

outputs are labeled by connector pins. With all larger blocks, the circuit and location information are written inside the block. If several logic elements from the same module appear together in a drawing, they may be enclosed in a dashed line: the location and type number are then written only once within the module boundary and the pin letters are written just inside the boundary where the signal lines cross it.

Some modules have connectors on both front and back; pins on the rear connector are identified by the prefix "R." Some modules are double height, with two front connectors. The location of such a module is given by two panel letters, e.g., 2DE17, and the front connector pins are prefixed by the appropriate panel letter (pins on this module would be designated 2DE17-DT, 2DE17-EB). Only the upper connector receives the power lines, so pins A to D of the lower connector are available for logic signals. Such modules usually also have two rear connectors, the upper one identified by the prefix "R," the lower by "S." Thus for a double-height module mounted in 2DE, the R connector is at the rear of panel D, the S connector at the rear of panel E.

On the logic drawings, the type of signal present at a pin connection is shown by a triangle or diamond. In DEC convention, timing is provided by pulses whose polarities are shown by open and closed triangles. These polarities depend only upon input requirements and represent no logical difference. Similarly, gating levels are represented by open and closed diamonds that represent the assertion polarity that satisfies the gate; neither voltage level categorically represents 1 or 0, true or false. A given logic function may have different assertion levels in different places depending upon gate input requirements. For example, if a function has a negative assertion level, the function is considered true when the line corresponding to it is at −3 vdc; for ground assertion, the function is considered true when the line is at ground. Sometimes a line carrying a logic level is shown connected to the input of a pulse amplifier or capacitor-diode gate, which produces a pulse output. In these cases, the output is triggered by a level transition at the input. If the input is shown as a diamond, triggering occurs at the leading edge, i.e., the diamond shows the assertion polarity of the logic function immediately after the triggering transition. An event triggered on a trailing edge is indicated by a composite symbol with a diamond showing the assertion polarity of the level, and a triangle showing the opposite polarity required for the input pulse (i.e., the triangle indicates the direction of the transition when the logic level is negated). Sometimes a leading edge is shown by a composite diamond and triangle of the same polarity, but this is not necessary. Occasionally in the in-out equipment,

a wide pulse is used to produce a delay by triggering events on the pulse trailing edge. The composite signal for this is a pair of triangles, the first showing the pulse polarity, the second the polarity of the triggering transition. Any nonstandard signal is shown merely by an arrow pointing in the direction of signal flow.



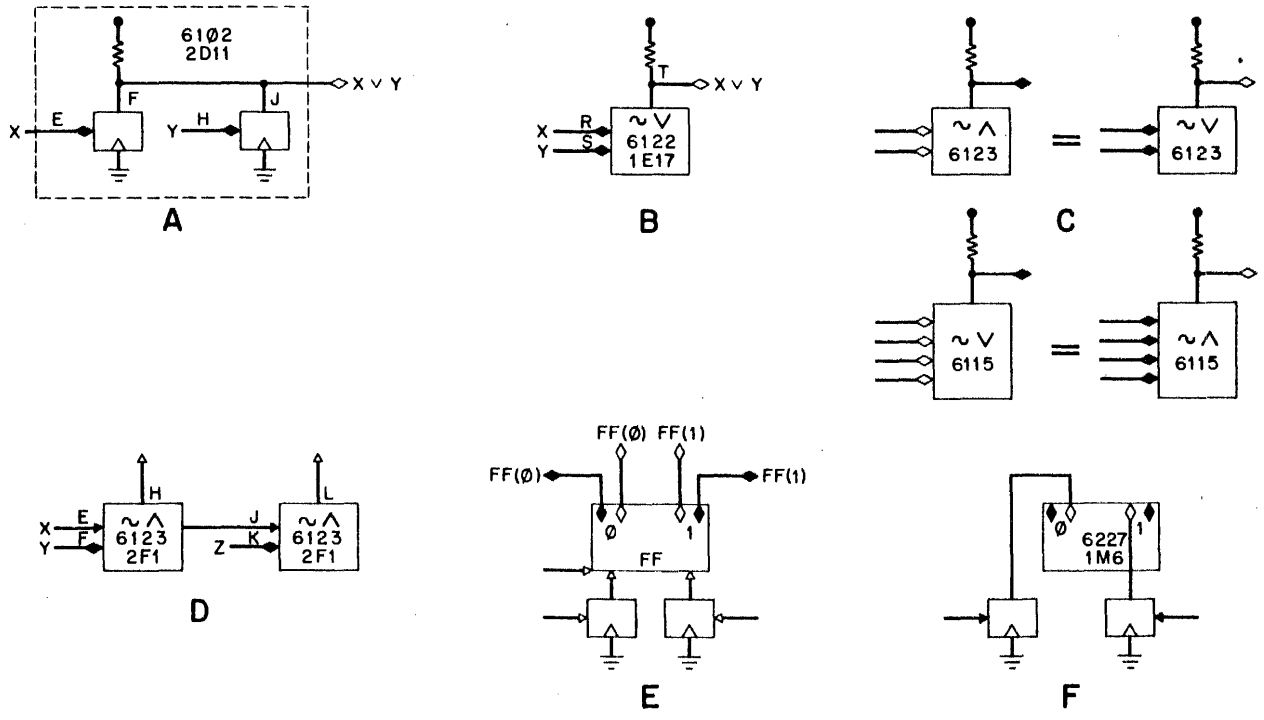


Figure 4-1 Logic Symbols

The upper part of Figure 4-1 shows the signal designations and the symbols for inverters and capacitor-diode gates. These are described fully in the introduction to the DEC module catalog. With the symbols in Figure 4-1 are examples of type numbers, location codes, and connector pin letters. The remaining lettered sections of Figure 4-1 show other conventions for the PDP-6 drawings.

The logic drawings show the function of every logic element in the simplest way consistent with the requirement that every pin connection be shown. Thus if two single inverters are connected to form an AND or OR gate, the individual inverters are shown in the drawing as indicated at A in Figure 4-1. However if the gate is produced by connections internal to a module, such as a pair of inverters with a common collector pin and internally grounded emitters, then it is shown by a block labeled for the appropriate logic function as shown at B. Blocks are used to represent inverter gates with as many as four inputs and diode gates with as many as eight (the in-out equipment uses capacitor-diode modules in which the gates have common pulse connections, but the individual gates are always shown). Since all such nets invert (a diode gate includes an inverter output) the blocks are always labeled "$\sim\wedge$" or "$\sim\vee$." The tilde ($\sim$) in these labels has no actual connection with logical function and may be ignored when learning the logic. Because an assertion level may be inverted without affecting its truth value, all gates are AND ($\wedge$) or OR ($\vee$) gates, and the tilde merely indicates that the output assertion level is of opposite polarity to the inputs. Each block also contains a location code and a type number; input and output pin connections are labeled in the usual way. Since the logical function of a gate depends both upon its logical configuration and the assertion polarities of its inputs, a given net may be used as either an AND or an OR gate—the equivalence is shown at C. Of course, a single level input may be replaced by a pulse in an AND gate, and all levels may be replaced by pulses in an OR gate. Inputs to a gate are generally at the left, outputs at the top, and a signal shown passing through a gate from left to right implies no logical change: D shows a pair of AND gates in which the pulse X, though labeled only once, is an input to both gates, whereas levels Y and Z each enter only one.

All other circuit elements except flip-flops appear as blocks that contain a mnemonic abbreviation of the circuit function. Some examples are delay DLY, pulse amplifier PA, pulse generator PG, clock CLK, bus driver BD, solenoid driver SD, majority gate MAJ, dc adder DCA.

Clocks, pulse amplifiers, pulse generators, and some one-shot delays have a pair of transformer-coupled pulse outputs (usually shown at the right of the block). When the input to such a circuit is triggered, a positive-going pulse appears at the positive output if the negative output is grounded, or a negative-going pulse appears at the negative output if the positive output is grounded. One-shot delays other than the 4303 have a logic level output that is asserted negative during the delay period. The symbol that represents a flip-flop is shown at E (the same type of symbol also represents the 4303 Integrating Delay and the 6131 DC Adder). In this rectangle, terminals S and T are drawn twice, showing the polarities associated with either state of the flip-flop. In normal convention the "0" is at the left and the 0-out terminal is represented by the left diamond in both pairs. Some flip-flops have a separate output terminal, represented by a fifth diamond, to drive an indicator.

The principal advantage in showing the two states at both assertion levels is that there is never any need to invert the name of a signal that appears as an input to a logic net: all logical conditions appear in the drawings with correct truth values. When a flip-flop output is used as the input to a logic net, the signal name indicates the enabling state of the flip-flop. To determine the physical source of the signal (the output terminal to which the signal line is connected), one must know both the signal name and the assertion level. For example, the signal FF(1) at negative assertion originates at the 1-out terminal of flip-flop FF; at ground assertion this signal actually originates at the 0-out terminal.

Two gatable inputs are shown at the bottom of the rectangle at E, with the 0-in terminal at the left. Direct pulse inputs (i.e., those that are not gated) are at the sides of the flip-flop: in the example a direct clear input is shown at the left. A flip-flop may also have a complement input, which is drawn at the bottom center. An unbuffered flip-flop may be set or cleared by grounding one of the flip-flop collectors; such a function may be represented in either of the ways shown at F.

Connections between flip-flops are shown in various ways on the logic drawings, always in the simplest way consistent with showing all pin connections. The clear line for an entire register is usually shown entering the lower left corner of the leftmost flip-flop, then out of the lower right corner on to the next flip-flop, and so on through the register. If the flip-flops in a

counter have count inputs and outputs, the count pulse is usually shown going from right to left, entering the center of the right side of each flip-flop and leaving at the center of the left (e.g., see the program counter, Figure 5-11). If flip-flops from different modules are connected for shifting, all of the shift gates must be shown; however, if the flip-flops within a module are connected internally as a shift register, the shift signal is shown in the same way as a count (e.g., the shift register modules in Figure 8-7). In all processor registers, most transfer input gates for a given flip-flop are included on the module containing the flip-flop. These gates are therefore shown as small rectangles with logic symbols and pin connections but without location codes or type numbers (refer to AR, Figure 6-5). In many instances no pin numbers are included because the input connections are internal to the module: e.g., the AR, MB, and MQ flip-flops are on the same modules and the connections shown between them have no pin numbers. If these internal logic gates are similar to the regular inverter and diode gates, i.e., if all inputs to a given gate are of the same polarity and the output is inverted, the block is labeled with the appropriate symbol, either "$\sim\wedge$" or "$\sim\vee$." However, in many cases there are nonstandard gates, e.g., one in which a ground level is gated by a negative pulse to produce a ground output. Such gates are labeled merely by logic function, "$\wedge$" or "$\vee$," and no attempt is made to indicate signal polarities other than the diamonds and arrows used for inputs and outputs. In some cases pulse inputs to individual bits of a register are made through NPN emitter followers. Since these perform no logical function and do not even change signal polarity, each is shown on the logic drawing merely as a small circle at the pin to indicate that the signal at the flip-flop input terminal through other input gates is not available at that pin.

State changes in the 10-mc flip-flops take place more rapidly than the duration of the input pulses. To compensate for this, many gated inputs to these flip-flops are made through delay elements, which are not shown on the logic drawings. Since in many instances the outputs of a register flip-flop condition its inputs, the flip-flop state change is delayed until the termination of the input pulse to prevent logical race problems.

In addition to the many modules containing flip-flops, pulse amplifiers, etc., there is also a hybrid module, the subroutine card SBR, which includes three circuits each containing a flip-flop and a gated pulse amplifier. The flip-flop 1 output is an input to the AND gate at the PA input; the PA output, besides being available at a connector pin, is connected internally to the

flip-flop clear input (see the SBR at the left in Figure 5-2). An SBR (or its equivalent) is used whenever a subroutine is called from any time chain. The same pulse that triggers the subroutine also sets the flip-flop in an SBR, enabling the input gate to the PA; the other input to the PA gate is the return pulse from the subroutine. At the completion of the subroutine, the return pulse triggers the PA, whose output both clears the SBR and restarts the time chain.

All logic drawings are laid out with rectangular map coordinates, numbered 1 to 8 from left to right and lettered A to D from top to bottom. Because a single drawing may contain a number of logic elements, coordinates are often included in figure references. For example, a reference to the circuit in "Figure 5-6B6" would mean the circuit located in block B6 of logic drawing 5-6.

4.2 SIGNAL NOTATION

All signal names in PDP-6 are mnemonics that indicate both the function of the signal and its source. Each register with associated logic and each control system, whether it occupies several drawings, one drawing, or only part of one, has a single mnemonic code of one to three letters, which appears in the drawing title and at the beginning of the name of any signal originating in this part of the logic. This source code may appear naturally as part of the signal name; if not, it is merely prefixed to the name. For example the arithmetic register AR and its associated transfer logic, flag logic, and AR subroutines require a number of drawings all with prefix code AR, and the pulse that shifts the contents of AR to the left is AR SH LT. On the other hand, the readin mode subroutine flip-flop, which is associated with read in and is shown on one of the drawings for the key logic, is designated KEY RIM SBR. All prefix codes and corresponding figures are listed at the left in Figure 4-2.

The name of a signal that transfers information from one register to another includes the names of the two registers with an arrow between them. The arrow invariably points to the left because the transfer logic is always associated with the receiving register and its name must therefore appear first in the signal designation. The name of a transfer signal specifies not only the registers and the direction of transfer, but also the type of transfer and the register bits involved if the signal acts on less than the entire register. Numerals representing register bits are merely appended to the register name; bit 8 in AR is AR8 and bits 0-7 in MB are MB0-7. Operations

that affect only half of a full-word register are indicated by appending LT or RT to the register name. The state of a flip-flop is represented by a numeral in parentheses, e.g., the 1 state of bit 8 in AR is AR8(1). Since transfers in effect transfer states, the type of transfer is also indicated by a symbol in parentheses following the register name. For example the signal that transfers all zeros into the bits of AR, i.e., the signal that clears AR, is AR ←(0). This action of course is not usually referred to as a 0 transfer but rather as the clear function. The actual transfer of zeros from register A to register B is as follows: the transfer pulse clears a given bit of B if the corresponding bit in A is in the 0 state. The pulse that produces this effect is B←A(0). Since B now contains ones only in bit positions that originally contained ones and also correspond to ones in A, the 0 transfer therefore produces in B the AND function of A and B. Similarly, the transfer of ones from A to B, B←A(1), produces the inclusive OR function. If B is cleared before the transfer, then after the transfer A and B both contain the same information and the pair of pulses B ←(0) and B ←A(1) transfer the contents of A to B. The same effect would be produced by setting all bits in B and then transferring zeros. If both the zeros and ones of A are transferred to B simultaneously so that B bits corresponding to zeros in AR are cleared and those corresponding to ones in AR set, no prior setting or clearing is necessary. This is a jam transfer and is written B←A(J). Since the jam transfer occurs at a single point in time, it is possible to switch the contents of two registers: the outputs from A can provide the gating levels for the transfer into B while the B outputs gate the transfer into A. The signal that triggers both transfer pulses at once is labeled B(J)↔A(J).

There are other types of transfers, such as B(0)←A(1), which clears bits of B that correspond to ones in A, i.e., transforms ones into zeros. The pulse that complements a given bit in B if the corresponding bit in A is 1, produces in B the exclusive OR function of A and B, and is written B←A(∀). Most other types of signals have fairly obvious names: MQ SH RT shifts the word in MQ to the right (a shift is a jam transfer from one bit to another in the same register); AR COM complements the word in AR. We have been discussing pulse signals so far but the names of gating levels are also quite descriptive: PC+1 INH inhibits incrementing the program counter; FC(E) PSE causes memory control to fetch the word in location E and then pause to wait for a restart for subsequent storage; SAC2 causes the store cycle to deposit a second accumulator. AC0 always refers to accumulator 0, but AC2 refers to the accumulator following the one addressed by the instruction (if the instruction AC address is 17, AC2 refers to accumulator 0). No more than two sources can have the same signal name, and any pair must have opposite

polarities so that the source of every signal is uniquely identified by its name and the associated polarity symbol. Whenever two logically equivalent signals have the same polarity, they are differentiated by adding an extra letter or number to one of the signal names; for example, MR CLR and MR CLR A are equivalent pulses. If one logic signal produces an equivalent signal through a bus driver, the buffered signal is indicated by the letter "B" at the end of the signal name. If the outputs of a flip-flop are buffered externally to the flip-flop module, the buffered signals are indicated by a "B" between the flip-flop name and the state numeral.

The pulses in the time chains for the various main cycles and subroutines all have the same three-part format: first the prefix code naming the chain, then the letter "T," followed by a number or number and letter combination specifying the pulse. For example, FT3 is pulse number 3 in the fetch time chain; DST13 is pulse 13 in the divide subroutine time chain. These pulses are not always in exactly the order that one might expect, and the reader should always consult the flow charts to determine the proper sequence. For example the first three pulses in normalize return are NRT0.5, NRT0, NRT0.1. In the execute cycle, ET0 and ET0A are logically equivalent and together are the first pulse in the execute cycle—the two labels indicate separate but equivalent pulse lines. The next pulse is ET1 but this is followed by ET3. In character operations CHT8B follows CHT8 but precedes CHT8A. But in most cases a letter following the number in a pulse name indicates the next pulse in the chain, usually following return from a subroutine. Thus the first pulse in the floating multiply chain, FMT0, calls the exponent calculate subroutine, and the next pulse, triggered by the subroutine through an SBR, is FMT0A. The SBRs and most control flip-flops that govern the time chains also have similar three-part names in which the "T" is replaced by an "F."

4.3 INSTRUCTION DECODING

Figure 4-2 is a tree which shows the decoding of instructions from the instruction register. The output signals in the figure appear as the gating levels in the flow charts that are described in the following sections. The purpose of the tree is to allow the reader to gain familiarity with these logic gates and to correlate them with the instruction codes. No attempt is made here to give a detailed explanation of every logical function; for this the reader should use the tree in conjunction with the discussion of the decoding hardware in 5.3.

Any code placed in IR is converted into gating levels to govern events that must occur in the fetch, execute, and store cycles and various special sequences to execute the particular instruction. The decoding begins at the left in the figure with the three most significant IR bits, which are converted into signals representing the eight instruction classes. These primary outputs act as gates to enable decoding of the remaining bits. In some cases the output enables a second decoder for several more bits whose outputs in turn act as gates for further decoding. In others the first output represents a single instruction group and it gates the decoding for all remaining bits. For the former type, the line extends only part way across the figure and then generates a number of branches at a single position; for the latter the line extends the length of the tree, and branches appear at several positions. Most of the groups of two, three, or four bits shown together are decoded by binary-to-quarternary or binary-to-octal decoders.

Groups of bits are actually decoded into individual outputs only where signal names are shown for each bit configuration. Where only parenthetical items are listed, the coding as shown produces the mode or action listed, but the hardware does not decode the bit configuration into individual outputs.

When IR receives an instruction code in the instruction cycle, the first three bits are decoded to generate one of eight primary command levels shown at the left in the figure. At the top is the decoder output for a UUO, which corresponds to three zeros in bits 0-2. The decoder output is conditioned by a flip-flop to prevent the generation of the actual command level while IR is clear awaiting an instruction code. Other conditions also generate the command level to perform a UUO when a user program attempts an illegal instruction. At the bottom of the drawing is the decoder output for an IOT, corresponding to three ones. If it is not a user IOT, the command level is generated and causes the decoding of bits 10-12 into the eight IOT instructions. Some of these individual instructions are ORed to generate composite functions. At the same time IR bits 3-9 are placed on the I/O bus to select the device.

Between the top and bottom entries in the figure are the decoder outputs for the six instruction classes that use the basic format (these are not in numerical order, but are instead listed so that classes with common decoding are adjacent). The configuration 001 indicates the floating-point instructions and character operations. If the second octal digit in the code is 0, 1, or 2, there is no further decoding as these 24 codes are not used. If bits 3-5 contain 111, IR6-8 are

decoded for two unused codes, the five character operations, and the single instruction floating scale. A 1 in bit 3 indicates the floating-point instructions and for these, bits 4 and 5 are decoded for the specific instruction, bits 7 and 8 for the mode. The logical condition for rounding (NR ROUND) is dependent upon bit 6, but is not dependent upon any command level because the gate is used only by the normalize return subroutine, which is called only by floating-point instructions.

The second basic instruction class corresponds to the primary command level IR 2XX, which represents a number of small instruction groups. Either 0 or 1 in the second octal digit specifies a full-word transfer for which bits 7 and 8 are decoded for the mode, and specific configurations of bits 5 and 6 generate the levels that control swapping and negating of the word. Second octal digits of 2 and 3 correspond to fixed multiply and divide, respectively. The 01 configuration of bits 3 and 4 (shown as 01-) generates a composite multiply-divide level to enable the net that determines the necessary fetch and store operations from bits 7 and 8. Further fetch and store operations are determined separately for multiply and divide from the state of bit 6, which specifies whether the operands are to be treated as integers or fractions. For division bit 6 is actually decoded into a further pair of control levels, but for multiplication the outputs of IR6 act as gates directly on the multiply sequence. The next three configurations of IR3-5 enable the decoding of IR6-8 for the shift operations, a group of miscellaneous instructions, and the pushdown and jump instructions (the absence of a signal name by any configuration of IR6-8 corresponds to an unused op code). The decoder output for JRST, like IOT, generates the true command level only if it is not executed as a UUO. For the JP group there are two additional control levels corresponding to the six instructions that actually execute a jump, and the three instructions that store the miscellaneous bits. The last group of instructions in IR 2XX are fixed addition and subtraction wherein IR6 determines the instruction, and bits 7 and 8 are decoded for the mode. These modes are equivalent to those for the Boolean instructions, which make up an entire class with the configuration 100 in IR0-2. IR BOOLE also enables the decoding of IR3-6 to determine which of the 16 instructions is specified.

Another single instruction group, the half-word transfers, is specified by 101 in the first three IR bits. For these instructions bit 3 determines which part of the destination register shall receive the half word. Bits 4 and 5 determine the effect on the nontransfer half: a 1 in either bit clears AR, and further decoding enabled by bit 3 determines whether that half shall then be

set to all ones. A 1 in bit 6 causes the operand to be swapped before the transfer so that the destination register receives the half word from the opposite half of the source register. Bits 7 and 8 are decoded for the mode.

The remaining two IR0-2 configurations, 011 and 110, are for the arithmetic compare instructions and the logical compare instructions (ACBM). The former class includes the ACCP group in which an accumulator is compared against either C(E) or E, and the MEMAC group in which either AC or C(E) is compared against zero. For the arithmetic class, 00 in IR3-4 specifies the ACCP type, and a 1 in bit 5 indicates a direct comparison, i.e., against C(E); the other three configurations of bits 3 and 4 specify the types of MEMAC instructions, any of which generates the composite MEMAC command level. The state of IR5 determines whether the comparison tests AC for a jump or memory for a skip. In the ACBM group, bits 3 and 4 determine the action on the masked bits. A 1 in bit 5 specifies that the mask is C(E), and a 1 in bit 8 swaps the mask before the test. The remaining bits in the two classes determine the skip or jump condition as shown; for the logical comparison the condition is determined by bits 6 and 7, for the arithmetic comparison by bits 6-8. The level for ACCP or MEMAC appears in the term for bit 8 because this part of the condition is not used by the ACBM instructions; otherwise no instruction levels need appear because the test level generated is used only by these instructions.

4.4 FLOW CHARTS

The remaining figures in this chapter are flow charts of all operations that can be executed by the arithmetic processor and the four common in-out devices. These figures show every event, and in so far as possible, show the flow of operations in a manner that is equivalent to the actual gating and timing in the hardware (the terminology is from the logic drawings unless italicized). Certain intermediate pulses are shown only implicitly (for example ET1 is shown clearing AR, when actually it triggers AR ←(0), which in turn triggers ARLT ←(0) and ARRT ←(0), which together clear both halves of AR). If an event is prefixed by CFAC, the time pulse is routed via the subroutine interface to the register gating.

Each flow chart is based on a sequence of time pulses shown along a vertical line. Except for insignificant intervals, such as the delay across a pulse amplifier or inverter, time between pulses is shown by breaks in the line. Pulses always appear in ellipses and events in rectangles.

A pair of single horizontal lines breaking a vertical line indicates a delay; between the lines is listed the delay time, or the signal that must be asserted to continue the flow. A break shown by double lines indicates a subroutine call; the upper term identifies the subroutine, the lower term names the pulse returned by the subroutine to restart the calling sequence. A flow line that terminates with an arrow indicates that the flow continues with the pulse listed below the arrow (numbers in parentheses are figure references); a line that terminates in an ellipse indicates that the flow along this path ends with the events associated with the pulse. In a rectangle, a condition written to the left of a colon must be satisfied in order for the event written to the right to occur. A condition written on a line must be satisfied for flow to continue along the line. When several vertical lines branch from a horizontal line, the conditions are written above the vertical lines.

The key cycle, through which initial entry into processor operations must be made, is shown at the left in Figure 4-3. Key functions that require a sequence of events are shown in columns associated with the key time pulses. Some console operations, such as examine and deposit, require only the key cycle; others, such as start and read in, use the key cycle to provide entry to the main sequence. Figure 4-4 and 4-5 show the main sequence. The flow charts for the instruction, address, fetch, and store cycles show all possible events; the execute flow chart shows the main time chain for the cycle, including all exits to special instruction sequences and all operations that are not produced by any instruction, e.g., clearing AR CRY0 at the beginning of the cycle. The basic memory and AR subroutines can be called from the main sequence and are shown, respectively, in Figure 4-3 and at the right in Figure 4-8.

The remaining figures show the execution of instructions following the fetching of the operands. Those instructions that are timed by the execute cycle are shown in tables in Figure 4-6, 4-7, and at the left in 4-8; instructions, other than IOTs, that are executed by special sequences are shown in Figures 4-8 to 4-10. These special sequences may call the basic SC subroutines (Figure 4-8, right) and the special arithmetic subroutines shown in Figures 4-11 and 4-12. Figure 4-13 shows the IOT instructions, some of which require the entire execute cycle as well as the special IOT sequence. In-out operations that take place outside of the processor are shown in Figure 4-14; for each of these an IOT is required at the beginning or the end, but otherwise the sequence goes on independently of processor operations.

For every instruction, Figures 4-6 to 4-10 and 4-13 list the complete decoding, the instruction action, the initial registers, and the initial and final gates. The decoding is shown as performed by the hardware; whenever the decoding does not correspond exactly to the meanings of the bits in the instruction codes, further explanation is given in italics. The action is given in the programming sense (i.e., in terms of accumulators, memory locations, control registers, flags), but in the notation of the logic drawings: AC may mean either an accumulator or its contents, E may mean either an address or the addressed location; the meaning is evident from the context. The states of the processor registers are shown following the fetch cycle; MA always contains the effective address and is not shown. The initial gates are those that control the fetch and execute cycles. It should be understood that PC is incremented at ET0 whenever the PC inhibit is not shown as an initial gate, unless there is some special situation (e.g., the processor is in a priority interrupt cycle or the instruction is being executed from the console). For all instructions that are executed by the execute cycle, or return to it, the final gates (i.e., those that control the store cycle) are listed at the end of the sequence. In all cases, registers and gates are listed in terms of the instruction modes even though the modes for fixed multiply and divide are not decoded individually.

Since the complete timing for the execute cycle is defined in Figure 4-5, those instructions whose events are timed entirely by the execute pulses are shown in tables. In a table entry for ET3, double vertical lines indicate the call of an AR subroutine; the subroutine is at the left, the return pulse at the right. The pulses from ET6 to ET9 are omitted when the instruction does not use the second half of the cycle.

4.5 EXECUTE CYCLE FLOW

Every main cycle, subroutine, and special instruction sequence is explained with the description of the logic that generates the time chain and gating levels for it. However, for instructions that are performed by the execute time pulses it is impossible to consider the multitude of possible events in the description of the execute cycle. For most such instructions the reader can easily determine how the specific sequence of events produces the desired result by inspecting the tables in Figures 4-6, 4-7, and 4-8. A few of the less obvious sequences are, however, described here (in the following it is assumed that the reader is familiar with the action of each instruction).

In a half-word transfer (Figure 4-6, left) the positions of the source and destination words after the fetch cycle depend upon the mode; i.e., the fetch cycle always places an accumulator in AR and a word from memory in MB, and the mode specifies which of these is the source and which the destination. The execution of the instruction requires that the source word be in MB and the destination in AR, so ET0 switches MB and AR for the memory mode and transfers MB to AR for the self mode. If the transfer is to be from one half of the source to the opposite half of the destination, then ET1 swaps the two halves of the source in MB. If the instruction is to perform any operation on the unused half of the destination, ET1 also clears AR. Then ET4 jams the specified half word into the appropriate half of AR and completes the action on the other half by setting it, if required. At ET10 the result is transferred from AR to MB for either the memory or self mode, i.e., those modes that require the store cycle to deposit the result in location E. In most instructions the result is in AR at the end of the execute cycle and thus any mode requiring storage in E usually includes the AR to MB transfer at ET10. Exceptions include instructions such as EXCH in which the correct words are already in MB and AR at ET0.

In a full-word transfer, ET0 performs whatever transfer is necessary so that the word to be moved is in both MB and AR. Then if the halves are to be swapped, the swap is made by ET1 in MB; if the word is to be negated—which may happen for either MOVN or MOVM—the AR negate subroutine is called at ET3. By ET4 the result is in MB only for MOVS, so for this instruction MB and AR are switched. Of course only a transfer from MB to AR would be necessary, but the gate for the bidirectional transfer is required for other instructions (no instruction requires a transfer limited to one direction). The other instructions in Figure 4-6 are quite straightforward with the possible exception of the logical compare group, ACBM. At the end of the fetch cycle, the word to be used as the mask is in MB, and the word to be tested in AR. For convenience let us refer to the latter as the data word, and the AND function of it with the mask as the test word. ET0 completes the construction of the mask by swapping the two halves of MB, if this is necessary. ET1 then produces the test word by transferring zeros from AR to MB, and at the same time it adjusts the data word for the complement and set actions by complementing bits of AR corresponding to ones in MB (the exclusive OR function) or transferring ones from MB to AR. ET4 then moves the test word to AR and the data word to MB. At this point the clear action has not been handled and it would seem that the mask has been lost. However, any ones now in AR must be a subset of the ones in the original mask, and these correspond exactly to all the ones

within the masked bit positions in the data word in MB. For the clear action the next three time pulses clear these bits by complementing AR, transferring zeros to MB, and complementing again to restore the test word. At ET9 the test is made according to the contents of AR, and the adjusted data word is transferred into AR at the same time so that it is available for subsequent storage in AC. The bidirectional transfer listed in the table is not necessary, but is available because of the requirements of other instructions. The store cycle deposits the data word in AC unless it has not been changed from the original.

In JRST (Figure 4-7), the second half of the execute cycle performs a series of operations that seems to be pointless, since the store cycle does not deposit anything. Pulses 5 and 6 transfer the current program address from PC to MB, and after the jump address has been transferred to PC, pulses 9 and 10 transfer MB to MA. If the program does continue, these operations are pointless, but the JRST can halt the processor and may be used for error halts in maintenance programs. When the processor stops, PC displays the jump address and MA displays an address one greater than that of the location that contained the JRST.

CHAPTER 5

MAIN SEQUENCE CONTROL

This chapter describes the processor response to console inputs and explains the way in which the main control elements operate on instructions. Of the five major cycles, four are described in detail here: retrieval of an instruction from memory and decoding it from the instruction register; calculation from memory and decoding it from the instruction register; calculation of the effective address; retrieval of the operands; and storage of the result. The execute cycle is discussed only to the extent of basic operations, and timing events required for the execution of individual instructions are described in other chapters. The many arithmetic and data subroutines are described in Chapter 6 with the arithmetic registers; Chapter 7 describes the memory subroutine and memory address and data transfers; Chapter 8 includes the decoding of IOT instructions and control over transfers between the processor and input-output equipment. The various test conditions that affect the program sequence are discussed here, but any arithmetic or logical operations necessary to produce them are described in Chapter 6.

5.1 CONSOLE CONTROL

In general, the console switches provide continuous control levels that gate certain operations within the processor; whereas the keys are momentary contact switches that allow the operator to trigger specific sequences of events or to stop the processor. The hardware that provides control and timing for the various console operations is shown in Figures 5-1 and 5-2 (flow chart, Figure 4-3).

When the operator turns on computer power at the console, the 4303 Integrating Delay in the lower right of Figure 5-1 comes on in the 1 state, turning on a clock that provides a string of master power clear pulses. These pulses clear RUN so the processor cannot begin operations inadvertently, and they also trigger the master clear and the master start (Figure 5-2, right). MR START clears the flags and the in-out system including all flip-flops in the priority interrupt system and the processor I/O interface; MR CLR clears the major control registers and flip-flops

in the processor proper including all SBR flip-flops. The master clear also occurs at the beginning of every new operating sequence, i.e., at the beginning of every console operation other than memory continue (see below) and at the initiation of any new instruction (IT0 or any equivalent pulse). The master start is generated only by the power clear and when the operator specifically wishes to clear the I/O system or place the processor in the executive mode. The flags and several special control flip-flops are cleared only by MR START because they must carry information over from one main sequence to another.

The left half of Figure 5-1 shows the logic inputs from the keys and switches, and nets that generate a number of composite functions to control events common to two or more console operations. Several of the nets AND key functions with a state of the RUN flip-flop. For example, a net in A3 allows the input from the EXECUTE key to affect the processor only if it is not in operation. Since examine and deposit may be performed while the processor is running, these two operations are governed not by levels from the keys but rather by two pairs of sync and start flip-flops (upper right). A pulse in the key time chain, KT0A, normally clears all four flip-flops, but for examine or deposit the appropriate sync flip-flop is set instead. Then, if the processor is running, the corresponding start flip-flop is set at the beginning of the execute cycle. The sync flip-flops provide gates for the events in these operations; the start flip-flops control the insertion of the operation between two main sequences. For this purpose, two of the nets at the left generate the AND functions of RUN (1) with functions of the start flip-flop states. If at the end of a main sequence, either flip-flop has been set and the processor is to continue, the signal from the net in C5 causes the processor to enter the key cycle at the end of the main sequence. However, if neither flip-flop has been set, the signal from the net in C3 causes the processor to begin a new main sequence.

Pushing any key other than the two STOP keys generates the level KEY MANUAL through the net in the top center of the figure (the entire net is disabled during the power clear period). The turnon of this level triggers a pulse generator in the upper left of Figure 5-2 to start the key time chain. This chain is broken between KT0A and KT1 by the condition RUN (0) so that KT0A triggers the following triplet of pulses only if the processor is not running. There are, however, two gates that bypass the RUN condition: if the operator selects EXAMINE or DEPOSIT while the processor is running, the appropriate start flip-flop allows the final pulse in a store cycle to trigger KT1; furthermore memory continue goes directly from KT0A to KT1

because it is assumed that the processor is "running"—i.e., RUN is 1—but that the operator has previously stopped it at the end of a memory cycle by inhibiting the memory subroutine return. This is why memory continue is the only operation in which KT1 does not generate the master clear: somewhere an SBR is legitimately waiting for the subroutine completion. Memory continue triggers the MC restart at KT1.

For read in, KT1 sets the readin mode subroutine flip-flop shown at the lower left. When the processor is operating in readin mode, the normally unused core registers at the bottom of memory replace the 16 flip-flop registers of the fast memory. In all other respects, read in is exactly equivalent to start; thus the operator may keep a loading program stored permanently in the readin area of memory, and the processor switches to normal operating mode whenever an instruction is taken from any location beyond 17 ($\sim$ MA18-31 = 0). For console operations other than read in, KT1 clears the flip-flop for normal operation, but the clear is inhibited by the CONTINUE keys so that READ IN can be single stepped (the clear net is in Figure 51D1).

A few of the functions produced by the key time pulses are shown on drawings associated with the appropriate function (such as operations involving PC) but most are shown in the center portion of Figure 5-2. Start, read in, examine, and deposit all clear MA and transfer an address into it from the console ADDRESS switches. Examine next and deposit next, instead of loading MA, increment it. The three operations that send data into the computer, deposit, deposit next, and execute, clear AR and load it from the DATA switches. Also shown in the key drawing is the gate through which the DATAI for the processor loads the DATA switches into AR. CPA indicates that the IOT has selected the processor and the assertion of the DATAI level triggers the transfer PA (see 8.3).

Following the various setup operations, start, read in, and instruction continue all generate KEY GO (upper right). This pulse places the processor in operation by setting RUN (Figure 5-2) and starting the instruction cycle. As long as RUN remains set, the completion of every store cycle triggers a new instruction cycle so that the processor performs one instruction after another in the program. Either the operator or the program may stop the computer at the end of a store cycle by clearing RUN at ET0. The executive program clears it by a 1 in bit 10 of a JRST; a user program cannot stop the processor. The operator clears it by pressing the INSTRUCTION STOP key, but the logic gate is bypassed by a pulse generator that is also connected

to the key. The action through the PG is slow compared to the processor time chains, but this ensures that RUN will be cleared even if some malfunction should interrupt the normal procedure.

The other console functions do not place the processor in normal operation. Execute transfers AR to MB at KT3 and enters the instruction cycle at the point at which an instruction would normally be in MB after retrieval from memory but it does not set RUN, so the processor stops at the completion of this single instruction. In the four examine and deposit operations, KT3 requests a memory read or write as required and sets KEY RE/WR (Figure 5-2 B6, C1). To deposit information in memory, KEY WR also transfers AR to MB. Upon completion of the memory subroutine, the MC restart generates the read/write return clearing the SBR.

The upper right portion of Figure 5-2 shows the gates that generate KT4 for the repeat mode. If the REPEAT switch is on, KT4 triggers a repeat delay (upper left): the 4303 state change at the end of the delay period then retriggers the key time chain as long as any initiating key is held on. To use the memory repeat function, the processor must already be running when the operator presses the STOP key for memory; then KT4 is generated every time the memory stop flip-flop is set. To repeat start, instruction continue, or read in, the operator must hold on both the initiating key and the INSTRUCTION STOP key; then at the completion of each instruction, ST7 triggers KT4. Since execute does not start the processor, the operator need not use the STOP key to repeat it; in this case, KT3 triggers KT4. Similarly, for the four examine and deposit operations, the read/write return triggers KT4. Furthermore, examine or deposit may be repeated while the processor is running: if one of these operations interrupts the normal transition from store cycle to instruction cycle, the appropriate start flip-flop causes KT4 to trigger KEY GO. This pulse does not affect RUN, which is already 1, but it does retrigger the instruction cycle and clear the start flip-flops.

5.2 PROCESSOR CYCLES

There are five cycles in the main sequence: instruction, address, fetch, execute, and store. All but the execute cycle perform only operations that are common to all instructions or to groups of instructions, and these four are described in detail in this section. Instruction, fetch, and store can call only the memory subroutine; whereas the address cycle can call both the memory and the add subroutines.

The execute cycle performs some general control operations but is limited primarily to the specific operations necessary for the execution of individual instructions, including entry into special instruction sequences. For this reason, the description of the execute cycle included here is limited to timing, general control functions, the calling of the simple AR subroutines, and the entry and return for special sequences.

In the fetch, execute, and store cycles, the gating levels that govern the sequence of time pulses and the calling of subroutines are OR functions of instruction conditions. An input condition may be an entire instruction class or a single instruction, or it may be a single mode either within an instruction or in an instruction class. The reader is assumed familiar with the standard outputs from the instruction decoders (4.3) and only special conditions are discussed here.

a Instruction

The lower half of Figure 5-3 shows the time chain that controls the retrieval of instructions from memory (flow chart, Figure 4-4). When the operator starts the processor or causes examine or deposit to interrupt normal processor operations, KEY GO begins a new instruction cycle by triggering IT0. If the processor is in normal operation and no examine or deposit has been synced by key timing, the final pulse in the store cycle of an instruction starts the instruction cycle of the next one. IT0 generates the master clear, clears the memory address register, and after a slight delay sets IF1A, the SBR for the memory subroutine that will subsequently retrieve an instruction. IT0 also generates PI SYNC (8.2b) which strobes the priority interrupt system provided the processor is not already in a PI cycle. If the PIR strobe produces no request or if the processor is already in a PI cycle, the sequence continues to IT1 directly. If the strobe does produce a request, the sequence continues instead to IAT0, which triggers the master clear, places the processor in a PI cycle by setting PI CYC, and then triggers IT1, which calls the memory subroutine and supplies a memory address for instruction retrieval.

In a normal sequence, i.e., PI CYC(0), the address comes from the program counter; if the processor is in a PI cycle, however, the appropriate PI channel address is transferred into MA. Furthermore, if PI OV is 1 (this can occur only in the second consecutive PI cycle when a block IOT is completed), MA is incremented by 1 at the same time that the channel address is

transferred in, so the processor performs the instruction in the second location associated with the channel. When the instruction is available in MB, the memory restart triggers IT1A, which transfers the instruction code and AC address from MB into IR. If the instruction was retrieved from a location above 17, IT1A also clears the readin mode flip-flop in case the processor has been in read in.

The instruction cycle may also be entered late for several situations. If a UUO appears in the program, UUO T2 starts a new instruction execution by setting IF1A and making a read request to retrieve the instruction in location 41; the memory restart then continues the chain automatically by triggering IT1A. After retrieval of the operand in the fetch cycle of the execute instruction, XCT T0 returns to IT1A so that the operand is executed as an instruction. Similarly, the console operation execute triggers IT1A after the contents of the DATA switches have been transferred to MB via AR.

b Address

The calculation of the effective address for an instruction is governed by the logic shown in the upper half of Figure 5-3 (flow chart, Figure 4-4). In a normal sequence, IT1A starts the address cycle by triggering AT0. However, the cycle is also triggered by CHT9 for the second part in a character operation: in this case, the processor has already handled the pointer and must now calculate the effective address for the operand as specified by the pointer. AT0 transfers the address portion of the instruction word from MB to AR, transfers the indirect bit and the index register address to IR, and clears MA in preparation for subsequent memory access.

AT0 also generates PI SYNC (8.2b) which strobes the priority interrupt system provided the processor is not already in a PI cycle. If the strobe discovers any request, PI SYNC generates IAT0 which returns the processor to the instruction cycle. If there is no request or if the processor is already in a PI cycle, the sequence continues the address cycle by triggering AT1 provided the operator has not pressed the INSTRUCTION STOP key within the preceding 100 μsec (IF1A being 0 guarantees that the return is to the address cycle rather than the instruction cycle.) If the instruction specifies no index register (IR14-17 = 0), AT1 jumps directly to AT4. However, if address modification is to be performed, AT1 triggers AT2, which transfers the index register address to MA, makes a memory read request, and sets A LONG. The return from

memory triggers AT3, which sets AF3A and requests the add subroutine to add the contents of the index register to the address specified by the instruction. The subroutine return transfers the calculated address from AR back to MB and moves the cycle on to AT4, which clears the left half of AR. If the instruction specifies an indirect address (IR13 is 1), the cycle continues to AT5 which transfers the calculated address from MB back to MA and makes another read request to retrieve the new address. AT5 also sets A LONG, clears the indirect bit and index register portions of IR, and sets AF0. The last action allows the memory restart to trigger AT0, restarting the address cycle.

If the instruction specifies a direct address, the processor goes on from AT4 to the fetch cycle. At the completion of address calculation, MA, MQ, and ARLT are all clear; the effective address calculated for the instruction is contained in the right halves of both MB and AR. The contents of MBLT are equal to the last word retrieved from memory, provided the final address cycle did not index.

c Fetch

Figure 5-4 shows the logic that controls the retrieval from memory of the operands necessary for the execution of an instruction (flow chart, Figure 4-4). The lower part of the figure shows the generation of the control levels for the cycle. In all instruction groups, individual instructions or specific instruction modes which do not require an accumulator, the net in the lower left generates an inhibit for that function. The bottom two inputs, CH INC OP and CH ∿ INC OP, both apply to the first part of character operations and indicate whether or not the pointer shall be incremented after it is retrieved. The level IR 254-7 is asserted for the four included instruction codes: the first two are jumps; the third is XCT; the fourth is not used. All other inputs that generate FAC INH are standard instruction situations (5.3). There are a few instructions which, after retrieving the accumulator, require the retrieval either of a second accumulator or of a word addressed by half of the already retrieved accumulator. The three levels that govern these situations are shown in the center of the figure. The cycle fetches AC2 for any double word shift operation and for fixed point division, which uses a double-length dividend. POP and POPJ both pull out the word addressed by ACRT, which keeps the final address for the pushdown list. Fetching of the word addressed by ACLT

is required by JRA and BLT: the former to restore AC as the return from a JSA which stored AC in E and saved E and PC in AC; the latter to retrieve a word for subsequent storage in the location addressed by ACRT.

Two levels govern the retrieval of a word according to the effective address, FC(E) and FC(E) PSE. The first reads and releases the memory so it can rewrite automatically. The second makes a read/write request: the memory subroutine thus pauses after fetching so that information can later be written into the same memory location (usually in the store cycle and controlled by the same level). For the first part of character operations, FC(E) retrieves the pointer if it is not to be incremented; whereas the pause is required if an incremented pointer is to be written back in. Similarly, in the second part, a character deposit requires the pause while a load does not. An IOT BLK also retrieves a pointer which is automatically indexed and thus requires the pause. The remaining inputs to both nets are all standard instruction situations, but the reader should take special note of two inputs to FC(E): IR MD FC(E) and IR FP. The first represents all fixed-point multiplication and division operations that do not use E as an operand; the second includes all floating instructions except floating scale. All of these instructions have modes that store the result in E; however, all require significantly greater execution time than other instructions and they request read to release the memory.

In the upper half of the figure are the delays and SBR flip-flops that provide the fetch time chain. The normal entry is from the address cycle at AT4 when no further deferring is required. If the address calculation requires no indexing or deferring at all, A LONG is 0 and the entry into FT0 is delayed; otherwise the entry is immediate. There is also an entry into FT0 for a block IOT: after the pointer has been indexed and stored, IOT T0A triggers the fetch cycle for the data instruction that follows (8.1). FT0 performs no outside operations but continues the proper fetch sequence. If the instruction requires no accumulator, the sequence goes directly to FT5. To fetch AC, the sequence goes to FT1, which transfers the AC address from IR to MA and requests a memory read. The memory restart triggers FT1A, which is also produced directly by BLT T6. In a block transfer, each word is retrieved by a fetch cycle; but the store and index operations are performed by the special BLT subroutine. The subroutine returns to FT1A bypassing FT1, because the instruction must fetch AC, which contains the initial source and destination addresses, only for the first transfer.

The sequence from FT1A and the operation performed by it depend upon whether additional AC operations are necessary. All sequences include a switch between MB and AR so that AC is in AR and E in MB. If there are no extra AC fetch operations, FT1A clears MA and switches MB and AR. To fetch a word addressed by AC, FT1A clears MA; and if the address is in the left half, swaps the halves in MB. It also triggers FT3, which loads MA with the selected half of AC, makes the MB-AR switch, and goes on to FT4. If a second accumulator is required, FT1A adds 1 to MA, switches MB and AR, and skips to FT4 which triggers the memory read for either type of extra AC fetch. FT4 also sends MB to MQ to save E. The memory subroutine returns to FT4A, which clears MA, and switches MB and MQ so that MQ contains the extra word fetched and MB again contains E. FT4A also triggers FT5, which results directly from FT1A if no additional AC operations are required. FT5 transfers E from MB to MA and then triggers FT6 or FT7 depending upon whether the instruction requires a memory read or read/write. The memory return then completes the cycle by triggering FT6A, which results directly from FT5 if the effective address is not used for memory access.

At the completion of the cycle, MA contains E; and AR and MB contain AC and C(E) if both words were fetched, although the AC halves are swapped if the cycle also fetched the word addressed by the left half. If there was no AC fetch, AC contains E with the left half clear; and if there was no C(E) fetch, MB contains E with the left either clear or in the same state as at the end of the address cycle (the latter case occurs only if no fetch operations were performed at all). MQ is clear unless an additional AC operation was required, in which case it contains the extra word.

d Execute

After fetching the operands, the processor performs the operations necessary for the execution of the instruction. Figure 5-5 shows the logic for the execute time chain, the flow chart is Figure 4-5. Only the simpler instructions are actually executed by the ET pulses; for the more complicated ones, the pulses trigger appropriate subroutines.

The lower portion of the figure shows the generation of three levels that control the execute sequence. Two are inhibit levels that break the chain for subroutines; reentry into the main sequence is made from the subroutine usually—but not always, at ET10 (XCT and UUO, for

example, both return directly to the instruction cycle). The subroutines for all IOTs and for FSB start at ET4 and thus inhibit ET5. All other instructions requiring execution by subroutine inhibit ET4 although this does not mean that the subroutine entries are made at ET3. In fact, most of them are at ET0 but the execute chain continues to ET3. There is also a group of instructions (including fixed-point add and subtract and some of the data transmission and executive instructions) whose transfer and logical operations are triggered by the ET pulses but which require an AR subroutine. These instructions all generate the level AR SBR which inhibits ET4 but only for a pause: AR SBR causes ET3 to trigger the SBR at the left at the same time that it triggers an appropriate subroutine. The subroutine return then triggers ET4 to continue the chain. AR SBR includes several instructions which generate both inhibits: floating subtract requires the negate subroutine at ET3 and enters the floating add subroutine at ET3 and enters the floating add subroutine at ET4; both block IOTs index the pointer at ET3 and switch to an IOT subroutine.

Almost all logical and transfer operations triggered by ET pulses occur in the first half of the chain; most instructions using ET6 to ET9 affect the program counter. These instructions generate E LONG (right) which causes ET5 to go to ET6 instead of skipping to ET10.

In the execute cycle flow chart, the only events for which complete conditions are listed are general operations independent of specific instructions and operations involving PC, MA, and the flags. The many logical and transfer operations on AR and MB that actually execute the Boolean, data transmission, and other instructions occur almost exclusively at pulses 0, 1, and 4; to determine which events are required for a specific instruction, refer to the appropriate instruction flow chart. The following description of the execute cycle discusses only the sequencing of the cycle, entry to and return from subroutines, and the more general operations. For events involving individual registers, refer to the appropriate section of the logic in this and following chapters.

To begin the execute cycle, the final pulse from the fetch cycle FT6A, triggers ET0 and ET0A simultaneously and also triggers a delay for a subsequent ET1; ET3 follows automatically from ET1. The two 0 pulses are logically equivalent but two pulse lines are required because of the many events they trigger. Usually ET0 increments the program counter so that the next instruction will be taken from the next memory location in sequence. The only circumstances

that inhibit PC+1 are those in which ET0 will occur again before the next instruction in normal sequence is to be performed. ET0 also clears the AR carry flip-flops for use during the cycle, synchronizes those console operations which may interrupt the normal sequence from store cycle to instruction cycle (5.1), and clears RUN for the halt instruction (JRST with a 1 in bit 10). The first three execute pulses also handle hold and dismiss operations for the priority interrupt system (8.2b), restoration of flags, and entry into user mode. ET0 provides subroutine entry for character operations, both fixed and floating multiplication and division, and floating addition; ET3 starts the subroutines for BLT, FSC, UUO, and all shift operations. XCT also stops the chain at ET3 in order to return to the instruction cycle.

ET4 follows immediately from ET3 only if there is no inhibit. For the instructions that generate AR SBR, ET3 triggers the appropriate AR subroutine and sets ET4 AR PSE. The subroutine return, ART3, clears this SBR and triggers ET4, which in turn continues to ET5 unless there is an entry to the FSB subroutine. If ET4 follows from ET3 without pause, ET5 follows immediately except for the in-out transfer instructions: for these ET4 triggers the IOT subroutine, but at its completion time pulse IOT T3 returns to the execute cycle by triggering ET5.

Most computational and data transmission instructions skip the second half of the execute cycle by having ET5 go directly to ET10. However, if E LONG is asserted, the chain continues through all the remaining pulses without interruption. ET7-9 perform the necessary program control operations (5.4); the final two pulses in the cycle regulate the flags (6.2e, 8.3). Since most instruction results that are to be deposited by the store cycle are produced in AR, most instructions switch AR and MB either at ET9 or ET10. ET10 sets up MB and triggers the store cycle not only for most instructions executed in the execute cycle but also for many of those executed by subroutine. The return for fixed-point division is at ET9 (provided the division could be performed); return is made at ET10 for character operations, all floating-point instructions (all of these except floating scale are made from the normalize return subroutine), fixed-point multiplication (which also returns via NRT6), and the block transfer routine if the block is complete.

e Store

Information resulting from the execution of an instruction is deposited in memory by the store cycle (logic, Figure 5-6; flow chart, Figure 4-5). The sequencing of the store time chain is controlled by four levels: FC(E)PSE from the fetch cycle, which gates in the write restart for any instruction that used a fetch and pause before execution; and three levels that gate in write requests and are generated by the store logic. SC(E) is generated by three types of instructions: those which merely clear location E and deposit information in it without having required a fetch; those which store C(E) in a location other than E; and those which require so much execution time that they requested only the read rather than the read and pause in the fetch cycle to free the memory during execution. SAC INH is generated by instructions that address no accumulator, by computational and data transmission instructions in the memory mode (that mode which stores the result only in memory), or by test instructions that have no result to store. Instructions that compare a memory word against zero store the word (which may be indexed) in both memory and an accumulator, but AC storage is inhibited if AC0 is addressed (the net in C6 decodes IR9-12 for no AC selection). The input BLT LAST inhibits AC storage in the cycle following completion of a block transfer. If a priority interrupt stops the block before completion, the current addresses are stored in AC so that the block may be continued after the break. Some arithmetic and shift instructions generate SAC2 to store MQ in a second accumulator. This event occurs for all double-length shifts, floating-point instructions that store the low-order halt, and fixed-point operations that have a double-length result.

To generate a write request for SC(E), ET10 triggers ST1; whereas, it triggers ST2 to provide a read/write restart for SC(E) PSE. Either time pulse sets SF3, and at the completion of the memory subroutine the return triggers ST3. If there is no storage according to E (or whatever address has replaced E during the execute cycle), ET10 triggers ST3 directly. If there is also to be storage in an accumulator, ST3 clears MA and ST5 loads the AC address into it from IR9-12, transfers AR to MB, and makes the write request. The memory restart triggers ST5A which goes on to ST6 if a second AC storage is necessary. In this case, ST6 increments MA to address the second AC, transfers MQ to MB, and calls another write. ST6 is also delayed slightly to generate ST6A which sets SF7, the SBR for the memory subroutine call. This delay is necessary to guarantee that the return from the previous subroutine cannot trigger both

ST5A and ST7. The return from this storage then triggers ST7 which also follows directly from ST3 if there is no AC storage at all, or from ST5A if there is no second AC storage. ST7 is the return time for the divide subroutine if the division could not be performed, and for a character operation that terminates after the first part (i.e., an instruction which operates not on a character but only on the pointer). Since all FP/CH codes inhibit ET4, those that are not used for instructions cause ET3 to trigger ST7; thus all unused codes (except the UUOs) are interpreted as no-ops. Furthermore, if at any time a user program attempts to address a protected area of memory, a priority interrupt is requested on the CPA channel and the sequence jumps directly to ST7. If RUN is 0, the processor stops at this point. Otherwise, ST7 returns to a new instruction cycle unless the operator has requested examine or deposit, in which case, it returns to the key cycle.

5.3 INSTRUCTION CONTROL

For each instruction to be performed, the 18-bit instruction register receives the instruction code, indirect bit, and AC and index register addresses from MB (Figure 5-7). Although information may be transferred into IR only from MB, the IOT time pulse T0A directly sets IR12, which changes a block IOT instruction into the corresponding data instruction after the pointer has been indexed. For effective address calculation, the indirect bit IR13 controls the repetition of the address cycle, and IR14-17 provides the address of an index register to be used in the calculation. Although there are 16 accumulators, only 15 of them can be used as index registers. A 0 address in IR14-17 indicates no index register selection and the address cycle performs no indexing. The generation of the appropriate address cycle gate, IR14-17=0, is shown in the lower right of Figure 5-8. For the fetch and store cycles, the AC address is supplied by IR9-12. For IOT instructions, IR0-2 are all ones, and IOT control decodes the instruction specified by IR10-12. A device is selected according to the code supplied by IR3-9; the bus drivers for the in-out selection lines are at the top of Figure 5-7.

The lower portion of the figure shows the pulse signals that control IR. MR CLR clears the entire register at the beginning of every main sequence. In every instruction cycle, IT1A transfers the instruction code and AC address (or instruction and device codes for an IOT) into IR0-12 from MB; at the beginning of every address cycle, AT0 transfers an indirect bit and index register address into IR13-17 from MB. If the calculated address is indirect, AT5 clears

IR14-17 in preparation for reloading at AT0 when the cycle repeats. When the first part of a character operation is completed, the processor returns to the address cycle to calculate the effective address for the character from the pointer. Thus CHT8A clears IR13-17 in preparation for loading at AT0, which is triggered by the final pulse in the first part of the character time chain.

The next three logic drawings, Figures 5-8 to 5-10 show the main decoding of IR0-8 to control the execution of individual instructions. In this section of the processor, the decoding is carried down in many cases to individual instructions and instruction modes. There are, however, a few command lines which represent instruction pairs whose differences are minor; trivial additional decoding for these is shown with the hardware that actually executes the instruction. For example, the IR multiply and divide outputs represent both integral and fractional operations; the floating command outputs represent both the standard instructions and those that round. Figure 5-8 shows the primary decoding into major classes, decoding into command levels for fixed-point multiply and divide and for all floating-point instructions, and final decoding for all single instructions without modes that correspond to unique 9-bit codes. All of the Figure 5-8 outputs, although representing functions at several different levels in the decoding hierarchy, have the prefix IR. Outputs from the groups decoded in Figures 5-9 and 5-10 have the appropriate group prefix. Figure 5-10 also shows the timing for SCT and UUO.

The binary-to-octal decoder in the upper left of Figure 5-8 performs the first stage in instruction decoding by determining which of the eight primary instruction classes is specified by IR0-2. The decoder gating input P is grounded so that the decoder is always on. All codes in the UUO class produce exactly the same operations and require no further decoding. In the IOT class, the eight instructions specified by IR10-12 are decoded in IOT control. Outputs 1 and 2, IR FP/CH and IR 2XX, gate other decoders shown in this figure; further control in the classes represented by outputs 3, 4, 5, and 6 is shown in Figures 5-9 and 5-10. The primary command level IR FP/CH includes the instructions for floating-point arithmetic and character operations, and instruction codes 100 to 131 which are not used. IR FP/CH ANDed with the condition 011 in IR3-5 (i.e., codes of the form 13X) gates the lower left decoder, which produces the command levels for FSC and the five character operations. At the right, the

primary level gated by IR3(1) represents all floating-point instructions other than FSC: bits 4 and 5 are further decoded to determine the instruction, bits 7 and 8 to determine the mode. Bit 6 does not appear here but controls rounding at the normalize return subroutine.

The top center decoder is gated by IR 2XX to convert IR3-5 into eight command levels, each representing eight instructions or two instructions with four modes. The 0 and 1 outputs each represent two of the move instructions and are ORed to generate the command level for the group. At the left, IR SH is further decoded into the six shift instructions plus two unused codes (the two types of arithmetic shift are ORed in B4 for use by the overflow logic); at the upper right IR 25X is decoded into seven miscellaneous instructions plus one unused code. Just below, a gate generates the signal IR 254-7 to inhibit AC fetch and storage, which is not required for these codes.

Although the decoder gated by IR 2XX generates two outputs for fixed-point multiply and divide, the condition representing both of these outputs also generates a combined command level IR MD (Figure 5-8, center). Further decoding produces the appropriate fetch and store gates. All modes but the immediate (01 in bits 7 and 8) fetch C(E). Only fractional division uses a double-length dividend and fetches AC2. The two modes with a 1 in bit 7 store the result in memory; whereas only the memory mode inhibits storage in AC. For all modes that do store in AC, a second AC storage is required for the remainder in division and for the low-order half of the double-length product in fractional multiplication.

Note that three of the decoder outputs at the top of the figure have a suffix "A:" these are IR UUO A and IR IOT A in the upper left, IR JRST A in the upper right. These decoder outputs do not drive the command lines for the corresponding instructions. Instead, they are applied to the executive logic (5.5) to determine in the latter two cases whether the instruction is allowed or must be executed as a UUO. The command level for UUO is generated by the executive logic; the command lines for the other two instructions are driven by the gates in C4, each of which is enabled by the appropriate decoder output when the instruction is not being executed as a UUO. The gate just below ORs the 1 states of IR9 and IR10 for use by the executive logic in testing a JRST.

ACCP V MEMAC, ACBM (Figure 5-9)

For the ACBM group (upper right), bits 3 and 4 are decoded for the action on the masked bits: do nothing clear, complement, set. IR5(1) selects C(E) for the mask direct, as against immediate which uses E. A 1 in bit 8 swaps the AC halves before masking. The net in the upper left decodes bits 3 and 4 for instructions in the class that includes both ACCP and MEMAC: one fourth of the instructions do an arithmetic comparison of AC against either E or C(E); the other three types compare C(E) or AC against 0 and either merely test, or add or subtract 1 before testing. Any instruction in the latter three types generates the main control level MEMAC; the state of bit 5 determines whether the test word shall come from memory or an accumulator. In ACCP, a 1 in bit 5 specifies a comparison with memory (direct); otherwise the test is made against E (immediate).

The logic nets at top center in the figure test the skip or jump condition for all three groups, ACCP, MEMAC, and ACBM. The condition is determined by the pair of gates at the left: bit 7 selects the condition that AR is 0; bit 8 that the test word is less than the standard, be it 0, C(E), or E. The upper gate functions only for ACCP and MEMAC: the logical comparison employs an AND function and the only test is whether or not the masked bits are all zeros. In an arithmetic comparison, the function AR = 0 represents equality of the test word and the standard. The function representing the inequality is the exclusive OR of the AR sign bit and the overflow condition. This function is true when the subtrahend is greater than the minuend; but since the standard is subtracted from the test word, it is true when the test word is smaller. In MEMAC, the overflow condition is automatically false (for a complete description, see the flag logic, 6.2e) so the entire function is true when AR0 is 1, i.e., when the test word is less than 0.

FWT (Figure 5-9)

For a full-word transfer, the four standard modes are decoded from bits 7 and 8; a left-right swap is made for MOVS (bits 5, 6 = 01); and the word is negated either when moving the negative or when moving the magnitude while the word is negative, both situations being represented by the condition IR6(0) V AR0(1) in MOVN or MOVM.

HWT (Figure 5-9)

For a half-word transfer, the standard modes are decoded from bits 7 and 8; a 0 or 1 in bit 3 specifies whether the half word shall be transferred into the left or right half of the destination; and a 1 in bit 6 specifies that the source word shall be swapped before the transfer is made so that the left half of the source is transferred into the right half of the destination or the right into left. The other levels control the operation on the other half of the destination. If bits 4 and 5 are both 0, there is no action. If either bit is 1, AR is cleared; the word is constructed by loading a half word into one half and complementing the other half if it should be all ones. If bit 4 is 0, the other half is left clear; however, on IR4(1) the other half is set to all ones (i.e., complemented) if the instruction requires that it be set or that its bits be made equal to the sign of the transferred half and that half is negative. The gates to the left generate the appropriate functions. For a transfer to the right, the upper net generates HWT LT SET if instruction bit 5 is 0 or the sign bit MB18 of the right half word is 1. The lower gate performs an equivalent function for the transfer left.

BOOLE, AS (Figure 5-10)

Since BOOLE and AS have the same modes, the OR function of the two instruction levels gates a decoder for bits 7 and 8. For further decoding of AS, the command level is merely gated by the states of bit 6 to determine whether addition or subtraction is required. To decode for the 16 Boolean operations, bits 4, 5, and 6 are applied to binary-to-octal decoders, one gated by the 0 state of bit 3, the other by the 1 state. The table at the left lists the 16 operations by name, function, and number, shows the result for each of the four possible pairs of operand bits, and lists the basic functions which, taken together, produce the required result for a specific instruction. Each operation is performed by three of the execute time pulses and may require from one to three of the basic functions.

JP (Figure 5-10)

Bits 6-8 of the jump and pushdown group are decoded into eight individual instructions. For controlling transfers between MB and AR at the end of the execute cycle, there are two subsidiary levels—one that represents all JP instructions except JSP, and another that represents the pushdown and pullout instructions. For PC control, JP JMP represents the six instructions

that jump, i.e., all except PUSH and POP. Three of the instructions that save PC (PUSHJ, JSR, JSP) also generate JP FLAG STOR to save the miscellaneous bits (JSA also saves PC but it stores E instead of the miscellaneous bits).

XCT, UUO (Figure 5-10)

IR XCT causes ET3 to trigger XCT T0, which returns the processor to the instruction cycle at a point beyond the memory subroutine so that the processor then executes the operand in MB as an instruction, just as though it had been retrieved by the instruction cycle. For all codes in the UUO class, the execute clears MA and MBLT, then transfers address 40 into MA and the instruction code into MBLT. The last event is equivalent to UUO T0, which triggers a memory write to deposit the trapped instruction, with its address portion replaced by the calculated effective address in location 40. The memory return triggers UUO T1 to index MA, and UUO T2 makes a read request and sets IF1A in the instruction cycle logic. The memory return automatically triggers the remainder of the instruction cycle so the processor performs the instruction in location 41.

5.4 PROGRAM CONTROL

Each instruction in the program is retrieved from the memory location addressed by the 18-bit program counter (Figure 5-11). At the beginning of every execute cycle, the counter is stepped one position so that instructions are taken from consecutive memory locations. The program controls its own sequence by means of skip and jump instructions. Skip instructions cause the processor to skip one instruction in the normal sequence if a specified condition is satisfied; the skip is implemented by advancing PC one extra position at the end of an execute cycle. Jump instructions transfer program control to any chosen location, sometimes upon satisfaction of a condition, by loading a new address into PC. An address can be transferred in only from MA so any input from the console ADDRESS switches or a jump address from MB must be made via MA and the transfer must be preceded by a clear. The flip-flops are connected in a carry configuration so a pulse at the PC+1 input to PC35 adds 1 to the contents of the counter.

Figure 5-12 shows the control logic for the counter: three control pulses are produced in the upper half; the lower half shows the generation of the gates that control counting, skipping,

and jumping. Every program begins with the console operation start or read in, in which KT1 clears PC and KT3 loads the ADDRESS switches into it. For normal counting, PC is incremented at the beginning of every execute cycle. The circumstances which inhibit program counting at ET0 are those in which another execute cycle will occur before the next instruction in normal sequence is to be performed. Character operations generally require two main parts, the second beginning with the address cycle, so PC is not incremented during operations on the pointer except in the single instruction that affects only the pointer and has no second part (C3). XCT, UUO, and a block IOT BLT all involve execution of instruction pairs and counting occurs in the second execute cycle. Since a PI cycle interrupts the normal sequence between instructions, the count must be inhibited in it because the processor has not yet executed the currently addressed instruction. Counting is inhibited throughout a block transfer because the BLT subroutine returns to the fetch cycle to process each word. When the block is complete (MQ0 = 0), PC is counted directly from the subroutine at BLT T5A. The inhibit also applies to an instruction executed from the console.

For changes in the program location out of normal sequence, PC SET causes ET7 to clear and ET8 to load, whereas PC+1 (ET9) causes ET9 to count. The net for PC SET has as input all unconditional jump instructions and an enable level which is asserted for any conditional jump when the condition is satisfied. Conditions include JFCL when the addressed flag is 1, the add-1 jumps when AR has the appropriate sign, and those arithmeitc compare instructions that use an accumulator when the test is satisfied. An extra count occurs at ET9 on two unconditional subroutine jumps so that the subroutine begins one place beyond the storage location of PC or AC. Any other extra count is for skipping and is represented by PC+1 ENABLE. The skip conditions include the two IOT status test instructions when the appropriate condition appears in AR, and the satisfaction of the test condition for any logical compare instruction or those arithmetic instructions that compare memory against 0 or an accumulator against either E or C(E). There is another conditional skip for a block IOT that does not use the PI system: if the indexing of the pointer did not produce a carry into AR0, IOT T0A counts PC (the computer performs the next instruction in normal sequence only if the block is complete). If any jump or skip occurs (other than an IOT block skip), the OR gate at the right generates the level PC SET $\vee$ PC+1 for setting the PC change flag (6.2e).

5.5 EXECUTIVE LOGIC

The executive logic allows the executive routine to control the sharing of processor and memory by a number of programs. The executive routine selects a user program and the area in core assigned to it by loading the protection and relocation registers, placing the processor in the user mode, and jumping to a location appropriate to the selected program. The user program may be interrupted temporarily by a block IOT in a PI cycle (which is under control of the executive routine and is hence unrestricted); but if a JSR is performed in a PI cycle (such as following the completion of a block IOT or for servicing some other type of interrupt), the processor leaves the user mode with control returning to the executive routine. Other than for priority interruptions, the user program has control until it attempts to use a protected area of memory or to perform an illegal instruction. The former action causes the processor to go the end of the current main sequence and triggers a priority interrupt on the processor channel; the latter causes the processor to perform the illegal instruction as a UUO (all UUOs are unrestricted but automatically return control to the executive routine).

Figure 5-13 shows the executive logic except for the memory protection and relocation registers which are discussed with the memory address logic (7.1c). The pulse amplifiers at top center provide the clear and set pulses for PR and RLR when the DATAO clear and set pulses are gated by CPA, i.e., when a DATAO for the processor appears in the program. The only unprogrammed clear for these registers is the master start because their states must remain until deliberately changed by the program. Similarly, the user flag and the illegal operation flip-flop (A3, B5) are also cleared only by the master start because they must maintain control functions from one main sequence to another. The remaining three flip-flops in Figure 5-13 provide synchronization and are cleared by the master clear. However, since the set function for EX PI SYNC is a level output from the PI cycle flip-flop, it remains set as long as PI CYC is 1 even though there may be a master clear between a pair of PI instructions. After PI CYC is cleared, the sync remains set until the beginning of the next main sequence. EX UUO SYNC is set at the beginning of every address cycle and then cleared at the end of the main sequence. The purpose of this flip-flop is merely to prevent the generation of the UUO command level during the time that IR is clear in the instruction cycle.

To set up a specific user program, the executive routine loads PR and RLR with a processor DATAO. It then jumps to the location for the program with a JRST that also sets the executive

mode sync flip-flop (B3). The setting may be done either by programming a 1 in bit 12 of the JRST, or by restoring the flags (a 1 in bit 11) provided that a user program was running at the time the flags were stored. Instructions that store the miscellaneous bits with PC store EX USER in bit 5, but the restoring JRST does not act on the user flip-flop directly. Instead, it sets the executive mode sync if MB5 is 1. The processor does not leave the executive mode until the end of the main sequence in which the sync is set: at this time, the transition of the sync back to 0 sets EX USER.

The addresses in the user program are checked against PR to determine whether they are legal and relocated to the area assigned to the program. The net in the lower right generates a level that inhibits both relocation and protection; the inhibit is always asserted if the user flag is 0. The net also inhibits the relocation of fast memory addresses so that these locations are available to all programs (the protection inhibit is really unnecessary here because these addresses are ipso facto less than the minimum block size). There is also no protection or relocation of addresses that occur in a PI cycle because the instructions executed are under control of the executive routine and must be unrestricted even if a user program is running. Again only the relocation inhibit is necessary for the PI channel addresses, but both inhibits are necessary for the addresses given by the instructions. If the interrupt should require a jump to the executive routine, the JSR that calls the routine also stores and clears the user flag (B2).

The nets in the upper right monitor user instructions by receiving the IR decoder outputs for JRST, IOT, and UUO to generate the UUO command level. The JRST or IOT decoder output drives the corresponding command line only if the instruction is not executed as a UUO. When the system is in user mode, EX IR UUO replaces a JRST that attempts to dismiss a priority interrupt or halt the processor and replaces user IOT (a block IOT in a PI cycle is not part of the user program). Any UUO regardless of mode enables EX IR UUO (after the UUO sync is set); this thus provides a means by which a user program can communicate with and return control to the executive routine. EX IR UUO inhibits relocation since it must use addresses 40 and 41, and at ET1 it sets EX ILL OP. The 1 state of this flip-flop then continues the inhibit into the JSR in location 41. At ET7 the JSR clears the user flag and at ET8 the illegal operation flip-flop. Just in case a block IOT that does not overflow should interrupt the JSR, EX ILL OP is cleared by all block IOTs.

The illegal operation flip-flop is also used to inhibit relocation during an examine or deposit that is inserted between two main sequences while the processor is running. For this purpose EX ILL OP is set by ST7 at the same time that the key cycle is triggered and cleared by the read/write return.

CHAPTER 6

ARITHMETIC LOGIC

The first half of this chapter describes the registers used for arithmetic calculations. Three full-word registers, AR, MQ, and MB, are used for computations on full-word and half-word fixed-point numbers and the fractional parts of floating-point numbers. All data transfers and logical operations on computer words are also performed in these registers. Besides the full-word registers, there are two 9-bit registers SC and FE that are used for floating-exponent calculations and for various subsidiary computations such as calculating the size and position portions of the pointer in character operations and counting the number of steps required for fixed- and floating-arithmetic operations. Included with each register is a discussion of its input gating, its control logic, and any other hardware associated specifically with the register (e.g., the flag logic and AR subroutines with AR, the SC subroutines with SC).

The second half of this chapter describes the time chains that control the execution of data and arithmetic instructions outside of the execute cycle. The test discusses the generating conditions for the pulses in a given chain and also describes the events that occur at each step. In many instances, a number of pulses from the different time chains all must trigger the same operation in an arithmetic register. Many subroutine time pulses are therefore connected to the register control logic through OR gates in a subroutine interface (6.5). Lines from this interface are labeled by the functions they perform and all have the prefix CFAC (computer floating-arithmetic connection). When only one or two subroutine time pulses trigger a given event, they are supplied directly to the register logic from the subroutines. In the following discussion of the arithmetic registers, the significance of the generating conditions for an individual register operation is given only for the execution of those instructions performed in the execute cycle and for other events in the main sequence. To determine the significance of any event triggered by a subroutine time pulse, either directly or through the subroutine interface, the reader should refer to the discussion of the appropriate subroutine.

The three full-word arithmetic registers are contained in the same set of double-height modules, that is, MB_i, AR_i, and MQ_i are all on the same module (it also contains the ith bit of the

memory indicator register MI, which is described in the memory logic). Each 36-bit register is controlled as a pair of half registers and is shown in two logic drawings, the first for the left half in mounting panels D and E of bay 2, the second for the right half in panels H and J of the same bay (e.g., see MB, Figures 6-1 and 6-2). The flip-flops of all half registers are in panel locations 5 to 12 and 16 to 24. All control pulses are supplied to them by a pair of pulse amplifiers in locations 14 and 15. On the register drawings, these are shown merely as blocks with the pin connections labeled; the actual pulse amplifiers with input gates are shown in the drawings of the control logic associated with the registers. Each PA output drives 18 input gates and is connected to 18 register modules for ordinary 0 or 1 transfers, but to only 9 for jam transfers. All inputs to every register bit are labeled by signal name; but since all the registers are contained in the same set of modules, connections between corresponding bits are made internally and pin connections are listed only for external signals.

6.1 MEMORY BUFFER

Figures 6-1 and 6-2 show the left and right halves of the full-word memory buffer. All transfers to and from memory are made via MB, but the register is discussed here because it holds one of the operands in most arithmetic and logical instructions. Each MB flip-flop has a direct clear input (which receives the register clear pulse) and gatable clear and set inputs. The MB modules include six sets of internal gates, but the gatable inputs are also available at the connector so that transfers can be made with single-bit pulses from external gates. Transfers of either zeros or ones may be made into MB from either AR or MQ. The two halves of MB may also be swapped; i.e., the left may be jam transferred into the right and the right into the left. For various executive instructions, the program counter may be stored in the right half of MB; the instruction register in MB LT. Transfers of information into MB from the memory bus (7.2) are made via single-bit pulse set inputs shown at the bottom of the figures. Clear inputs are also available, but these are used only for bits 1-8.

The external gates for the single-bit inputs other than those from the memory bus are shown at the top in Figure 6-4. Besides the clear gates for bits 1-8, there are also set gates for bits 0-8 that parallel the memory bus inputs. The gates are controlled by three transfer pulses, two of which set or clear bits 1-8; whereas the third stores the miscellaneous bits in MB0-5.

The generating logic for these transfer pulses is shown in the upper right of Figure 6-3. The set and clear functions occur in floating-point operations to nullify the exponent part of the register and are conditioned by the MB sign bit (refer to the appropriate subroutine). Storage of the miscellaneous bits occurs at ET6 in several of the subroutine-calling jumps. These bits include the four AR flags which can be used by a subroutine but return to their original states following the interruption. The miscellaneous bits also include CHF7 and EX USER. The latter is saved so that the executive routine can return control to a user program by restoring the flags. Saving CHF7, which is set at the end of the first part in a character operation, allows the computer to return properly to the second part if there is a priority interrupt between the two.

The remainder of Figure 6-3 shows the logic that governs all of the regular MB transfers. The top section shows the pulse amplifiers that drive the register gates; these PAs are triggered through OR nets whose inputs come either from the subroutine interface (6.5) or from the transfer gates in the center section of the figure. Timing inputs to these gates are supplied by the various main sequence and subroutine chains; the gating levels for the execute time pulses are supplied by the OR nets shown in the bottom section.

The clear function (upper left) is not required for transfers to MB from AR or MQ since these are always jam transfers. For transfers in from memory, a signal from memory control clears MB before the single-bit pulse inputs arrive via the bus. For a UUO, ET1 clears only the left half of MB leaving the effective address in the right half, and the ET3 loads IR into MBLT (D2, B7). The transfer pulse also triggers the UUO subroutine (5.3). MB must also be cleared before the transfers from PC that occur in various jumps. These jump instructions generate the level MB PC STO (C3) which clears MB at ET5 (B1) and transfers PC into it at ET6 (A6, B6). All but the restoring jump JRST also save the miscellaneous bits along with PC (A8). There is one jump, JSA, that transfers PC at ET6 but is not included in MB PC STO; this instruction gates ET6 via the diode net in D6. No prior clear is required because the left and right halves of MB are swapped at ET0 (D7) placing E in the left half and leaving the right half clear.

The two diode nets in D7 provide six conditions for the left-right swap of MB. Three of these conditions are standard instruction modes and a fourth, JSA, is described above. For the

other two, CONO makes the swap so that E is available over both halves of the I/O bus, and BLT does it to restore the address pair to its original configuration in order to store the data word in the location specified by the right half. These two levels gate the appropriate execute time pulses at B2 to trigger the transfer PAs in A2, 3. The swap is also triggered at FT1A (B2) whenever the processor must fetch a word addressed by ACLT (5.2c). The fetch cycle swap includes BLT, which then requires the second swap to restore the original address pair at ET1.

The logic that controls transfers from AR to MB occupies the entire center portion of the drawing from top to bottom. Almost all the transfers are of zeros and ones simultaneously, but there are several cases in which zeros are transferred alone (B3). In the ACBM group, a 0 transfer is always made at ET1 (the group command level is inverted through the diode net in D5) to AND the data word with the mask; then at ET6 zeros are transferred again if the masked bits are to be cleared. The 0 transfer is also used in a character deposit to produce an actual transfer: the character portion of MB contains all ones and all bits outside of the character in AR are also ones. The other three control pulses trigger the transfer of zeros and ones together: they jam transfer AR to MB, but two of the signals (one from the subroutine interface) are also applied to AR control (6.2c) to trigger the transfer from MB to AR at the same time. The double transfer always occurs in the fetch cycle to move AC to AR and E to MB; FT1A does it (D3) if there is no fetch of an additional word addressed by either half of AC, otherwise FT3 makes the transfer (B3). The MB-AR switch is required at three points in the block transfer subroutine (6.6a); the remaining transfers are at execute times gated by levels generated in Figure 6-3 (ET0, C2; ET4, D8; ET9 and ET10, C6). In a few instances, the transfer is actually necessary in only one direction but additional hardware would be required to eliminate the superfluous one. For an explanation of these cases as well as any other events that are not immediately obvious in the flow charts, refer to 4.5 (also see below).

The single transfer from AR to MB (B5) follows index register modification in the address cycle at AT3A to move the calculated address back to MB; ST5 triggers the transfer in order to store the result of an instruction in an accumulator. Any deposit or instruction execution from the console uses the transfer because the contents of the DATA switches can be sent to MB only via AR; AR also goes to MB at the beginning of the multiply and divide subroutines (6.8b, c). In the execute cycle, the transfer may be made at either ET0 or ET10: the former involves only standard instruction situations (D5), but the latter is complex and requires some comment.

In most instructions, the result appears in AR; and if it is to be stored in an accumulator, the transfer from AR to MB occurs at ST5 (the MB-AR switch at ET9 in ACBM is made specifically to move the result from MB to AR because the transfer in the opposite direction occurs automatically for AC storage). However, if the result is to be stored in E, either by requesting a write or restarting a read/write, the result is transferred from AR to MB at ET10 in preparation for the store cycle unless a transfer inhibit is asserted. The net that generates the transfer gating level is in C7; the net for the inhibit is in D4. The conditions generating the inhibit represent situations in which the result is already in MB or is being moved there by a 2-way transfer. In EXCH, the switch is made at ET0 (C2); in a character deposit, the character is inserted into the data word in MB. The transfer is also inhibited for all instructions in the jump and pushdown group: instead these instructions use the 2-way transfers at ET9 and ET10 (C6). At ET9, the switch is made for all instructions except JSR, whose result is already in MB (flowchart, Figure 4-7 left). In JSP, JSA, and JRA, the ET9 transfer results in the appropriate configuration for the store cycle. In the other four instructions, represented by the logic level JP ∧ IR6(0), the result is already correctly placed before ET9, so a second switch of MB and AR is made at ET10. This double switch is made so that the address from AC right is available to MA at ET10 in PUSH and PUSHJ (the transfers are unnecessary in POP and POPJ, but extra hardware would be required to eliminate them).

The remaining transfer is that of both ones and zeros from MQ (Figure 6-3, upper right) which is triggered by two signals from the subroutine interface, one for a single transfer, the other for a switch of MB and MQ. The conditions within MB control that trigger the transfer (B5) include FT4A to return E to MB following a second accumulator fetch operation, and ST6 prior to storing a second accumulator. The transfer also occurs at the beginning of the block transfer subroutine (6.6a). In the execute cycle, it is used at ET0 in three of the JP instructions (C1) to move to MB the word fetched from a location addressed by half of AC.

6.2 ARITHMETIC REGISTER

Figures 6-5 and 6-6 show the left and right halves of the full-word arithmetic register. Each AR flip-flop has a direct clear input (which receives the register clear pulse), gatable clear and set inputs, and two complement inputs, one of which accepts a positive-going pulse, the other a negative-going pulse. The AR modules include ten sets of internal gates, but the

gatable 0 and 1 inputs are also available at the connector so that transfers can be made with single-bit pulses from external gates.

The AR outputs are connected within the 6205 modules to the input gating of the memory buffer and are also available through the module connectors for connection to the shift gates in other AR flip-flops, the in-out bus, and various other places in the processor. The outputs of the sign bit AR0 are used throughout the processor for control purposes, such as in the program control test nets and in many of the arithmetic subroutines, so these outputs are buffered by the drivers shown in the lower left of Figure 6-5. The 0 outputs of all AR bits are also available at the connectors through diodes that are joined externally to form large AND gates as shown in the left of Figure 6-10. Assertion of the output from a single AND gate indicates that the corresponding portion of the register contains all zeros (note that bit 9 is not included among the four gates). The first stage decoder outputs are further ANDed in the two nets in C3: the lower net decodes AR for the condition that every bit is 0, the upper net for the condition that bit 9 is 1 and bits 10 to 35 are all zeros. The latter condition, represented by assertion of the signal AR = FP HALF, is necessary for normalizing in floating-point operations. The floating-point fraction -1/2 (AR bits 0 and 9 both 1, and zeros in bits 10 to 35) is considered normalized even though bit 9 is the same as the sign bit.

In addition to the arithmetic register gating and control, this section describes the addition algorithm, the AR subroutines, and the flag logic.

a AR Gating

The external gates for single-bit pulse inputs are shown at the bottom of the two AR drawings, Figures 6-5 and 6-6. The only inputs presently used are for bits 0-8 and the gates for these are shown in the lower part of Figure 6-4. The upper pair of transfer pulses merely clear or set bits 1-8; the lower pair (the two gates at the lower left are triggered by the same transfer pulse as the row above) provide jam transfers from SC1-8 to AR1-8 and from SC3-8 to AR0-5.

The bottom two sets of gates in the register drawings, Figures 6-5 and 6-6, provide 1 transfers into AR from the I/O bus and the console DATA switches. Both of these transfers are preceded by the clear pulse for the register. The next two rows are jam transfer gates for right and left shifting; gating levels for a given bit are the 0 and 1 outputs of the adjacent bits. Since

at the register extremities the connections vary depending upon the type of shift, there are special inputs for the left shift gates at AR35 and for both sets at AR0. The generating nets for these special shift inputs are shown in the upper portion of Figure 6-7. At the left are listed the different types of shifts with the time pulses at which they occur and block diagrams showing the shift configurations. These diagrams are also drawn at the appropriate places in the flowcharts. Among the level inputs to the shift nets are several composite functions that represent groups of conditions, all of which require the same shift type. The arithmetic shift of AR and MQ combined (B4) is required by the corresponding shift operation, but it is also required in the normalize return and floating-add subroutines. Another composite function is SHC DIV, which is asserted by any type of division provided NRF2 is 0 (D4). The flip-flop condition does not apply to fixed division but is necessary in control over floating division so that the control level cannot affect shifting in the normalize return subroutine, which follows all floating-point arithmetic subroutines. Just above is another composite function (C4), which is asserted during a division or a combined logical shift.

The three shift input nets use these composite functions as well as individual instruction levels to determine the effect of any given shift on the AR extremities. In left shifting, AR0 receives the state of AR1 unless some type of arithmetic shift is being performed, in which case AR0 is unaffected. The center net controls AR0 whenever AR is shifted right. The upper two gates make MQ35 the source of data for AR0 on a combined rotation and AR35 the source on a single rotation. The gates at the lower right of the net disable the 1 input so that AR0 is automatically cleared in a character load or a logical shift; the gate at the lower left disables both inputs so that AR0 is unaffected by any right shift in multiplication or division, or a right arithmetic shift. The net at the right provides shift left input to AR35 from AR0 in a single register rotation, from MQ1 in any double length arithmetic shift, and from MQ0 in a combined logical shift, combinedrotation, or division shift. The gate at the lower right disables the 1 input so that AR35 is automatically cleared in a single left arithmetic or logical shift, or a character deposit.

Above the shift gates are three rows of gates that use the outputs of the corresponding bits in MB. The bottom two rows supply 1 and 0 transfers that provide the OR and AND functions of MB and AR when used separately, but provide a jam transfer when pulsed together. The third set is connected to the AR complement inputs and is conditioned by ones in MB. Pulsing

this set of gates produces in AR the exclusive OR of MB and AR. The next row of gates above the MB gates provides a simple complement function, i.e., pulsing these gates complements all AR bits. The top two sets of gates generate carry pulses, both of which are applied to the flip-flop to the left of the module containing the gate that generates the pulse. For example, the gates on the AR8 module use AR8 and MB8 outputs as level inputs, but the carry output complements AR7 (this method of placing the gates saves pin connections because the level inputs are internal to the module). The two sets of gates provide related carry functions, a ripple carry and a carry initiate. The lower set, the ripple carry, is a serial function, i.e., there is no control pulse applied to all gates simultaneously. The chain starts at AR35 with the pulse AR+-1T1, which is applied both the the AR35 complement input and the lower carry gate. This pulse, which occurs only when adding 1 to the contents of the register, complements AR35; and if AR35 is 1, it also triggers the ripple carry to AR34. This second carry in turn complements AR34; and if that bit is a 1, carries on to AR33. The chain continues through the register in this manner except for a break between AR18 and AR17: the carry out of AR18 automatically enters AR17, but the latter may be pulsed independently in order to index two 18-bit words simultaneously.

The upper set of carry gates provides the full-register carry-initiating function for addition. This arithmetic operation is carried out in two stages, first a partial addition, then a carry function. The partial addition is the exclusive OR function of MB and AR (AR $\leftarrow$ MB($\forall$)). After the partial sum (the result of the partial addition) has been formed, the full-register carry-initiating pulse triggers the upper set of carry gates to change the exclusive OR into the arithmetic sum. At the end of the operation, the number in AR represents the sum of the contents of MB and the previous contents of AR.

For any given bit, the partial sum (the exclusive OR function) of two numbers is actually the correct sum if there is no carry into that bit. But if there is a carry for that bit, the partial sum is the opposite of the arithmetic sum. For each bit where both summands are 1, the carry initiate directly complements the next more significant bit. However, since the processor cannot sense the prior state of a flip-flop, it instead senses the corresponding configuration of the partial sum. If after partial addition, AR_i is 0 and MB_i is 1, both bits must originally have been 1 and the carry therefore complements AR_{i-1}. Anytime a bit is complemented, a ripple carry is initiated into the next more significant bit, but this carry is inhibited if the

bit was complemented from 0 to 1. At each stage, a carry produced through the upper gate goes not only to the complement input of the next more significant bit but also to its lower gate. Thus, a carry initiated by the partial addition of two ones ripples up the register until it terminates when a 0 bit is complemented. That this algorithm does produce the correct sum of two binary numbers is proved below.

b Addition Algorithm

Let A be the original contents of AR, B the contents of MB, PS the partial sum produced in AR by the partial addition, and S the arithmetic sum of A and B. For convenience, let A and B be positive binary fractions whose sum is less than 1, i.e., there is no overflow. A bit of the partial sum PS_i is equal to a bit of the sum S_i if there is no carry into S_i. If there is a carry, PS_i is the complement of S_i. Since there can be no carry into the least significant bit, $PS_{35} = S_{35}$.

To understand the operation of the two carry functions, divide PS into sections from the right so that the first section starts with PS_{35} and ends at the first bit PS_i that satisfies the conditions $PS_i = 0$, $A_i = B_i = 1$. The second section starts with PS_{i-1} and extends to the next bit that satisfies the same conditions as PS_i. Proceed in this way through the entire partial sum. Since there can be no carry input to the least significant bit of the partial sum, it must be correct. If this bit is 1 or if it is 0 resulting from the partial addition of two zeros, there is no carry out and the next bit of the partial sum is also correct. Proceed with each more significant bit of the partial sum until reaching the bit PS_i, which is 0 resulting from the partial addition of two ones. This bit is also correct; therefore, all bits in the first section are correct.

Because the partial sum in PS_i generates a carry, PS_{i-1} is not correct and a 1 from the first section is carried into it by the carry initiate. If PS_{i-1} is 1 (resulting from the partial addition of 0 and 1), there is a ripple carry into PS_{i-2}. The ripple carry propagates up the register until a 0 bit is encountered. If this 0 is the result of a partial addition of two zeros, no further carry is generated; all further bits are correct up to the next 0 that results from the partial addition of two ones, i.e., up to the end of the section. If the 0 that terminates the ripple carry results from the partial addition of two ones, there must be a carry into the next bit. However, the partial addition of two ones is the condition that ends the section and the

carry initiate begins a new ripple carry in the next section. Consequently, the carry complements all incorrect bits of the partial sum. At the completion of the carry operation, the result S is the correct sum of A and B.

The preceding example shows that the addition algorithm works for the special case of two positive numbers. Before proving the algorithm for the remaining cases, including negative operands, some further facts should be understood:

The sign bits are included in the partial addition, i.e., the partial sum of two minus signs (ones) is a plus sign (0).

Both carry functions are applied to the sign bit AR0, which is treated as though it were a next more significant bit of AR.

The sign bit conditions both carry functions: there is a carry out of AR0 if two negative numbers are added, or if there is a carry out of AR1 and the sign of the partial sum is minus. Carries into and out of the sign bit (i.e., carries out of AR1 and AR0) are used to detect overflow.

Assume that the binary point is to the left of the most significant bit, i.e., all positive numbers are 35-bit, fixed-point fractions. The computer representation of the positive number x is therefore +.[x] where the brackets enclose the number contained in AR1-35. The sign of this number is represented by the state of AR0. In 2's complement arithmetic, the negative of a number is produced by changing the sign and subtracting the magnitude from 1. The computer representation of the number -x is therefore -.[1 - x]. With this representation, there is no negative 0; the magnitude 1 - 0 overflows, changing the sign back to plus. Furthermore, the largest negative number is -1, represented by the configuration -.[0].

The four cases of addition of two positive 35-bit fractions are:

$$x + y$$
$$(-x) + (-y)$$
$$x + (-y), \quad y \leq x$$
$$x + (-y), \quad y > x$$

Since the 2's complement format allows a representation for -1, either x or y may be 1 in the second case and y may be 1 in the fourth case. In the first case, which is discussed above, the contents of AR after addition represent the number $+.[x + y]$. If $(x + y) \geq 1$, the carry out of AR1 changes the sign. Consequently, if the addition of two positive numbers results in a negative answer, it is apparent that the sum has exceeded the capacity of the register. The processor detects the overflow by checking the sign bit carries: there is a carry from AR1 but none from AR0. The contents of AR then represent the number:

$$-.[x + y - 1]$$

In case two, the addition of two negative numbers, the partial addition and all carries except that into the sign bit would be:

$$\begin{array}{l} -.[1 - x] \\ \underline{-.[1 - y]} \\ +.[1 + 1 - x - y] \end{array}$$

If $(x + y) \leq 1$, the AR1 carry changes the sign and the complete result is:

$$-.[1 - x - y]$$

which is the computer representation of $-(x + y)$. If $)x + y) > 1$, there is no carry into the sign bit, and its absence in the presence of a carry from the sign bit indicates overflow. AR then contains:

$$+.[1 - (x + y - 1)]$$

In case three, the addition of x and -y, where y is less than or equal to x, the partial addition and all carries except that into the sign bit would be:

$$\begin{array}{l} +.[x] \\ \underline{-.[1 - y]} \\ -.[1 + x - y] \end{array}$$

Since $y \leq x$, it follows that $(1 + x - y) \geq 1$. Hence the AR1 carry changes the sign and the complete result is $+.[x - y]$. Since the signs of the operands are different, the magnitude of the result cannot exceed the larger operand and there can be no overflow. Although there is an AR1 carry, the minus sign resulting from the partial addition allows it to ripple through producing an AR0 carry.

In case four, the addition of x and -y with y greater than x, the partial addition and the carry function are:

$$\begin{array}{l} +.[x] \\ \underline{-.[1-y]} \\ -.[1+x-y] \end{array}$$

Because $y > x$, it follows that $(1 + x - y) < 1$. Hence there are no carries from AR1 or AR0 and no overflow. The above result is the 2's complement representation of the number $x - y$, i.e., $-(y - x)$.

Addition is also used in fixed-point subtraction with the minuend x in AR, the subtrahend y in MB. The subtraction could be performed by taking the 2's complement negative of x, adding -x to y and taking the negative of the result. It is much simpler however to complement the word in AR, then add and again complement the result in AR. The complement of a word, which is produced by exchanging all ones for zeros and zeros for ones, is equivalent to the arithmetic 1's complement in which the sign is changed and the magnitude is subtracted from all ones, i.e., from $1-2^{-35}$. The complement of x is thus:

$$1 - x - 2^{-35};$$

adding y to $\sim x$ yields:

$$1 - x + y - 2^{-35}$$

which equals:

$$1 - (x - y) - 2^{-35}$$

which is the 1's complement of $x - y$. Overflow is indicated in the same way as in addition; that is, by an AR1 carry without an AR0 carry or vice versa. Overflow is properly indicated even for operations involving -1 because the 1's complement is 2^{-35} more negative than the 2's complement (i.e., when subtracting -1 from 0, the proper carries occur in the addition of -1 to the complement of 0).

c AR Control

Figure 6-8 shows the generating logic for the AR control pulses. The pulse amplifiers that drive the register gates are in the top section; these PAs are triggered through OR nets most of whose inputs come either from the subroutine interface (6.5) or from the transfer gates in

the left center section. Timing inputs to these gates are supplied by the various main sequence and subroutine chains, and the gating levels for the execute time pulses are supplied by the OR nets at the lower right. Six of these diode nets are used almost exclusively to gate the AR events necessary for the execution of the Boolean instructions. The flow chart (Figure 4-8) lists at each time the instructions that require each event, and also lists the specific sequence of events necessary for each of the 16 Boolean functions. Because there are no complement gates for MB, all functions that require the AND with the complement of memory use the OR and complement the result; the same is true for the converse. BOOLE 14, which places the complement of memory in an accumulator, clears AR first and then uses the exclusive OR as a 1 transfer. All other functions are quite straightforward. The other diode nets in Figure 6-8 gate the events necessary for the data transmission, IOT, and compare instructions; the functions that are not immediately obvious from the flow charts are described in 4.5.

The pulse amplifiers that clear AR are shown in the upper left corner of Figure 6-8. To leave the desired address in the right half, only the left half is cleared in the address cycle and the block transfer subroutine. For console operations, both halves are cleared by a signal from the key logic (5.1) and the subsequent transfer from the DATA switches is triggered through the PAs in B7. The other AR clear input is generated by the nets in B3 and C3. Subroutines that require the clear include divide, floating add, and multiply. The clear may also occur at the first two pulses in the execute cycle: it is required at ET0 for four of the Boolean functions (D2), at ET1 by any input IOT (D6) and in any half-word transfer that affects the other half of the destination location. For input instructions, IOT T3 triggers the subsequent transfer into AR from the bus (B8). For an HWT, there is no further action if the instruction is to clear the other half, or the transfer half is positive and the instruction is to extend it; however, to set, or to extend a negative half, the other half is set by complementing through the appropriate gate in B1.

All other complements affect the entire word. One input to the PA gates (A2) comes from the subroutine interface; another is the first pulse in the add-subtract time chain—this pulse occurs only if the subroutine is to subtract (d below). Corresponding time pulses for the index and negate subroutines trigger the complement through the top gate in C2. The gates below that provide the complement at ET0 or ET4 for Boolean functions, and at ET5 and ET7 for the ACBM

group in case the masked bits must be cleared at ET6. The bottom gate complements at the end of the carry function in an AR subroutine if the complement control flip-flop has been set.

The four PAs in A3 and A4 are triggered in several combinations to provide 0, 1, and jam transfers from MB into either or both halves of AR. All four must be triggered for a full-word jam transfer. The top two inputs provide for a simultaneous interchange of MB and AR; for any such transfers not made through the subroutine interface, the generating logic is shown with MB control (6.1). Half-word jam transfers include transfer of the address portion of the instruction word into ARRT at the beginning of the address cycle, and transfer of the selected half word in an HWT triggered by ET4 through the gates in B2 with gating levels supplied from the nets in C7. The left transfer also occurs in a CONO so that E is available on both halves of the I/O bus. All other pulses that trigger the PAs in A3 and A4 are for full-word, one-directional transfers. Transfers of zeros alone (B4) are required in both the deposit and load character sequences (6.6b), at ET1 in four of the Boolean functions (D4) and at ET6 for the two IOT status test instructions (D8). The 1 transfer (C5) is used to set the masked bits in an ACBM (D4) as well as to provide a 1 transfer in SETM (BOOLE 3) and an OR function in four of the other Boolean instructions. The gates for the jam transfer are located between B3 and B4 in the figure. This transfer is required in fixed-point multiplication (6.7a), at ET0 in various data transmission, compare and IOT instructions (D5), and in character operations in the first part and in the load sequence of the second part (6.6b).

The PAs for the exclusive OR function of MB and AR (A5) are triggered when ET1 is gated at B4 by the level from D5. This function complements the masked bits in an ACBM, provides a 1 transfer in SETCM (BOOLE 14) and an exclusive OR in XOR and EQV (BOOLE 6 and 11). The exclusive OR pulse amplifiers are also triggered by AR AS T1 in the add-subtract subroutine to provide the partial addition. The next pulse in the chain initiates the carry function through the pulse amplifier in B6. The remaining PAs in the upper right of Figure 6-8 provide for the left and right shifts, which are triggered from the subroutine interface.

The control pulses for the special AR gates that are external to the arithmetic register modules are shown at the right in Figure 6-9. To understand the significance of these transfers, the reader should refer to the description of the subroutines in which they occur. In the exponent calculate and floating add subroutines, AR bits 1-8 are cleared or set according to whether

the sign bit is 0 or 1 in order to nullify the exponent part of the register. The jam transfer from SC1-8 into AR occurs in the normalize return and floating-scale subroutines to insert the exponent in the result; a similar transfer of SC3-8 to AR0-5 occurs in the first part of a character operation to insert the new position if the pointer has been incremented.

d AR Subroutines

Figure 6-9 shows the logic governing the negate, index and fixed-point add and subtract subroutines; the flow chart for these is in the upper right of Figure 4-8. In the lower right of Figure 6-9 are the nets that provide the level gates to trigger and control the subroutine time chains. The add gate is asserted only for fixed-point addition, the subtract gate for both fixed-point subtraction and the instructions that arithmetically compare an accumulator against memory. The add 1 and subtract 1 levels are generated by instructions requiring upward or downward indexing of either the entire AR or its two halves independently. AR+1 is asserted in any MEMAC instruction that adds 1 to the test word, in the two add-1-to-both jump instructions, and to index the pointer in any pushdown instruction or block IOT. Similarly, AR-1 is asserted for the MEMAC instructions that decrement the test word and to decrement the pointer in the pullout instructions. Between the add 1 and subtract 1 nets is a net that generates the level AR+-1 LTRT, which causes any entry into the carry chain at AR35 to enter the chain at AR17 as well. The double indexing occurs in the two add-1-to-both jumps and for operations on the pointer in a block IOT or in any pushdown or pullout instruction (these last four are included in the level JPA IR6(0)). The net in the lower right corner generates the level that causes the execute cycle to pause after ET3 for an AR subroutine. AR SBR is generated by any of the four subroutine calling levels described above, and also for negation in a full-word transfer and for a floating subtraction (this last case is for negation of the subtrahend before entering the floating-add subroutine).

The AR subroutines are actually two separate time chains, each with multiple entries, that join for the return to the interrupted sequence. The chain in the upper left handles indexing and negation; downward indexing and negation both require a preliminary pulse to complement AR. Subtracting 1 is done by complementing, adding 1, and complementing again; negation by complementing, then adding 1. All subroutines in the execute cycle are triggered at ET3. AR-1 triggers AR+-1 T0, which complements AR and also sets the complement control

flip-flop (C5). The complement pulse for negation may be triggered by a signal from the subroutine interface (6.5) as well as by ET3 in a floating subtraction or the appropriate FWT. Following either T0 pulse, there is a delay to allow the register complement to function before triggering AR+-1 T1 which carries into AR35 adding 1 to the contents of the register. This latter pulse is also triggered directly by time pulses from block transfer, normalize return, and character operations as well as by the input for execute cycle incrementing. For indexing both halves of AR together (this occurs in a block transfer as well as in those instructions that generate AR+-1 LTRT), the same pulse that carries into AR35 is also gated to carry into AR17. Of course, a carry into AR17 occurs automatically whenever there is a carry out of AR18.

Just to the right of the index and negate chain is a separate time chain for addition and subtraction. The first pulse in the chain, which complements AR, is necessary only for subtraction and is triggered not only in the execute cycle, but also from the subroutine interface, and in a block transfer to determine whether the block is complete. AR AS T0 sets the complement control flip-flop, and after a delay sufficient for the complement to function, triggers the partial add pulse AR AS T1. The chain begins at this pulse for any execute cycle or subroutine addition and also when adding an index register in the address cycle. After the partial addition, the chain continues to the next pulse which initiates the full register carry.

There are three pulses that initiate carries in AR. These are AR+-1 T1 in the right half, AR17 CRY IN in the left half and AR AS T2 which triggers carries at any required point throughout the register. All three pulses are applied to the OR gate in B1 cutting off the transistor just to the right by grounding pin 2F5Y. ORed with these carry initiating pulses is a vast OR gate (lower left) that receives the carry outputs from all AR bits. Once any carry function is initiated, all further carries are from the ripple carry, i.e., a carry for any bit can come only from a contiguous bit. Successive carries overlap because of the carry speed and the OR gate is enabled until all carries die out: 2F5Y then goes negative triggering the carry completion pulse. If the complement of AR is not required (i.e., the subroutine is not performing a subtraction) the completion pulse triggers the subroutine return ART3. However if the complement control flip-flop has been set, the carry completion complements AR, and after an appropriate delay triggers the return. ART3 clears the complement control flip-flop and triggers the next pulse in the time chain that has called the AR subroutine.

e AR Flags

The jump-addressable flags and associated logic are shown in the right half of Figure 6-10. Although some AR flag control functions also affect the user flag and the flip-flop that differentiates the two parts of a character operation, only the four AR flags can be sensed or cleared directly by the program. The PC change, overflow, carry 0,and carry 1 flags are cleared initially by the master start rather than the master clear because their states must remain from one main sequence to the next. Program control over the flags is exercised by three instructions, JRST with a 1 in IR11, JFCL, and the CONO for the processor. The last instruction, which governs the many flags in the processor I/O interface (8.3), clears the PC change flag if a 1 is programmed in bit 29 or the overflow flag if bit 32 is 1. JFCL selects flags with bits 9-12 of the instruction word and clears the selected flags after sensing them to make the jump. The JRST, which must be addressed indirectly, clears all the flags and restores them according to the first four bits of the word taken from memory as the direct address. The flag clear and set pulses generated here also handle the restoration of CHF7 and EX USER.

The remaining gates include conditions for setting the flags so that the program may determine whether or not certain events have occurred. Any jump or skip sets the PC change flag provided the instruction in process is not a JRST that is restoring the flags. All other gates are for arithmetic conditions indicating overflow. Overflow in most fixed-point cases is determined from the pair of flip-flops in the lower right, which are cleared at the beginning of every execute cycle and then set by carries out of AR0 and AR1. Since overflow is indicated by the presence of one of these carries in conjunction with the absence of the other (as is shown in the discussion of the addition algorithm in b above), the level AR OV SET is asserted when the two flip-flops have oppositestates; i.e., the overflow level is the exclusive OR of the two carry flip-flops. In fixed-point addition or subtraction or in the MEMAC instruction group—the latter condition being relevant only for those MEMAC instructions that index the test word—ET10 transfers the states of the carry flip-flops to the corresponding flags, and if overflow has occurred, sets the overflow flag. The remaining gates in C6 set the overflow flag on an overflow signal from the subroutine interface (6.5); on the attempt by an FWT to form the negative of -1; and on the loss of a significant bit in an arithmetic shift to the left, i.e., if a 1 is lost in a positive number or a 0 in a negative number (the exclusive OR input for this gate is generated in E5).

The top nets at the far right in Figure 6-10 provide control levels for use in various arithmetic subroutines: the exclusive OR of AR0 and MB0, and the complementary function of AR0 and SC0. The remaining nets at the lower right test for the inequality conditions in arithmetic compare instructions (5.3). The lower net ANDs the overflow condition with ∿ MEMAC, and this output is exclusive ORed with AR0 in the upper net. In a MEMAC, the output of the lower gate is automatically false and the output of the upper gate is true or false as AR0 is 1 or 0. In this case, since the comparison is with 0, the function specifies whether the test word is negative or positive, i.e., less or not less than 0. In the ACCP group, a test word from AC is compared against a standard, either E or C(E). The comparison is made by subtracting the standard S from the test word T. The four possible configurations of T and S result in the following signs and overflow conditions:

$T \geq 0$, $T < S$: - and ∿ OV

$T \geq 0$, $T \geq S$: + if ∿ OV, - if OV

$T < 0$, $T < S$: - if ∿ OV, + if OV

$T < 0$, $T \geq S$: + and ∿ OV

Hence, the function AR0 ⊻ AR OV specifies the relation between the test word and the standard: it is true for $T < S$, false for $T \geq S$.

6.3 MULTIPLIER-QUOTIENT REGISTER

Figures 6-11 and 6-12 show the left and right halves of the full-word, multiplier-quotient register. Each MQ flip-flop has a direct clear input, which receives the clear pulse for the register, and gatable clear and set inputs. The MQ modules include four sets of internal gates, but the gatable inputs are also available at the connector so that transfers can be made by single-bit pulses from external gates (at present, these are used only for MQ0).

The upper two sets of gates provide transfers of zeros or ones from MB. The lower two rows are jam transfer gates for right and left shifting; gating levels for a given bit are the 0 and 1 outputs of the adjacent bits. Since the connections at the ends of the register vary depending upon the type of shift, there are special inputs for the left shift gates at MQ35, the right shift gates at MQ1 and both sets at MQ0. The generating nets for the special shift inputs are shown in the lower portion of Figure 6-7. At the left are the different types of shift with the

time pulses at which they occur and block diagrams showing the shift configurations. These diagrams also appear at the appropriate places in the flow charts. Among the level inputs to the shift nets are two composite functions that represent groups of conditions, all of which require the same shift type. The arithmetic shift of AR and MQ combined (B4) is required by the corresponding shift operation, but it is also required in the normalize return and floating-add subroutines. Another composite function is SHC DIV which is asserted by any type of division provided NRF2 is 0 (D4). The flip-flop condition does not apply to fixed division but is necessary in control over floating division so that the control level cannot affect shifting in the normalize return subroutine, which follows all floating-point arithmetic subroutines.

The four shift input nets use these composite functions as well as individual instruction levels to determine the effect of any given shift on the MQ extremities. In left shifting (C4), MQ0 receives the state of MQ1 unless some type of double-length arithmetic shift is being performed in which case AR0 is the source. The next net controls MQ0 whenever MQ is shifted right. AR0 is shifted into MQ0 by any double-length arithmetic shift or by the final multiplication shift, i.e., when the shift counter contains 777. In all other circumstances, AR35 provides the input. The next gate to the right causes information to be transferred directly from AR35 to MQ1 in any double-length right arithmetic shift (MQ0 is skipped in the shifting), otherwise MQ0 shifts into MQ1. The final net connects AR0 to MQ35 for any left combined rotation but causes MQ35 to receive the complement of AR0 in a division shift. The two gates below provide constant inputs to MQ35. The left gate grounds the 0 input so that ones are always shifted in the first part of character operation; similarly zeros are shifted in a combined logical shift, in any double-length arithmetic shift, and in the character deposit sequence. There is a single additional gate that affects MQ0 when MQ is not itself shifted. In the first floating division shift at DST10A, only AR is shifted but AR35 must enter MQ0. The pair of gates at the left in Figure 6-13 allow entry through the single-bit pulse inputs, leaving the remainder of the register unaffected.

For use in multiplication, the register actually has an additional bit MQ36 (Figure 6-13, right). This bit receives the state of MQ35 on any right shift but is of significance only in multiplication (6.8b). For the multiply subroutine, the net at the right supplies a level indicating when bits 35 and 36 are equal. Another net in C4 supplies the exclusive OR function of MQ35 and MB0 for use by the divide subroutine (6.8c).

At the top of the figure are the pulse amplifiers that control the register. MQ is cleared only at the master clear and the shift PAs are triggered only from the subroutine interface (6.5). The remaining PAs provide transfers of zeros and ones from MB. Transfer of ones occurs in the deposit character sequence (6.6b). The jam transfer occurs on a signal from the subroutine interface that switches MB and MQ, and at the four time pulses listed at the gate in B4. The transfer at FT4 saves E in MQ if there is to be a second AC fetch operation; FT4A transfers this additional word into MQ at the same time that it returns E to MB. The remaining transfers occur in the multiply and divide subroutines (6.8b, c).

6.4 ARITHMETIC SHIFT COUNTING

In addition to the three full-word arithmetic registers, the arithmetic logic also includes two 9-bit registers for use in auxiliary computations and counting steps in arithmetic operations. The shift counter and the floating-exponent register are shown in Figure 6-14. Each module in the figure includes one flip-flop from each register and all associated gating. FE is used only for storage of intermediate results. In floating multiplication and division, the exponent is calculated in SC and stored in FE while SC counts the number of shift steps in the operation on the fractions. FE also provides temporary storage for the position portion of the pointer in a character operation.

Since actual computations are performed in SC, there is considerable gating associated with its flip-flops, including a carry chain; but FE is used only for storage so it includes only a direct clear and two sets of transfer gates connected to the flip-flop collectors. These provide 1 transfers from SC and from MB0-5 to FE3-8.

a SC Gating

Below the SC flip-flops in Figure 6-14 are the gates that implement the transfers, partial addition, and carry logic. In addition to gatable clear, complement, andset inputs, each flip-flop also has a carry input and a carry output. These are connected from one flip-flop to the next so that a pulse at the SC+1 input to SC8 adds 1 to the contents of the counter. The clear input is used only for the register clear. There are three sets of complement gates: the top one provides a simple complement function and the other two provide the partial add and carry functions which are described below. The remaining gates are connected to the 1 inputs.

The top gate provides an ordinary 1 transfer from FE to SC. The middle gate provides a complement transfer from MB18, 28-35 into SC. The last gate is a diode net (not part of the 6203 module) which receives no level inputs, but instead control pulses are applied to individual gates in order to place a specific number in the counter. In multiplication and division, for example, SC receives the complement of the number of steps to be counted; it is then incremented until it contains all ones, terminating the operation.

At the bottom of the figure are two networks, the lower a carry chain, the upper a set of level gates that supply the data inputs to the partial add (exclusive OR) gates at the complement inputs to the SC flip-flops. The partial add gates are used not only for partial addition but also for transfers provided SC is cleared or set first. The source of information for the partial add gates varies depending upon the operation in which SC is being used. The nets that generate the data levels are sets of four AND gates ORed together; each AND gate receives an enabling level that is common to all nets and an input from a single bit of a source register. For example, the top set of AND gates places the complement of MB0-5 into SC3-8, i.e., with SC clear, a 0 in a given MB bit causes the partial add pulse to set the corresponding SC bit by complementing it. For this function, the first three data levels are automatically asserted. For the other three sets of AND gates, the data levels are asserted by ones in the source register: the second set enables input from MB6-11 to SC3-8 with the first three bits negated; the bottom two sets enable input to SC from AR0-8 and MB0-8.

To calculate the exponent in floating-point operations and the pointer in character operations, numbers must be added in SC. The addition is performed with essentially the same algorithm used in the arithmetic register (6.2b). First, the partial add pulse produces in SC the exclusive OR of the contents of SC with the number represented by the data inputs, then the carry pulse adjusts the partial sum to produce the arithmetic sum. The carry function for SC differs from that in AR in that no ripple carry is used. The carry connections from one flip-flop to the next are for indexing and are not associated in any way with the full-register carry function; no carries propagate from one bit to the next when a bit is complemented by the carry pulse. Instead, as soon as the partial sum is formed, a series of level transitions from right to left across the carry chain determines the carries for all bits, and all bits are adjusted simultaneously by the carry pulse. The conditions for a carry are the same as those in AR. There is no carry into

the least significant bit (D8) and a carry out occurs only when two ones are added—a condition that is indicated by a 1 in the data and a 0 in the partial sum. For the other bits, there are two carry out conditions, one dependent upon a carry in. For any bit, there is a carry out if both summand bits are ones, or if the partial sum is 1 and there is a carry in. After the level changes have propagated through the chain defining the carries for all bits, the carry pulse complements those bits that receives carries; the SC8 carry supplies the gating level for SC7, and so on through the register. Since there can be no carry into the LSB, the carry gate for SC8 is disabled (B8).

b SC Control

The various functions of SC and FE are triggered directly from the subroutine time chains including the SC subroutines. There are no connections through the subroutine interface and SC functions are triggered by execute time pulses only prior to entry into a subroutine. The logic governing the SC and FE functions is shown in Figure 6-15. The conditions governing FE, which is used only for temporary storage, are quite simple: it is cleared only on the master clear (B6), and may receive an exponent from SC in the exponent calculate subroutine for floating multiplication or division (B7), or may receive the position portion of the pointer in the first part of a character operation (B4). In all cases, the information is subsequently transferred to SC (B2).

The complement transfer of MB18, 28-35 into SC (C8) is made at ET0 in a shift operation or floating scale. The only other function triggered from the execute cycle is the complement pulse, which occurs at ET1 in FSC if AR is positive (B3). All other functions govern SC for shifting or calculating in the data and arithmetic subroutine instructions, and they are triggered by pulses from the special time chains. In most instances, these pulses are ungated although many of those that trigger the complement are gated by sign conditions. Read the appropriate subroutine description (all are included in the final three sections of this chapter) to determine the significance of each SC event in a given subroutine. SC is cleared through the net in the upper left (which includes the master clear), complemented through the nets in the top center, and its partial add function is triggered through the net in the upper right. This last function is used for addition only in the SC add subroutine in which it is followed by the carry (B3); in all other cases, the exclusive OR is used for transfers. The source of information for the data

levels to the partial add gates is determined by the enabling levels in the upper right of Figure 6-16. A flip-flop in the logic for the appropriate subroutine enables the required input for each transfer or partial addition. There are no flip-flops associated with the SC add subroutine; instead the enable level is derived from a flip-flop in the main subroutine that calls for the SC addition. SC may also be incremented by 1 through the nets shown below the enable levels.

In the lower left of Figure 6-15 are several control signals derived from the SC outputs. The function in D6 indicates that the first three bits are all ones and hence SC contains a 1's complement negative less than or equal to 63 in magnitude. The net in D1 decodes SC0-7 for all ones and its output is ANDed with SC8(1) for a signal indicating that a shift-count has been complemented, i.e., SC has counted to -0 which is all ones. The termination of the count may also be indicated by a pulse through the pulse amplifier in C4. When SC0-7 are all ones, the next count pulse in the shift-count, multiply or divide subroutine triggers the PA to produce a leading edge at output SC8B. The PA output is ORed with the -0 configuration, so SC8B remains asserted even after the PA output disappears.

c SC Subroutines

There are two subroutines associated with the shift counter, an add subroutine for use in calculations on exponents and pointers, and a shift and count subroutine that counts the number of steps required in an operation and shifts the intermediate result at each step. The logic for these subroutines is shown at the left in Figure 6-16 and the flow charts are in the lower right of Figure 4-8. Listed with the shift-count subroutine are the entry conditions, the control levels governing the type of shift, and the pulses to which the subroutine returns in the interrupted sequences. Similarly, the flow chart for add lists the entry, the source enabled for the partial addition, and the return for each call.

The time chain for add is in the upper left in Figure 6-16. The first pulse always clears the character control flip-flop CHF1, although this is of relevance only in character operations. SAT1 then triggers the partial add, and after a delay sufficient for all level transitions through the carry chain, SAT2.1 triggers the carry. SAT3 then returns to the interrupted sequence.

The shift-count subroutine is used only in character operations, shift operations, and the floating-add subroutine. All other shifting counted by SC is produced directly by pulses from the arithmetic subroutines. Entry into the shift-count sequence is at SCT0, which performs no operation but provides a delay before the first shift. If SC does not contain - 0 (indicated by the condition that either SC8 is not 1 or SC0-7 does not contain all ones), SCT1 increments SC and triggers the appropriate shift (6.5). If the counting is still incomplete, the output of the delay triggered by SCT1 again triggers SCT1 for a new shift and count. When the count is complete, the delay output returns to the interrupted sequence via SCT2.

6.5 SUBROUTINE INTERFACE

Because the same event is often required at many different times in the various subroutines, the processor includes a subroutine interface that collects signals from the subroutines to reduce the number of signals applied to the control logic for the arithmetic registers. For example, all shifts of AR or MQ are triggered through the nets at the left in Figure 6-17. The lower set of nets, which includes level gates, is for shifting on SAT1 in the shift-count subroutine. For a shift operation, AR and MQ are shifted left if bit 18 of the instruction word is 0, right if bit 18 is 1. Note that even though the program may request the shift of a single accumulator, the logic shifts both AR and MQ, the latter being empty. The level gates for other SC-controlled shifts are supplied by subroutine control flip-flops. The first part of a character operation requires an MQ left shift; this is followed in the second part by a shift left of both in the deposit sequence or an AR shift right in the load sequence. Floating addition requires a right shift of both registers. The upper set of gates allows pulses from the subroutine time chains to trigger shifts. The regular division shift is both registers left at DST14A, but two other divide pulses produce right shifts of AR alone and one left shifts MQ alone. Similarly the multiplication process shifts both to the right at MST2, but a final shift at the end of the subroutine moves MQ alone to the right. The normalize return begins by shifting both registers right in case there has been overflow in calculations with the fractions, and then the regular normalizing process shifts both to the left. In floating division, a 2-bit overflow is possible; so following the divide subroutine, the final pulse in the floating-divide instruction sequence (FDT1) shifts both registers right to supplement the single shift that begins the normalize return. The effects at the register extremities for all of these shifts are controlled

by the special shift inputs shown in Figure 6-7 and described with the gating for the registers (6.2a, 6.3). The exact configurations for all shifts are shown in block diagram form in Figure 6-7 and each shift is shown at the appropriate place in the flow charts.

The remaining nets in Figure 6-17 are mostly for gates that collect subroutine time pulses for transmission to the register gating. In some cases, the pulse inputs are gated by levels, particularly in the upper right nets that detect overflow. All connections from subroutine time chains, other than for functions listed at the top of the figure, are made directly to the control logic for the registers. All subroutines pulses that trigger the AR negate or add subroutine, the switch of MB and MQ, or the AR complement are routed through the subroutine interface. For the other functions, which include entry to the AR subtract subroutine, the transfer of MQ to MB, the switch of MB and AR, and overflow, most pulses are routed through the interface but some are connected directly to the register gating. For example, the net in the upper right of Figure 6-17 handles overflow for multiplication, division, and all floating-point operations, but overflow in an arithmetic shift operation is handled by a net included with the flag logic (6.2e).

Except for one special case, all SBR flip-flops and control flip-flops in the subroutine logic are cleared at the beginning of every main sequence. For this purpose, the master clear triggers several pulse amplifiers to drive additional clear lines. Two of these with prefix MP are in the lower right of Figure 6-21, a third with prefix DS is in the upper right of Figure 6-26.

6.6 DATA SUBROUTINE INSTRUCTIONS

Three types of data transmission instructions switch to subroutines for their execution. Flow charts for all three types are in Figure 4-9. The block transfer moves an entire block of words from one area in memory to another. The character operations handle single characters smaller than a word and can insert a character into a word in memory or retrieve a character from a word without affecting the rest of it. Shift operations move the bits of a word or pair of words to the left or right. There are several shift configurations differentiated mainly by the effects

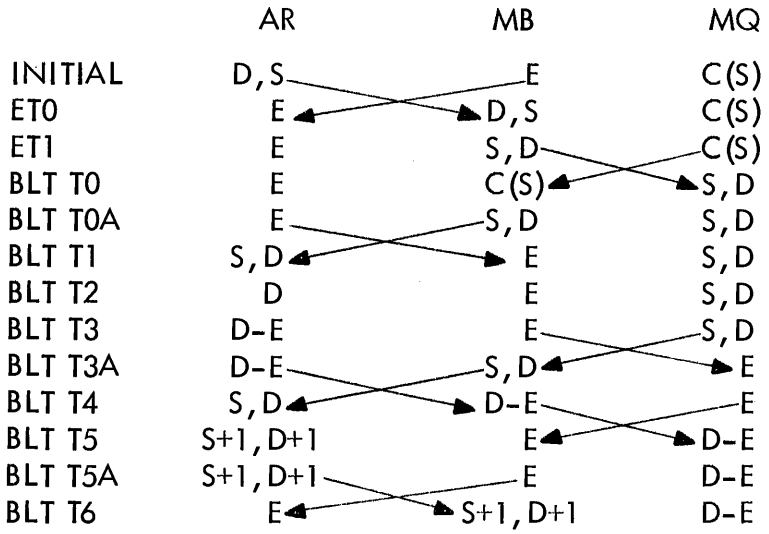
of the shift on the register extremities. The last group may be viewed as logical operations rather than data transmission; the arithmetic shift is equivalent to multiplying the word by a power of 2.

a Block Transfer

A flow chart of the block transfer instruction is at the right in Figure 4-9 and the time chain is in Figure 6-18. The left and right halves, respectively, of the accumulator addressed by the instruction provide source and destination addresses S and D. The first fetch cycle retrieves a word from location S. The subroutine then stores the word in location D, increments both S and D, and returns to the fetch cycle to retrieve and store a second word according to the incremented source and destination addresses. The entire sequence is iterated until D equals the effective address E.

The first fetch cycle retrieves AC, swaps its halves so that S is available to MA, and fetches C(S). At the beginning of the execute cycle, AR contains (D,S), MB contains (0,E), and MQ contains C(S). The first execute pulse switches MB and AR to save E and bring (D,S) to MB. ET1 then clears MA and swaps the MB halves so ET3 can transfer D to MA. ET3 also triggers the first subroutine pulse BLT T0 (Figure 6-18, upper left) which switches MB and MQ to save (S,D) in MQ and make C(S) available to memory from MB. It also requests a memory write to store C(S) in D.

Upon receipt of the memory return, the BLT time chain transfers MQ to MB so that the addresses are now in both registers. The next pulse then switches MB and AR so that E is now in MB and (S,D) in AR. BLT T2 clears ARLT and the next pulse calls the subroutine to subtract E from D. Following the AR subroutine, BLT T3A places E in MQ and returns (S,D) to MB. BLT T4 then moves D-E to MB and the two addresses to AR. The next pulse moves the subtraction result to MQ, bringing E to MB, and triggers the subroutine that adds 1 to both halves of AR, incrementing both addresses. Upon the return BLT T6 saves E in AR, moves the new addresses to MB and reenters the fetch cycle at FT1A. This is the point just following the retrieval of an accumulator, so the processor repeats the entire procedure using the incremented addresses in MB as though they had just been retrieved from AC. For convenience, the following table shows the contents of AR, MB, and MQ following each pulse in the sequence (or following a subroutine called by the pulse).



Since E is initially greater than D, the result of the address subtraction is negative until the cycle following that in which the indexing of D makes it equal to E; then the result is 0 so the sign is positive. The transfers at BLT T4 and BLT T5 move the result to MQ, and BLT T5A tests MQ0 (D4, B8) to determine whether the block is complete. If MQ0 is 0 at this time, the program counter is incremented (the normal program counting at ET1 is inhibited throughout the block transfer) and the subroutine returns to ET10 instead of going on to BLT T6. There are no operations in the store cycle and the processor goes on to the next instruction.

Since a block transfer may use many main sequences, the subroutine includes provision for strobing the priority interrupt system at every BLT T4. If a PI request is generated, the level BLTDONE is asserted even though MQ0 may not be 0 (if it is, the subroutine terminates in the usual manner). This prevents the final MB-AR switch at BLT T6, so the incremented addresses are still in AR and BLT T5A goes directly to ET10. Since BLTLAST is negated (D6), there is no store-AC inhibit and the current addresses are stored in the accumulator in place of the original ones. Following the store cycle, the processor returns to the instruction cycle ostensibly to repeat the same instruction but it is interrupted by the PI request. After all requests have been serviced, the program returns to the interrupted block transfer, fetches the new addresses from AC, and begins where it had previously left off.

b Character Operations

There are five instructions in the character operation group, four of which require two main sequences for execution: the first part fetches and if necessary increments the pointer, the

second handles the character designated by the incremented pointer. The flow chart for all character operations occupies the left half of Figure 4-9 and the logic for the two parts is shown in Figures 6-19 and 6-20. The top of the flow chart lists the different instructions, the main control levels governing their execution, and the configuration of the pointer. The first fetch cycle fetches the pointer according to the effective address calculated from the instruction; the address cycle in the second part calculates the effective address of the operand from the I, X, and Y portions of the pointer. Within the operand, the character is defined by the P and S portions of the pointer: S specifies its size; P specifies its position as the number of bits remaining to the right of the character in the word. Two of the instructions merely fetch the pointer in the first part and then one enters the load sequence in the second part, the other the deposit sequence. The load sequence retrieves a character of size S from position P in the word in location E and loads it right justified into AC. The deposit sequence fetches a character of S bits from the right end of AC and inserts it at position P in C(E). Two other instructions increment the pointer in the first part and use this new pointer in the load or deposit sequence that follows. The last instruction merely fetches and increments the pointer and then returns to the instruction cycle to continue the program, skipping the second part.

The major control levels for the first part are derived from the instruction command levels and two control flip-flops CHF5 and CHF7, both of which are set by the final pulse in the first part. The first flip-flop distinguishes the two parts, the second compensates for the fact that a PI request can interrupt the program between the two parts. The level CH INC is asserted during the first part of any instruction that increments the pointer (Figure 6-19A2). This level is ANDed with CHF7(0) to generate CH INC OP, whose assertion indicates that the first part must actually increment the pointer. If the pointer is incremented and a priority interrupt occurs, the program must not reincrement upon repeating the first part after returning to the interrupted sequence (the first part is repeated only to fetch the pointer that was lost). Unlike most control flip-flops, which are cleared by the master clear, the state of CHF7 (D3) must remain from the first to the second main sequence and is cleared by the master start via the flag clear (6.2e). When it is set by CHT9 at the end of the first part, the sequence returns to the address cycle at which point a priority interrupt may occur. If an interrupt merely executes a block IOT, CHF7 remains set for the return. However if there is a jump to a subroutine, the JSR saves CHF7 with the flags and clears it so that it may be used by the break

routine. When the routine is complete, the restoring JRST again clears CHF7 via the flag clear and restores its original state from MB4 at the same time that it restores the flags. Then when the program repeats the first part, CHF7(1) inhibits CH INC OP and instead causes CH INC to assert CH ∿ INC OP (C5). This level is also asserted in the first part of the two instructions that do not increment the pointer. Finally, CHF7 is cleared by either sequence in the second part.

The first part fetches the pointer from location E: CH ∿ INC OP allows memory to rewrite the pointer, but CHINC OP requests a fetch and pause so that the incremented pointer may subsequently be deposited. Since there must be only one program count for each instruction, PC+1 is inhibited by all but the IBP (CAO), which uses only the first part. If there is no pointer incrementing, ET0 triggers the subroutine time chain at CHT6 (Figure 6-19B4); otherwise it starts at CHT1 (upper left). The first pulse transfers the pointer to AR and sets CHF1, enabling the zeros of MB0-5 as data inputs to SC so that the partial add at the next pulse loads -P-1 (i.e. ∿ P) into SC. CHT3 then sets CHF2 and calls SC add. CHF1 is not cleared until the first pulse in the SC add chain to give the CHT2 partial add a little more time. CHF2(1) then enables the ones from MB6-11 for the partial add in the SC subroutine so that upon the return SC contains -P+S-1. If there are not enough bits left for another character; i.e., if $S \geq P + 1$, the result in SC has a positive sign and the instruction must go on to the next location for the character. Thus if SC0 is 0, the chain continues to CHT4, which clears SC and calls an AR subroutine to index the address portion Y of the pointer to the next location. Upon completion, CHT4A loads -37 (i.e., -P-1 for a character of 0 size) into SC and returns to CHT3 to call SC add again; this time the addition of S generates the complement of the position portion of the pointer for the new location. If SC0 is 1 at the junction following CHT3A, the chain skips to CHT5 (B3), which complements SC so that it now contains P-S, the position of the next character in the same location or the first character in the next location. Actually CHT4A loads -229 into SC so that an S larger than 36 cannot put the processor into a loop. The result is then interpreted mod 64 so the correct position results. Thus if the pointer must go to the next location and S is larger than 36, the new P is 100 -S rather than 36 -S.

The chain then continues to CHT6, which is the starting point for any nonincrementing instruction. CHT6 again sets CHF2 but there is no subroutine call: CHF2(1) merely enables the MB6-11(1) data inputs to SC so that the partial addition at CHT7 transfers S into it. For

CH ∿ INC OP, the chain then skips to CHT8B. CH INC OP uses the CHT6 and CHT7 operations already mentioned because S is still available from the old pointer in MB; additional events specifically for CH INC OP are that CHT6 inserts the new P into the pointer by transferring it from SC3-8 into AR0-5 (dropping SC0-2 means SC mod 64 is transferred) and clears SC. CHT7 then moves the new pointer to MB after which CHT8 restarts the read/write memory cycle to deposit it. The return triggers CHT8B which clears both CHF6 and CHF2, transfers the new P from MB0-5 to FE and complements S in SC. If incrementing the pointer is the only operation required by the instruction, the subroutine terminates here and the sequence returns to ST7 for a new instruction cycle. For any character operation other than an IBP, CHT8B calls the shift-count (C2) which shifts MQ left S places loading ones in at MQ35. Upon the return, CHT8A clears SC and IR13-17, the latter in preparation for receiving the I and X portions of the pointer in the address cycle following the first part. Then CHT9 transfers P from FE to SC, sets CHF5 so that the next execute cycle will select the second part, and sets CHF7 in case there is a PI request at the beginning of the address cycle to which the sequence then returns.

The two chains for the second part are shown at the top in Figure 6-20. Both sequences start with MB containing the word retrieved according to the effective address of the pointer, and MQ containing a word made up of ones in the last S places at the right and zeros elsewhere. For the two load instructions, AR contains E. The initial gates include FC(E) as there will be no subsequent storage in E, and ET0 triggers the first pulse in the chain (A2). LCT0 moves the data word to AR and the mask to MB, complements P in SC, and calls the shift-count subroutine. The character is then right justified by right shifting AR P places. Following this, LCT0A transfers zeros from the mask into AR thus clearing all of AR except that part containing the desired character. The pulse also clears CHF7 and returns to ET10.

The two deposit instructions request a fetch and pause, and ET0 enters the deposit sequence (A4). DCT0 complements P in SC and calls a shift-count that moves AR and MQ left P places loading zeros in at the right in both registers. Upon subroutine completion, AR contains the character in the appropriate position and the ones in the mask are in the same position. The pair of pulses triggered by the return transfers the mask to MB, the ones from MB to MQ, and complements AR. Thus MB contains the mask, whereas MQ contains the data word other than in the character position which contains all ones. DCT1 then transfers zeros from the mask to AR clearing it other than in tne character position which contains the complement of the

character. At the same time, the data word with ones in the character position is moved back to MB. The next pulse again complements AR so that it contains ones outside of the character, and DCT3 then inserts the character into the appropriate position in the data word by transferring zeros from the character (all other bits are ones) into the all ones portion of MB. This pulse also clears CHF7 and returns to ET10. The subsequent store cycle restarts the waiting memory cycle to deposit the data word in E.

If the program specifies a size greater than 36, the character is at most the entire word. For $P \geq 36$, no character is processed. If both P and S are less than 36 but $P + S > 36$, a character of size 36 - P is loaded from position P or the right 36 - P bits of the character are deposited in position P.

c Shift Operations

The lower part of Figure 6-20 shows the logic governing the shift operations and the flowchart for them occupies the right portion of Figure 4-9. The three combined instructions generate the level SHAC2 (B6) which causes the main sequence to fetch and store a second accumulator. The direction of the shift is specified by bit 18 of the instruction word (0 left, 1 right), and the number of places to be shifted is specified by bits 28 to 35.

At the beginning of the execute cycle, AR contains AC; and for a combined shift, MQ contains a second AC. ET0 transfers the complement of MB18, 28-35 into SC. Since left shifting is considered to be positive and right shifting negative, it is assumed that if bit 18 is 1, bits 28 to 35 contain the 2's complement of the number of shifts desired. Thus if MB18 is 0, SC is already correct and contains the complement of a positive number; however if MB18 is 1, SC contains a positive number one less than the number of shifts. Thus MB18(1) gates ET1 to trigger SHT0 (B5), which adds one to SC. Then ET3 starts the subroutine chain for all instructions and the first pulse SHT1 complements SC on the condition MB18(1) so that SC now contains the correct complement. SHT1 also calls the shift-count, which counts SC up to all ones and at each count shifts AR and MQ left or right according to the state of MB18. The shift connections to the registers are made through the subroutine interface (6.5), and the special shift inputs that control the shift actions at the register extremities are shown in Figure 6-7 and described with the AR and MQ gating (6.2a, 6.3). Block diagrams below the flowchart show the configurations for all twelve types of shift.

Since an arithmetic shift multiplies fixed-point numbers by powers of 2, an overflow condition is included in case significant bits are lost in a positive shift. In a single- or double-left arithmetic shift, the overflow flag is set if a 1 is shifted out of AR1 in a positive number, or a 0 in a negative number (6.2e). The return from the shift-count triggers SHT1A which retruns to the main sequence at ET10.

6.7 ARITHMETIC INSTRUCTIONS

This section describes fixed multiply and the floating-point instructions. Each of these instructions goes from the execute cycle to a special sequence which may or may not call an arithmetic subroutine. Fixed add and subtract are both performed within the execute cycle and are described with the arithmetic register (6.2). Fixed divide enters directly into the divide subroutine from ET0 and is described with that subroutine in 6.8c.

a Fixed Multiply

A flow chart of the two fixed-multiply instructions is in the left part of Figure 4-8; Figure 6-21 shows the special time chain for them. Both integral and fractional multiplication use AC and either C(E) or E itself as operands, but the product in the latter case is a double-length fraction, whereas in the former it is assumed that the desired integer is in the low-order half of the double-length product. Both instructions enter the special sequence at ET0 by triggering MPT0 (Figure 6-21, upper left). This pulse sets the appropriate bits in SC to count 35 steps and sets MPF2 if both operands are negative. It also calls the multiply subroutine (6.8b) and waits until MST6 returns to MPT0A. If both operands are negative and the result is also negative, MPT0A sets the overflow flag. This can occur only if -1 is multiplied by -1, whose answer +1 overflows generating the representation for -1. If integral multiplication is being performed (IR6(0)) and the result is negative, MPT0A complements the high-order half in AR. At this point, the fractional process is complete and the sequence returns to ET10 via NRT6, the final pulse in the normalize return subroutine. In the store cycle, the low-order half is stored in a second accumulator for all but the memory mode wherein no accumulator is stored. For integral multiplication, MPT0A continues the chain to MPT1, which transfers the resulting integer in the low-order half from MQ to MB and sets the overflow flag if the high-order half is not clear. MPT2 then transfers the result to AR so that the sequence can make use of the standard transfer and store functions following the return to ET10 via NRT6.

b Floating Scale

This instruction allows the program to change the exponent of a floating-point number without affecting the fractional part. The number in AC is multiplied by 2^y where y is the number contained in bits 28 to 35 of the effective address. This number is interpreted as positive or negative in 2's complement notation as the sign, bit 18, is 0 or 1. The flowchart for the instruction occupies the left quarter of Figure 4-10, and the time chain is shown in the lower right of Figure 6-19.

The first pulse in the execute cycle transfers the complement of MB18, 28-35 to SC. The next two pulses then adjust SC according to the sign of the number in AR: if positive, ET1 complements SC; if negative, ET3 adds one to SC by triggering FST1. In either case, ET3 triggers FST0, which calls the SC add subroutine. During the subroutine, the 1 state of FSF1 enables the AR0-8(1) data inputs to SC. The return pulse FST0A transfers the new exponent from SC1-8 to the exponent part of AR; and if the signs of AR and SC are different, it sets the overflow flag.

To see that the above sequence of events produces the correct exponent and properly detects overflow or underflow, consider the various cases keeping in mind that the floating-point exponents from -128 to +127 are represented by the numbers 0 to 255 and that the scale factor in E is in 2's complement notation. Let x and y be the absolute values, respectively, of the exponent part of AR and the scale factor in MB. Thus if AR is positive, AR1-8 contains x; otherwise $\sim$ x, i.e., 255 - x. On the other hand if MB is positive, MB28-35 contains y; otherwise the 2's complement, i.e., 256 - y. At each step, C(SC) is a function of the signs of AR and MB as follows:

	AR+, MB+	AR-, MB+	AR+, MB-	AR-, MB-
ET0	-[255 - y]	-[255 - y]	+[y - 1]	+[y - 1]
ET1	+[y]	-[255 - y]	-[256 - y]	+[y - 1]
ET3	+[y]	-[256 - y]	-[256 - y]	+[y]
SC ADD	+[x + y]	-[255 - (x + y)]	+[x - y]	-[255 - (x - y)]

Hence with no overflow or underflow, SC and AR have the same sign and SC contains the proper representation of the new exponent. However, if in the first two columns $x + y > 255$ (overflow)or in the last two $x - y < 0$ (underflow), AR and SC have opposite signs.

c Floating Add-Subtract

Both of these instructions use the floating add time chain shown in the upper half of Figure 6-22; the flow chart occupies the center portion of Figure 4-10. For floating add, ET0 triggers the special sequence (Figure 6-22A1); but for floating subtract, ET0 switches MB and AR so that the subtrahend is then in AR, and ET3 calls the negate subroutine. Since the number to be subtracted has now been replaced by its negative, the operands may instead be added and ET4 triggers the floating add sequence.

If the signs of the operands are the same, FAT0 complements SC to all ones. It also sets FAF1 enabling the MB0-8(1) data inputs to SC so that the partial addition at FAT1 produces a 1 transfer if the signs are different, but transfers the complement of the MB sign and exponent if the signs are the same. In either case, the sign of SC is opposite that of AR. FAT1 also triggers SC add, and shortly after, FAT1B clears FAF1 and sets FAF2 (A3) to enable the AR0-8(1) data inputs to SC. Since the signs of AR and SC are different, at least one exponent is represented by its complement, and hence the result of the addition in SC is a 1's complement negative of the difference between the exponents unless there is overflow. Let x and y be the absolute values of the exponents of the numbers in AR and MB. The possible signs, exponent representations, and results are as follows:

AR, MB signs	AR0-8	SC Before	SC After	Result
AR+, MB+	+[x]	-[255 - y]	-[255 - y + x]	$y \geq x$: signs $\neq$
AR+, MB-	+[x]	-[255 - y]	-[255 - y + x]	$y < x$: signs =
AR-, MB-	-[255 - x]	+[y]	-[255 + y - x]	$y > x$: signs $\neq$
AR-, MB+	-[255 - x]	+[y]	-[255 + y - x]	$y \leq x$: signs =

Hence, if the MB exponent is greater than the AR exponent, the AR and SC signs differ after the addition. If the AR exponent is greater, the signs are the same and FAT1A switches MB and AR because the number with the smaller exponent is the one that is shifted. If the exponents

are equal, the signs may or may not be the same but it matters not whether the transfer takes place. Since the result in SC is a 1's complement, no further action on it is necessary if it is negative, and in this case FAT1A jumps directly to FAT4 (A6). However if overflow has produced a positive result, the number in SC is one less than the difference between the exponents (since a 2's complement addition was performed on numbers in 1's complement notation); in this case FAT1A triggers FAT2 which adds one to SC, and FAT3 complements it in preparation for the shift-count.

If the number with the smaller exponent must be shifted more than 63 places (a condition represented by at least one 0 in SC0-2), the addition can affect neither the fraction nor the low-order part so FAT6 clears AR (A8). However if fewer than 64 shifts are required, FAT5 ensures that all bits to the left of the fraction MSB are of no significance by loading the sign into AR1-8, sets FAF3(B4) to generate the correct arithmetic shift inputs to the registers extremities (6.2a, 6.3), and calls the shift-count subroutine. After the number has been right shifted in AR and MQ so that its bits correctly match the MB bits in order of magnitude, the return triggers FAT5A which follows directly from FAT6 if there is no shifting. FAT5A clears SC and again sets FAF1 to enable the MB0-8(1) data inputs. Then FAT7 changes SC to all ones if MB is negative, and the partial add at FAT8 loads SC with the MB sign and exponent or their complements depending upon the state of SC. Thus SC always receives a positive sign and the absolute value of the exponent. The next pulse in the chain then nullifies the exponent portion of MB, and FAT9 calls AR add. If AR was cleared at FAT6, the addition merely transfers MB to it. The return triggers FAT10 which clears those control flip-flops that are still set and enters the normalize return subroutine. The return via NRT6 is directly to ET10 for storage of the result.

d Floating Multiply and Divide

These two simple sequences do little more than call the three subroutines necessary for the execution of floating-point multiplication and division. The flow charts are at the right in Figure 4-10 and the logic is shown in the lower half of Figure 6-22. Both chains are triggered at ET0 and the first pulse in each calls the exponent calculate subroutine. The return from FPT4 places in SC the complement of the number of steps required (27 for multiply, 30 for divide) and enters the appropriate subroutine. The return from multiply at MST7 triggers

FMT0B, which transfers the calculated exponent from FE to SC, sets NRF2 to set up the normalize shift gates, and enters normalize return. NRT6 returns directly to ET10 for storage of the result.

The divide subroutine first tests that division can be performed. If the divisor is less than or equal to half the dividend, the sequence sets the overflow flag and jumps directly from DST13 to ST7—the only normalized number that fails to satisfy this condition is a zero divisor. If the division is executed, DST21A returns to FDT0B, which transfers the calculated exponent from FE to SC and sets NRF2 to set up the normalize shift gates but does not enter the normalize return subroutine. This subroutine can compensate for only one bit overflow whereas floating divide, by doing 30 steps, deliberately generates two extra quotient bits, one for possible overflow, the other for rounding. The divide instruction sequence includes an extra time pulse, FDT1, which shifts AR and MQ right to compensate for the rounding bit. It then enters normalize return and NRT6 returns directly to ET10 for storage of the result.

6.8 ARITHMETIC SUBROUTINES

Besides the simple AR and SC subroutines, there are four arithmetic subroutines that are called by more than one arithmetic operation. Floating multiplication and floating division both begin with a subroutine that calculates the exponent. The multiply subroutine is used by both fixed and floating multiplication, the divide subroutine by both fixed and floating division. All floating-point instructions except floating scale call the normalize return subroutine to normalize the result of their arithmetic computations.

a Exponent Calculate

Floating multiply and divide call a subroutine to calculate the exponent before beginning operations on the fractions. The flow chart for exponent calculate is at the left in Figure 4-11 and the logic for the time chain is shown in Figure 6-23. The nets at the right in the logic drawing generate several levels necessary for execution of the subroutine. These are exclusive OR and equivalence functions of the operand signs and FMF1, the last term providing a distinction between multiplication and division. In multiplication the exponents are added, whereas in division the exponent of the divisor is subtracted from that of the dividend. Since

FMF1 is the SBR for the return to multiplication from the exponent calculation, it is 1 in multiplication but 0 in division; therefore, the sign functions for multiplication are exactly the opposite of those for division.

The first time pulse in either multiply or divide triggers the floating-point time chain (Figure 6-23, upper left). The first floating-point pulse sets SC1 thus loading 128 into SC. The next pulse complements SC if the sign of AR is different from the state of FMF1, i.e., the complement occurs if AR is positive in multiplication or negative in division. FPT1 also enables the AR0-8(1) data inputs to SC and calls the SC add subroutine. At the return FPT1A complements SC if the exclusive OR of AR0, MB0, and FMF1 is false; otherwise it adds one to SC. The latter action is triggered by FPT2, which is generated only if the appropriate condition holds (upper right); the gating for the complement is included in the complement net for SC. The next pulse then calls another SC addition, this time using MB0-8(1) as the SC data inputs enabled by FPF2(1). After this addition, the result is correct for two of the four cases; for the other two, MB0 and FMF1 the same, FPT1B complements SC. Then FPT3 transfers the calculated exponent to FE, clears SC, and nullifies the exponent portions of both MB and AR by loading the appropriate sign into bits 1 to 8. FPT4 returns to the interrupted special sequence.

First, consider multiplication. FMF1 is set, thus at FPT1 the complement occurs if AR0 is 0; then at FPT1A if the signs of AR and MB are not equal, SC is again complemented, otherwise it is indexed. Finally at FPT1B, SC is complemented if MB is negative. On the other hand, FMF1 is 0 for division, so the complement at FPT1 occurs if AR is negative; at FPT1A complementing occurs if AR0 and MB0 are the same, and indexing otherwise. Finally, FPT1B complements SC if MB is positive.

To see that these operations give the correct result, let x and y be the true exponents of AR and MB. Since excess-128 code is used, the exponent portion of AR is $x + 128$ if the number is positive; but if negative, it is $255 - (x + 128) = 127 - x$. Since the result must also be in excess-128 code, the sum of the exponents must be reduced by 128 for multiplication; whereas, in division the difference must be increased by the same factor. For multiplication, the above sequence of events operates in SC as follows (the table items show the sign of SC and the contents of SC1-8 as a positive number):

	AR+, MB+	AR+, MB-	AR-, MB+	AR-, MB-
FPT1	-[127]	-[127]	+[128]	+[128]
SC ADD	+[x - 1]	+[x - 1]	-[255 - x]	-[255 - x]
FPT1A	+[x]	-[256 - x]	+[x]	-[256 - x]
SC ADD	+[128 + x + y]	-[127 - x - y]	+[128 + x + y]	-[127 - x - y]
FPT1B		+[128 + x + y]		+[128 + x + y]

For every case, the result in SC is the correct exponent for the product (but expressed in positive form), unless there is overflow, as indicated by a negative result (SC0 = 1). The exponent remains in positive form during the multiply subroutine, and the subsequent normalize return subroutine checks for overflow and puts the exponent into correct form. For division, the sequence of events is:

	AR+, MB+	AR+, MB-	AR-, MB+	AR-, MB-
FPT1	+[128]	+[128]	-[127]	-[127]
SC ADD	-[x]	-[x]	+[254 - x]	+[254 - x]
FPT1A	+[255 - x]	-[1 + x]	+[255 - x]	-[1 + x]
SC ADD	-[127 - x + y]	+[128 + x - y]	-[127 - x + y]	+[128 + x - y]
FPT1B	+[128 + x - y]		+[128 + x - y]	

which again gives the correct result in positive form.

b Multiply

A single subroutine handles multiplication for both fixed point and floating point; the two types differ only in the number of steps—35 for fixed point, 27 for floating point. The subroutine flow chart occupies the center portion of Figure 4-11; Figure 6-24 shows the time chain. A pulse in the fixed- or floating-multiply sequence loads the complement of the number of steps into SC, sets an SBR and calls the subroutine. To control the AR and MQ extremities in multiply shifting, the two SBR flip-flops are ORed by the net at the right in Figure 6-24.

For fixed point, the subroutine time chain is triggered by MPT0, which is equivalent to ET0; for floating point, the entering pulse is FMT0A, which follows the exponent calculation. Either pulse triggers MST1 (upper left), which moves the multiplier from MB to MQ and the

multiplicand from AR to MB. AR is cleared shortly thereafter. Form MST1, the sequence may continue to either MST2, MST3, or MST4 depending upon the relationship between bits 35 and 36 of MQ, but every step includes the incrementing of SC and the shifting of AR and MQ right one place. Initially, MQ35 contains the LSB of the multiplier and MQ36 is 0. On each successive step, MQ35 contains the next more significant bit of the multiplier and MQ36 contains the bit MQ35 held on the previous step. Arithmetic operations are performed whenever there is a transition from 1 to 0 or vice versa in a pair of multiplier bits, rather than on the basis of a particular bit being 1 or 0. At the 3-way decision following MST1, there is no arithmetic action if the two bits are equal and in this case the chain goes to MST2. If there is a transition from 1 to 0 in the bit pair, the chain goes to MST3, which calls the AR add subroutine. For a transition in the opposite direction, MST4 calls AR subtract. Either return triggers MST3A, which goes to MST2. Thus MST2 appears in every loop whether any AR subroutine is called or not. This pulse increments SC and shifts AR, MQ right one place. It then returns the sequence to the 3-way decision to check the next bit pair (which has just been shifted into MQ35, 36).

At the 3-way decision, the equality branch has the additional condition that the step count is not complete, i.e., SC does not contain 777. In the final shift, MQ36 receives the multiplier MSB and MQ35, the sign. If these two bits are equal, MST2 leaves the loop, jumping to MST5 (lower left). If they are not equal, MST2 returns to the 3-way decision, only two of whose branches are now open—those which necessarily call AR subroutines. Since the step count is complete, the return jumps directly from MST3A to MST5. This pulse clears SC, shifts the low-order half of the product in MQ to the right, and makes its sign equal to that of the high-order half in AR. MST6 then provides the return to the appropriate instruction sequence.

The multiplication sequence is as follows: Initially, MB contains the multiplicand, MQ the multiplier, and AR is clear. AR and MQ are connected so that the low-order bits of the product are shifted into MQ as the multiplier is shifted out. At each step, the current bit of the multiplier is available at MQ35, and the effect of the multiplicand in MB on the partial sum in AR is one binary order of magnitude greater than in the preceding step because the partial sum was right shifted. Thus MB can be combined directly with AR. Since MQ36 is initially 0, the sequence shifts without calling an AR subroutine until a 1 is shifted into MB35.

At this transition, the sequence jumps to MST4 to subtract the multiplicand from AR, which is clear. The shifting then continues until the next transition (which will be from 1 to 0), at which time the sequence goes to MST3 to add the multiplicand to the previous partial sum. The process continues in this way, subtracting the multiplicand at every transition from 0 to 1, adding at every transition from 1 to 0. At the end, AR contains the correct sign and the high-order half of the double-length product. The low-order half is in MQ0-34 with the sign of the multiplier in MQ35. Thus, there is an additional right shift of MQ to move the correct sign from AR0 to MQ0 and to place the low-order fraction in the MQ magnitude positions.

To see that this procedure results in a correct product, consider the positive binary integer:

1 0 0 1 1 1 0 1 1

8 7 6 5 4 3 2 1 0

(The decimal digits below the binary digits are the powers of 2 corresponding to the bit positions.) This number is obviously equal to:

$$\begin{array}{rr} & 100000000 \\ + & 111000 \\ + & 11 \end{array}$$

Now an n-bit string of ones whose rightmost bit corresponds to 2^k is equal to $2^{k+n} - 2^k$, or equivalently $2^k(2^n - 2^0)$; i.e., $2^n - 2^0$ is a string of n ones and the 2^k shifts the string left k places. Thus:

$$\begin{array}{rcccc} 100000000 & = & 2^{8+1} - 2^8 & = & 2^9 - 2^8 \\ 111000 & = & 2^{3+3} - 2^3 & = & 2^6 - 2^3 \\ 11 & = & 2^{0+2} - 2^0 & = & 2^2 - 2^0 \\ \hline 100111011 & & = & & 2^9 - 2^8 + 2^6 - 2^3 + 2^2 - 2^0 \end{array}$$

In this last representation, each power of 2 that is subtracted corresponds to a transition from 0 to 1 (in the direction of increasing significance), whereas each that is added corresponds to the opposite transition. The largest term corresponds to the transition to the sign bit, which is 0 for a positive number. The multiplication algorithm in PDP-6 interprets the multiplier in this manner, alternately subtracting the multiplicand from the partial sum and adding it to the partial sum in the order-of-magnitude positions corresponding to the transitions. If a multiplier

of the same magnitude were negative, it would have the form:

1 0 1 1 0 0 0 1 0 1

- 8 7 6 5 4 3 2 1 0

in which the extra bit at the left represents the sign. The number is now equivalent to:

$$-2^9+2^8-2^6+2^3-2^2+2^1-2^0$$

wherein opposite signs correspond to opposite transitions. The algorithm may thus use exactly the same sequence for a negative multiplier: this time the subtraction of greatest magnitude is detected by the transition to the sign bit, which is now 1.

c Divide

The time chain for the divide subroutine is on two logic drawings, Figures 6-25 and 6-26; Figure 4-12 is the flow chart. Entry is at ET0 in fixed point, at FDT0A in floating-point. Either pulse loads SC with the complement of the number of steps required—30 for floating-point, 36 for fixed-point—through pulse gates at individual bits. For this purpose, ET0 gated by IR DIV generates DS DIV T0 (Figure 6-25A8). Although the subroutine is triggered only by ET0 or FDT0A, there are nevertheless six entries, depending upon the sign of the dividend in AR and upon whether the fixed division is integral or fractional. To make the latter distinction, the fixed-point command level is further decoded into a pair of levels according to the state of IR6 (B1).

For a negative dividend, entry is at DST0 (upper left) for integral or floating-point operations. The first subroutine pulse sets DSF7 (D6) to remember that the dividend was originally negative, and calls the AR negate subroutine. Following negation, the sequence jumps to DST10 for floating-point but continues to DST1 for division of integers. This is also the entry for integral division with a positive dividend. Integral division uses a double-length dividend just as does fixed-point fractional division, but only one accumulator is fetched and it is used for the low-order half. Since the fetch cycle brings AC to AR, DST1 moves the divisor from MB to MQ and the significant half of the dividend from AR to MB. DST2 then switches MB and MQ, placing the dividend in low-order position, and clears the high-order half in AR. The sequence then jumps to DST10 for computations.

The only other entry earlier than DST10 is for a negative fixed-point fraction, wherein ET0 enters at DST3 (A6). For all other entries, the dividend is either one word in length or is already positive. The sequence starting with DST3 is necessary to change a double-length negative dividend into positive form. DST3 sets DSF7 because the dividend is negative and interchanges MB and MQ. DST4 then switches MB and AR so the low-order half is now in AR, and DST5 calls AR negate. After the return (B3), the sequence branches depending on whether AR is clear. The 2's complement of a double-length number is formed by complementing the entire double word and adding one into the LSB. A carry into the high-order half is necessary if the 2's complement of the low-order half is null (the complement was all ones before the LSB addition). Thus if AR is clear, the sequence goes on to DST8, 9: this reverses the preceding interchanges, returning all words to their original positions, and DST9 calls AR negate to form the 2's complement of the high-order half (i.e., to complement it and then add one as required). If AR is not clear, the sequence goes instead to DST6, 7 which also returns all words to their original positions but merely complements AR.

The computational part of the subroutine begins at DST10, which is triggered through the net at the right in Figure 6-25. This pulse begins the subroutine for a positive dividend in floating point or fractional fixed point. Otherwise, it follows from one of the preliminary sequences discussed above. Entry is direct from DST2 or DST7; from DST0A, it is conditioned by $\sim$ DSDIVI, which in this case represents floating point; and if DST9 calls AR negate, entry is made at the subroutine return. For a floating division, DST10 triggers DST10A (lower left) which right shifts AR, moving the LSB of the high-order half into MQ0, so the double-length dividend now has the sign bit in both AR0 and AR1 and the 70-bit magnitude in AR2-35 and MQ0-35. This action closes up the hole between the two halves of the dividend but it also divides the dividend by 2. For fixed division the sequence goes to DST10B, which left shifts MQ, closing the hole but leaving the magnitude the same. The shift brings the complement of AR0 into MQ35 because of the division shift connections, but this is of no significance.

The next step in the process is to compare the divisor with the dividend to determine whether the division can be performed. Thus, the next pulse is either DST11 or DST12 depending upon whether the divisor is negative or positive. The former pulse calls AR add; the latter, AR subtract. Since the dividend is positive, the result of the computation is also positive if the magnitude of the divisor is less than or equal to that of the number in AR. There are actually

three different conditions being tested here for the three types of division. For a fixed integer, AR is clear and the result is positive only for a 0 divisor; if the divisor has anything of significance in it, the quotient cannot possibly be greater than $2^{35}-1$ which does not overflow. For a fixed fraction, the divisor is actually subtracted from the dividend and no overflow is allowed. For a floating fraction, the divisor is subtracted from half the dividend (the preceding right shift having divided it by 2), so there may be one bit overflow but the normalize return automatically compensates for this. Since all floating fractions are normalized, the only case that fails the test is that of a 0 divisor (a dividend of -1 could also fail but this number cannot occur as the result of any floating-point operation). After the return from the AR subroutine, DST11A tests the result. If it is positive, the sequence continues to DST13, which sets the overflow flag and jumps directly to the end of the store cycle. If AR0 is 1, the overflow cannot be greater than allowed in the particular type of division in progress, and DST11A continues the sequence to DST14A (Figure 6-26B7).

The next few pulses comprise the loop that performs the actual division. In division on paper, one subtracts out the divisor the number of times it goes into the dividend, then shifts the dividend one place to the left (or the divisor to the right) and again subtracts out. In binary computations, the divisor goes into the dividend either once or not at all at each step. The loop thus subtracts the divisor to generate a single bit of the quotient. If the subtraction does not overflow, i.e., if the dividend is larger than the divisor, the sign of the result is positive and a 1 is entered into the quotient. If there is overflow, a 0 is entered. To compensate for the overflow, one could add the divisor back into the dividend before going to the next subtraction step. However the PDP-6 algorithm instead shifts first and then adds the divisor back on in the new position. It then continues to shift and add putting zeros into the quotient until the result again becomes positive. This procedure generates the same quotient without ever going back a step.

The following processor operations correspond to the procedure outlined above. If the initial test subtraction produces a negative result, the sequence enters the loop at DST14A, which increments SC and triggers the division shift of AR, MQ to the left. This shift is equivalent to a combined rotation except that MQ35 receives the complement of AR0. The complement of the sign of the result is the next bit of the quotient: if the divisor does not go into the dividend, the result produces a 0 quotient bit. Each loop loads one bit of the quotient into

MQ35, and the low-order half of the dividend is shifted out from MQ as the quotient is shifted in. Following DST14A, the loop checks the previously generated quotient bit in MQ35 to determine what action to take next. If the quotient has received a 1, the divisor is subtracted; for a 0, it is added. The gate is the exclusive OR of MQ35 and MB0 (Figure 6-26, upper left) because the divisor may have either sign. If MQ35 is 1, DST14 subtracts the divisor if positive, or DST15 adds it if negative. A 0 quotient bit reverses the functions. Either DST14 or DST15 sets DSF5 (lower left) to gate the AR subroutine return which goes back to the beginning of the loop (A7). The loop iterates 36 times for fixed division, 30 times for floating. The first bit loaded into MQ35 is not actually part of the quotient but rather the sign bit to the left of the binary point. It must be 0 because the sequence enters the loop only if the divisor is larger than the dividend. The 36 fixed-point iterations generate the sign and 35 magnitude bits of the fractional or integral quotient. Floating-point requires 30 steps for the 27-bit fraction because there may be one bit overflow and an additional quotient bit is computed to allow running.

The test for termination of the process is made not at the end of the loop but in the middle. Thus each step is not shift and subtract, but rather subtract and shift, the first subtraction occurring before the loop. The test for completion follows DST14A. When this pulse generates the final shift and increments SC to 777, the assertion of the SC completion signal prevents the delayed DST14B from continuing in the loop (upper left) and it instead jumps out to DST16.

The remaining operations in the subroutine generate the correct remainder, adjust the signs of quotient and remainder, and place them in correct position. The final shift in the loop places the last quotient bit in MQ, but it leaves the remainder off one place to the left. Therefore DST16 right shifts AR. Since all operations have been performed on positive operands, the remainder should also be positive. If it is already, the sequence jumps directly to DST17A (C4). A negative remainder indicates that too much has been subtracted from the dividend, so either DST17 or DST18 adds the divisor back in (B4). Again this addition may call either AR subroutine, depending upon the MB sign. DST17A then checks the sign of the original dividend: if DSF7 is 0, the sequence jumps to DST19A (lower left); but if the dividend was originally negative, DST19 (A6) calls the AR negate subroutine. Thus at the end, the remainder has the same sign as the dividend. The setting of the SBR for the return from AR negate (lower right) is delayed slightly to prevent the return from the previous subroutine from getting

through both SBR. DST19A then moves the quotient from MQ to MB, and the divisor in the opposite direction. Next DST20 (A6) clears SC and switches the remainder presently in AR with the quotient in MB. The sequence then determines the proper sign for the quotient by checking the exclusive OR of DSF7 and MQ0, which is generated by the net in the lower left. If the operand signs are the same (A5), the result is already in correct form and the sequence jumps to DST21A. If the signs are different, the result should be negative and DST21 calls AR negate. DST21A then moves the remainder to MQ where it is available for storage in a second accumulator. The subroutine returns to ET9 for fixed division, FDT0B for floating division. The reader should note at this point that the remainder is the correct one for a fixed-point instruction; but in the floating case, the remainder in MQ9-35 is correct for a 29-bit quotient which cannot be stored in its present form. For further particulars, refer to the normalize return subroutine (d, p. 6-46).

As an example of the way this algorithm operates, consider a division of 3-bit fixed-point fractions with a dividend of +.100100 and a divisor of +.101. By paper computation, we obtain the quotient this way:

```
        .111
101 ) 100.100
      10 1
      -----
      10 00
       1 01
       ----
         110
         101
         ---
           1
```

Assuming the computer registers to be four bits in length, AR contains 0.100, MQ has 0.100, and MB has 0.101. Before starting the division, MQ must shift left to close the hole, giving MQ = 1.001. The sequence has four steps as follows:

```
        0.100 | 1.001
      -0.101
      ------
        1.111 | 1.001
1 ←     1.111 | 0.010
      +0.101
      ------
        0.100 | 0.010
2 ←     1.000 | 0.101
      -0.101
      ------
```

```
        0.011 | 0.101
3 ←     0.110 | 1.001
       -0.101
        ------
        0.001 | 1.011
4 ←     0.011 | 0.111
  →     0.001
```

The quotient is in MQ at the right, the remainder in AR at the left.

d Normalize Return

All floating-point instructions that operate on the fractional parts of the operands end by calling a subroutine to normalize the result. A floating-point number is considered normalized if the MSB of the fraction (bit 9) is opposite in state to the sign bit or if the magnitude of the fraction is 1/2 (the fraction -1/2 has the same magnitude representation as +1/2). The normalize return flow chart occupies the right half of Figure 4-11, and the time chain is shown in Figure 6-27.

Entry into the normalize return chain is immediate from FAT10, the last pulse in the floating add sequence (upper left). The first pulse in the chain, NRT0.5, sets NRF2 (C5), which enables the shift inputs at the AR and MQ extremities for a double-length arithmetic shift (6.2e, 6.3). NRT0 then adds one to the exponent in SC (which is in positive form, see 6.7c, 6.8a) and right shifts AR, MQ to accommodate any overflow that may occurred in the fractional computations. Since a right shift is equivalent to dividing by 2 and indexing the exponent is equivalent to multiplying by 2, the result is unchanged. The next pulse tests for a 0 result. Since the answer has already been shifted right, the net in the center of the figure generates the gating level as the AND function of AR = 0 and MQ1(0). The test can include all of AR because any operation on the fractions that produces a 0 result clears the exponent part of the register. If there is a 0 result, NRT.1 jumps directly to NRT6, the final pulse in the subroutine (B4). For a nonzero result, the chain goes to NRT1, which complements SC (so the exponent is now in negative form) and enters the normalizing loop (A5).

Multiply skips the right shift because it cannot overflow. The last pulse in the multiply sequence, FMT0B, sets NRF2 and after a delay that allows the exponent in SC to settle (C3) enters the chain at the 0 test. MQ1 is unnecessary for the 0 test in this case, but its inclusion merely

allows attainment of a significant result with certain unnormalized operands for which the result would otherwise vanish. For divide, there must be two right shifts, one to compensate for a possible overflow bit, the other because of the extra quotient bit generated to allow rounding (c above). FDT0B, the pulse in the divide sequence generated by the return from the divide subroutine, sets NRF2 and transfers the calculated exponent from FE back to SC. The final pulse in the sequence, FDT1, then triggers the required extra right shift of AR,MQ; the regular right shift occurs in normalize return. Divide enters at the beginning of the chain but is delayed slightly so that the total delay between FDT1 and NRT0 is ample to allow the preliminary shift-count to settle down. Even though there are two shifts, only MQ1 need be included in the zero test because MQ2 contains an extra quotient bit generated only for rounding.

In the normalizing loop, if AR9 and AR0 are the same and the fraction is not of magnitude 1/2 (the gating levels are generated respectively by the net in C2 and the decoding nets for AR, 6.2), NRT1 triggers NRT2. This pulse adds one to the negative exponent in SC thus decreasing its magnitude, and triggers an arithmetic left shift of AR,MQ. NRT2 also enters a delay whose output retriggers NRT2 if the number is not yet normalized. The loop continues until either AR9 and AR0 differ or the fraction has magnitude 1/2, at which time NRT2 goes on to NRT3. If SC is now positive, NRT3 sets the overflow flag. This does not necessarily mean that the normalizing loop counted from negative into positive exponents—the overflow or underflow may have occurred when the exponent was calculated before multiplication or division. Since the loop uses the negative representation of the exponent, it is already in proper form for insertion into the result if the result is also negative. However, if AR0 is 0, NRT3 complements SC.

At this point, the subroutine must determine whether any rounding action is required. The net that generates the round gate is in C3 and C4. The program specifies rounding by a 1 in IR6, but the gate is asserted only if a rounding action is necessary, i.e., if the MSB of the low-order part of the fraction is 1. The rounding sequence requires the repetition of the entire subroutine, so the assertion of NR ROUND first causes NRT3 to complement SC, making the exponent positive again for the initial right shift. Then NRT3.1 continues the chain to NRT5, which calls the subroutine that adds one to AR. The completion triggers NRT5A (lower right) which returns to the beginning of the subroutine (C2). Thus the entire routine is repeated in

case the rounding has overflowed (which can occur only if the fraction was all ones). NRT5A also sets NRF3 (D3) disabling the round net. Then when the chain again reaches NRT3.1, it automatically continues to NRT4, to which it goes on the first pass if no rounding is required. NRT4 transfers the exponent from SC to AR so the result is now complete and in proper floating-point form. This pulse also triggers NRT6 which returns to ET10 to store the result.

The rounding used here is in magnitude, i.e., away from 0—if the bottom part is $<1/2$LSB of the top part then it is ignored, but if the bottom part is $\geq 1/2$LSB then the magnitude of the top part is increased by 1LSB. The reason for conditioning the round on a 1 in MQ1 is that the action both rounds and places the result in correct 2's complement form. The answer, if negative, is a 1's complement unless the low-order part is null and a 1's complement is one greater in magnitude than a 2's complement. Adding one because of a 1 in MQ1 increases the magnitude of a positive result when the lower order MSB is significant, but decreases the magnitude of a negative result when the low-order MSB is not significant. Of course, a 0 can be null in a negative number, but only if the entire low-order part is null, in which case the high-order part is already a 2's complement. The program should always round unless the low-order half is actually going to be used.

A further caution is necessary concerning division. The divide subroutine computes 29 quotient bits and leaves the remainder in MQ9-35. Following the two right shifts, MQ0 contains the true sign of the remainder, and MQ11-35 contain a truncated remainder—the two least significant remainder bits are lost in the shifts. If there was overflow in the division after normalization, MQ still contains two quotient bits, otherwise only one. In order to use the low-order part of the result, the program must either reconstruct the true remainder or save the extra quotient bits and append them at the left end of a lower order quotient calculated from the given remainder.

CHAPTER 7

MEMORY LOGIC

In a PDP-6 system, the core memories and fast memories are separate units connected to the arithmetic processor by a memory bus. A core memory may contain an 8K or 16K bank, and a fast flip-flop memory may replace the bottom 16 locations in any core bank. The internal operation of these memories, their control functions, their timing, and the way they respond to processor requests are described in a separate manual. This chapter describes only the hardware at the processor end of the memory bus: the logic elements that request access to memory, provide the necessary addresses, and control the transmission and receipt of data.

A time pulse in a main cycle or a subroutine may request access to memory by triggering appropriate operations in the memory control section of the processor. Memory control places a request signal on the bus and the processor must wait for a response from the memory addressed by the high-order address bits. Once the memory is free and available to the processor, the time required by the processor to transmit or receive data depends upon the type of memory. For a core memory, this time is usually much shorter than the memory cycle. The fast memory contains no buffer, so a read requires only slightly less time than a write. For reading from a core memory, the processor must wait until the data is available and the memory rewrites the word automatically; for writing the processor need wait only until the memory acknowledges the request, at which time the memory stores the data in its own buffer and continues with the clear and write cycle.

In addition to the standard logic for controlling access to memory, this chapter also describes the user mode registers. These govern the protection and relocation of areas in core and are included in the description of the memory address logic.

7.1 MEMORY ADDRESS LOGIC

The memory system appears to a processor as one homogeneous unit: the processor may address any one of 262,144 locations merely by providing an 18-bit address from the memory address register. The actual address put on the bus is the sum of C(MA) and C(RLR). The bus control

portion of each memory decodes four or five address bits (depending upon whether a given memory contains a 16K or 8K bank), and a given memory responds only to the address wired in. If the bottom of a core bank is replaced by a fast memory, an additional selection signal is required; this signal is generated by the processor.

a MA Register

Figure 7-1 shows the 18-bit memory address register. Transfers of addresses into MA, which must always be preceded by a clear, may be made from the memory buffer, the program counter, or the console address switches. For the transfer of short addresses of accumulators, index registers, and PI channels, individual pulses may directly set bit 30 and bits 32-35. The transfer gates that produce the pulses are included in the MA control circuits (b below). The MA flip-flops are connected in a carry configuration so that a pulse at the MA+1 input to MA35 adds 1 to the contents of the register. The carry chain is broken, however, between the fourth and fifth bits from the right. A carry out of MA32 cannot go into MA31 if the processor is fetching or storing a second accumulator; such actions occur for double-length shift operations and certain arithmetic instructions that use a double-length operand or produce a double-length result. When such an instruction addresses location 17 as an accumulator, the second AC is in location 0. In all other situations that increment MA, the second address is in the normal order.

To address a location within a single memory bank, address bits 21-35 are supplied over the memory bus with ones asserted negative. Bits 26-35 are supplied directly by MA, bits 21-25 from the sum with the relocation register RLR. Bus drivers for the direct bits are shown above MA; those for the relocated bits, RLA, are with RLR in Figure 7-5. The high-order relocated bits, which select a memory, must be supplied in both states at ground assertion. Figure 7-2 shows the required bus drivers and all control connections to the memory bus. Bits 21 and 35 are supplied over the bus in both forms; that is, they are supplied both as bits to select a memory bank and to select a location within a bank. A switch at each 16K memory allows the operator to select between the bits for the two uses. If bit 35 replaces bit 21, the memory locations are interleaved; i.e., all odd addresses are in one bank, all even in another. With an 8K bank, addresses must be interleaved because five bits are required to select the bank.

At the upper left in Figure 7-2 are the bus connections for the memory control section of the processor. At the right is a net that decodes MA bits 18-31 for all zeros, a condition which means that the address in MA is 17 or less. The decoder output generates the fast memory selection signals for the bus since a fast memory replaces the bottom 16 locations in core (fast memory addresses are not relocated). Additional fast memories require extra decoding. The signal that selects the fast memory is also conditioned by the 0 state of the readin mode flip-flop. This mode allows the operator to start a program in the area of core that is replaced by the fast memory and is ordinarily inaccessible to the program. The ungated decoder output is used by readin mode control to determine when the program leaves the readin area.

In addition to a flip-flop and three transfer gates, each MA module contains a logic net that compares the MA bit with the corresponding console address switch (these are shown below the input gates in Figure 7-1). The outputs of these nets are ANDed (Figure 7-2, left) to assert a level when the address in MA is identical to the address in the switches. This signal is used by the memory indicator logic (7.2).

b MA Control

Figure 7-3 shows the circuits that control the transfer of addresses into MA. Every transfer requires a pair of pulses: the first triggers the clear PA through the net in the upper left; the second triggers the transfer, usually through one of the PAs at the top of the figure. Two of the transfer pulses, those for MBRT and PC are applied to the gates shown with the register. Figure 7-3 shows no pulse for the transfer from the address switches because that signal is generated by the key logic and applied directly to the register gates. The other transfer PAs in Figure 7-3 set individual MA bits through the gates in the lower right. A UUO loads address 40 into MA by setting MA 30. The transfer of a PI channel address sets MA30 and loads the channel number into MA32-34, producing the address 40 + 2n. The other sets of gates load accumulator and index register addresses into MA32-35 from the appropriate IR bits.

The address transfers are as follows: A signal from the key logic clears MA prior to any transfer in from the address switches. At the beginning of every instruction cycle, IT0 clears MA and IT1, loads PC into it to retrieve the instruction, unless the processor is in a PI cycle, in

which case IT1 loads the PI channel address. AT0 clears MA in preparation for the address cycle. If the instruction specifies an index register, AT2 transfers the address in from IR14-17 and AT3 clears it again. If the address is indirect, AT5 transfers in MBRT and the cycle returns to AT0 after the new address is retrieved. When the address calculation is complete, the processor continues to the fetch cycle, in which FT1 transfers IR9-12 into MA to fetch an accumulator and FT1A subsequently clears if addressed by either half of the accumulator; FT3 transfers in MBRT and the clear follows at FT4A. Finally, FT5 again transfers MBRT to fetch C(E) and the effective address is left in MA as the cycle ends.

MA transfers in the execute cycle are required only for particular instructions. UUO, POPJ, and BLT all clear MA at ET1. For UUO, ET3 sets MA30 so that the UUO subroutine will subsequently deposit the instruction code in location 40. For the other two instructions, MBRT goes to MA at ET3. This special transfer is required for POPJ because this instruction jumps to the location specified by the contents of the top location in the pushdown list (transfers to PC must be made via MA). BLT loads a new address into MA in preparation for the subroutine which subsequently moves C(E) to a new location. Three instructions, PUSH, PUSHJ, and JRST, clear MA and load it late in the execute cycle (ET9, ET10). The first two make the transfer so that the following store cycle will deposit C(E) or PC in the top location in the pushdown list; the third uses it to display the current instruction location in case the JRST is a halt.

At the beginning of the store cycle, the appropriate address for the deposit of C(E) is already in MA. If storage of an accumulator is also required, ST3 clears MA and ST5 loads it from IR9-12.

In the upper right of Figure 7-3 is a net that increments the address contained in MA. The PA is triggered by a signal from the key logic for the operations examine next and deposit next, by UUO T1 to switch from location 40 to 41, and by ST6 which is generated by the store cycle only for the deposit of MQ in a second accumulator. The two gated pulse inputs allow counting by FT1A to fetch a second AC, and by IT1 if the indexing of the pointer has overflowed in a block IOT that is using the PI system. Note that this latter event occurs only when performing the second instruction in the pair for a PI channel and that the channel address is transferred into MA at the same time. Since the channel address always ends in 0, MA+1 merely sets MA35.

c User Mode Registers

Figures 7-4 and 7-5 show the two registers that provide protection and relocation for the user mode. The bits in both registers are numbered 18-25 to correspond to the most significant eight bits in a memory address. Information is transferred into the registers by a DATAO for the processor, but the clear and set inputs have the prefix EX because the gating by the CPA selection level is included with the executive mode logic. The DATAO loads the protection register PR from bits 0 to 7 of the I/O bus, the relocation register RLR from IOB18-25.

The executive routine determines the size and location of the block assigned to a given user program by loading PR and RLR. Since both registers correspond to the most significant eight address bits, the numbers they represent are actually multiples of octal 2000 (all numbers in this discussion are octal), but identical contents do not represent the same multiple; the two registers have entirely different functions. An address in MA is compared with C(PR) and it is legal only if MA is less than or equal to PR. Since the remainder of MA may have anything from all zeros to all ones, if PR is clear, MA may contain any address from 0 to 1777, i.e., any address in which MA18-25 is clear. A clear PR represents a block of 2000 words; setting PR25 represents a block of 4000 words. The number in RLR, however, defines the first address in the assigned block as a multiple of 2000 since RLR is added to MA18-25. An address in a user program is legal if $C(MA) \leq C(PR) \times 2000 + 1777$, and each address must be relocated to the actual address to which memory access is made is $C(MA) + C(RLR) \times 2000$.

Every time a user program reloads MA, bits 18-25 of the address are compared with PR by the chain of majority gates at the top of Figure 7-4. Each gate receives a 1 signal from an MA bit, a 0 from the corresponding PR bit and a carry in from the previous gate in the chain. The carry output of the gate is asserted whenever at least two of the inputs are asserted (a block diagram of the gate is in the lower left). The carry mentioned here is really a borrow—the comparison is a subtraction of MA from PR and the outputs are labeled to indicate that a comparison is ∿ OK when the input conditions are fulfilled. At the beginning of the chain (upper right), the borrow in is disabled and there is a borrow out only if the MA bit is 1 and the PR bit is 0. At all other stages, the output is asserted whenever MA is 1 and PR is 0 or if there is a borrow from the previous stage and either MA is 1 or PR is 0. No actual

difference is produced and the time required for the comparison is the time taken for level transitions across the chain each time MA changes state. A borrow out at the left end indicates that MA is larger than PR; and the two outputs of the last gate, when asserted at the same polarity, indicate whether or not MA contains a legal address. The nets in the lower right AND these two levels with ∿EX INH REL so that the signals sent to memory control (7.3) can be asserted only when the memory is being protected.

At the same time that MA is compared against PR, the contents of RLR are added to MA to relocate the address to the assigned block. Every block starts at a multiple of 2000 because the 8-bit RLR is added to MA18-25. The addition is carried out by a chain of dc adders shown above RLR in Figure 7-5. The circuit, shown in block diagram at the lower left, has three inputs: a bit of MA, the corresponding bit of RLR, and the carry from the next less significant adder. It has two sets of outputs, one a bit of the sum, the other a carry to the next more significant stage. The sum circuit uses exclusive OR nets to generate a 1 output whenever an odd number of the inputs are ones; the carry circuit uses a voltage division network to generate a 1 output whenever two or more of the inputs are ones. The RLR input is ORed with an enable level (the negation of the inhibit from executive mode control) so that if no relocation is desired, the sum output is equivalent to MA and there are no carries. This enable level is applied to pin H of all adders. The carry inputs to the first adder in the chain are disabled (upper right). Each time MA changes state, the sum of MA and RLR appears at the RLA outputs after a settling interval required for level transitions across the chain.

7.2 MEMORY DATA LOGIC

Transmission and receipt of data at the processor end of the memory bus are controlled by 36 pulse amplifiers, each with AND gates at both input and output (Figure 7-6). For a write cycle, the input AND gates are enabled by the 1 states of MB bits. To transmit data pulses over the bus, memory control applies a transfer pulse to all input gates and triggers those PAs that correspond to ones in MB. To receive information from a read, memory control generates a level that enables all output AND gates; pulses that arrive over the bus are then gated through to set the appropriate MB bits (6.1).

Figure 7-7 shows 36 MI flip-flops that drive a set of console indicators to display the contents of a memory location. The flip-flops are included on the arithmetic register modules, and the MB connections to the input gates are internal to the modules. Whenever the address for a memory access is the same as that in the console address switches, the read restart or write restart from the memory subroutine clears MI and loads MB into it. The write restart includes the condition of a write request for a nonexistent memory so that MI displays the word that would have been written. The indicators also display the contents of any location that the operator examines or deposits information into. The equal address condition suffices for examine and deposit, but an additional gate is necessary for examine next and deposit next, which use MC RST1 to trigger the transfer.

7.3 MEMORY CONTROL

Figures 7-8 and 7-9 show the logic that governs requests for memory access by the processor (flow chart, Figure 4-3). The processor may request three types of memory action: read, write, and read/write—the last being available only for operations in normal mode, which we shall consider first.

Pulses from the main cycles and a few subroutines trigger the memory subroutine for the appropriate type of memory access through the gates, at the top in Figure 7-8. The reason for the access is written along with the triggering pulse. Each gate generates an output that triggers the specific type of access required; however, any specific request also triggers a general request pulse (A5), and either type that requires a read clears MB (B5). For any instruction that requires fetch and pause, FT7 triggers the read/write request, but there are three inputs to the net that generate the restart for the subsequent store (B6). Indexing of a pointer in character operations or a block IOT is performed by a subroutine which skips the store cycle and returns directly to an earlier point in the main sequence. In these special cases, the restart is supplied by the subroutine; whereas, it is supplied by ST2 for all fetch-and-pause instructions that include the store cycle.

The request pulses supply the request levels to the bus by setting up the flip-flops in the upper left of Figure 7-9. A read request sets MC RD and clears MC WR, a write request sets MC WR and clears MC RD, and the read/write request sets both. A request for any type of cycle

clears MC STOP and sets MC RQ, although the general request pulse is not applied directly to the latter flip-flop; instead it enters the two delay lines in the lower right of the figure. If the system is in executive mode, the output of the right delay triggers the PA in C7 to set MC RQ; however, if the system is in user mode, a longer delay is allowed to carry out the necessary protection and relocation operations (7.1c). Then if the user program has supplied a valid address, the request flip-flop is set; but if the address is greater than the PR maximum address, the illegal address pulse (C6) causes the processor to skip the remainder of the current main sequence by jumping directly to ST7 (5.2e) and triggers a priority interrupt on the channel assigned to the processor (8.3).

The bus levels that request a cycle and specify read or write are derived through bus drivers from the outputs of the three request flip-flops. (Note that the cycle request is generated from MC RQ only if a read or write is also specified. This is done to prevent a possible malfunction in processor memory control from generating a memory request which would not result in a response from memory.) MC RD also supplies the bus enable level that gates data pulses from the bus into MB (7.2). If the operator has pressed the MEMORY STOP key or if the ADDRESS STOP switch is on and MA now contains the selected address, the main request pulse (delayed) sets the stop flip-flop (D4).

Completion of the memory subroutine is controlled by the circuits in the lower half of Figure 7-8. When the addressed memory becomes available to the processor, it returns a pulse (lower left) that acknowledges receipt of an address. This pulse triggers MC ADDR ACK, which clears MC RQ. For write access, the acknowledging pulse also sends a write restart back to memory, transfers data from MB onto the bus, and triggers the restart pulse MC RST1 from the memory subroutine to the waiting sequence (C6). For read or the read portion of read/write, the processor must wait for the read restart from memory to trigger the MC restart. In this case, the return from memory triggers a preliminary pulse so that the subroutine return is delayed. MC RS T1 clears MC RD and supplies the subroutine return. If MC STOP has been set during the subroutine, neither the read nor the write restart from memory can trigger the MC restart, and the processor stops even though RUN is not clear. The operator may then trigger the return to the waiting sequence by pressing the MEMORY CONTINUE key, which triggers MC RS T0 at KT1.

For a read/write cycle, the response by the memory in the first part is the same as for a read cycle: MC ADDR ACK clears MC RQ and the read restart generates the completion pulses, of which MC RS T1 clears MC RD. For the second part, the write request signal alone remains on the memory bus. The read/write restart pulse supplied by the processor duplicates the action of the address acknowledgement in a write cycle by triggering the write restart (Figure 7-8, lower left). Thus the processor, which already has access to the memory, both restarts the memory and provides its own subroutine completion pulses at the same time that it transfers data over the bus from MB.

The other circuits shown in the two memory control drawings handle special situations. In the lower left of Figure 7-9 is an integrating delay Type 4303 which is placed in the 1 state by every request pulse. The delay remains continuously in the 1 state as long as triggering pulses keep arriving at intervals shorter than the delay period. But should there be no request for 100 μsec, the delay returns to the 0 state, and if MC RQ is still set at this time indicating that the last request for access to memory has not been granted, the 4303 state change triggers the nonexistent-memory pulse (the pulse is inhibited if there is a memory stop). This pulse sets an error flip-flop in the processor I/O interface to trigger a priority interrupt on the assumption that if the addressed memory does not respond within 100 μsec, there is no memory with that address connected to the bus. If the console MEMORY DISABLE switch is off, the error pulse generates a restart that clears MC RQ by simulating the address acknowledgement from memory. For write, the simulated MC ADDR ACK also triggers the subroutine completion pulse in the normal manner; for any other cycle (MC RD(1)), an additional pulse is generated to supply the MC restart.

The remaining logic is associated with a pair of synchronizing flip-flops (Figure 7-9, right) that compensate for memory stops between the two parts of a read/write cycle. If the operator has pressed the MEMORY STOP key or the ADDRESS STOP switch is on (which would allow a stop on the equal address condition), AT4 sets the split/cycle synchronizing flip-flop. There is no need to set it earlier in the main sequence because all instruction and address memory requests are for read only: a fetch-and-pause request can be made only in the fetch cycle. The 1 state of the sync flip-flop prevents FT7 from triggering the read/write request pulse and prevents the read/write restart pulse from triggering the write restart for memory. Instead,

separate read and write requests are triggered by FT7 and the read/write restart (Figure 7-8, left center) to prevent the processor from memory in the event it stops. There is an extra switch input to the sync flip-flop set gate that allows the operator to override the split cycle for maintenance purposes. In case the processor might stop while hanging onto memory, the read/write request pulse sets another flip-flop, the stop sync. If the operator should restart the processor with some operation other than memory continue, KT1 triggers the write restart to memory so that it may finish its cycle, writing the same word back into the addressed location.

CHAPTER 8

INPUT/OUTPUT

The PDP-6 input-output system includes the peripheral equipment and three sections of the arithmetic processor. The processor elements are in-out transfer control including the I/O bus, the priority interrupt system, and an I/O interface for the processor that allows IOT instructions to control the processor itself as a device. In addition to the above processor equipment, this chapter describes the control units for four of the more common in-out devices: photoelectric paper tape reader, paper tape punch, Teletype keyboard-printer, and card reader. Other equipment may be added to the system merely by connecting the associated control units to the I/O bus.

8.1 IN-OUT TRANSFER CONTROL

Figure 8-1 shows the logic that decodes the IOT instructions and times the transfer of information into and out of the computer via the I/O bus. The 36 cable drivers for the bus data lines are shown in Figure 8-2. The I/O device selection lines come from bits 3-9 of the instruction register and the bus drivers for them are shown on the IR drawing, Figure 5-7. Every device control unit contains a diode net which receives input from 7 of the 14 IOS lines, one from each bit in IR3-9. The configuration of these input connections determines the selection code for a particular device, which responds to IOT commands only when the appropriate number appears in the device code portion of an instruction word.

The cable drivers for the I/O bus are transceivers, i.e., they handle signal transmission both into and out of the computer. IOT control governs output from AR onto the bus by supplying transfer and reset levels to the drivers; any signal placed on the external bus by an input device is automatically made available through the drivers to the AR input gating, with no transfer signal required other than command levels sent to the device, followed by the driver reset. As an example, consider the driver for bit 0 shown in the upper left of Figure 8-2. A 1 in AR0 is represented by a negative level at input E (the input resistors are mounted on the AR flip-flop modules). The transfer and reset levels from IOT control are connected, respectively, to L and M. In an output sequence, L goes negative for 2.5 μsec, transferring the

input at E to outputs J, F, and H: the assertion level of the first and third is negative; the second is ground. J merely drives an indicator on the console in-out panel. F drives the external bus and is connected to the capacitor-diode inputs of the control registers and output buffers in the various devices. F and H together provide bus signals at both assertion polarities for use in I/O operations within the computer—in the priority interrupt system and in the I/O control of the processor. At the same time that the transfer level returns to ground, the reset signal brings input M to ground, placing a hard negative on the bus (F) for 2 μsec. This action discharges the capacitor-diode gates connected to the bus to ensure that there will be no conflict with a subsequent I/O operation. For input, since the outputs of the device control/status registers and input buffers are connected (through diode gates with no dc load) to the same bus lines, an addressed device merely places ones at ground assertion on the bus. With the control signals at both L and M off, a ground at F produces a negative level at H, which is connected to the AR input gating. The gate for incoming status or data is applied to the device control for 2.5 μsec and is followed by the 2-μsec reset to the bus drivers.

The levels and pulses that control the bus and devices are generated by the logic shown in Figure 8-1 (flow chart, Figure 4-13). When IR0-3 contains the code for the IOT class, the net shown at the lower left decodes IR10-12 to determine the specific instruction. The four OR gates at the right develop levels that control functions common to more than one instruction. The IOT time chain always begins at ET4 (upper left). For a block instruction, the chain starts at IOT T0, which restarts the second part of a read/write cycle to write the indexed pointer word back into memory (7.3). The return from the memory subroutine generates IOT T0A which clears MA, sets IR12 to convert the block instruction into its associated data instruction, and triggers the fetch cycle for the data. It also carries out the required skip and overflow operations depending upon whether the block transfer is using the PI system and whether the block is complete.

In any nonblock IOT, including a data instruction within a block, ET4 sets IOT GO. The 0-to-1 transition of this flip-flop triggers the initial setup delay provided the system is not still within the reset period following a previous IOT. If two IOTs occur too close together, the second waits until the fall in the first reset level itself triggers the delay. The terminating pulse from the initial setup delay, IOT T2, triggers a pair of delays: a restart delay whose terminating pulse is IOT T3, and a final setup delay whose terminating pulse is IOT T3A.

The level outputs of the initial and final setup delays are ORed to produce a 2.5-μsec gate that places information on the bus (the pulse IOT T2 keeps the gate well grounded during the initial to final setup transition). During this period, an output instruction connects AR to the bus through the I/O cable drivers; an input instruction connects the data buffer or status register to the bus in the device control. For the two output instructions, IOT pulses T2 and T3 trigger the appropriate clear and set pulses for the device. The initial setup delay allows the device selection from IR to enable the capacitor-diode gates in the device enough ahead of the clear; the restart delay allows enough time between clear and set. For any input instruction AR is cleared prior to the IOT chain by ET0; the transfer in from the bus is accomplished by IOT T3. This pulse also causes the computer to reenter the execute cycle at ET5. When all transfers are complete, IOT T3A triggers the reset delay whose level output prevents the initiation of IOT operations for 2 μsec and resets the bus.

In addition to the timing logic, Figure 8-1 also shows a pair of general control signals (upper right). Whenever computer power is turned on, -15 volts is applied to pin C of I/O cable 3, turning on all peripheral equipment. The reset signal at pin B clears the device control units on the power clear and when the operator triggers the reset from the console (both conditions are included in MR START, see 5.1). The program can also generate this reset by means of a 1 in bit 19 of the same CONO that controls the flags in the processor I/O interface.

8.2 PRIORITY INTERRUPT

The priority interrupt system allows the program to be interrupted by a signal on any one of seven channels arranged in a priority chain. The priorities assigned to the I/O devices and to the processor are completely under program control. Moreover, the program may turn any channel on or off and may request a break on any channel. The processor checks for requests at the beginning of every instruction and address cycle, and honors immediately the highest priority request that has been made. Any requests made after the processor has completed the effective address calculation must wait until the end of the current instruction except in a block transfer. The processor honors a request by entering a PI cycle and executing the instruction in location 40 + 2n where n is the channel number. No further interruptions can occur while the computer is in a PI cycle. If the break instruction is a block IOT and the block is not complete, the processor leaves the PI cycle and dismisses the break automatically.

If the break instruction is not a block IOT or the processor goes on to the second instruction in the break pair, the PI cycle terminates at the end of the non-IOT instruction, which must be a JSR to a break routine. This routine may then be interrupted at any time by a higher priority request, and the program must dismiss the channel with a JRST at the end of the routine.

The processor I/O interface and all device control units each contain a 3-bit PI assignment register: the program assigns a priority to a device by loading a channel number into its assignment register (0 means no selection). The outputs of the register are applied to a gated binary-to-octal decoder whose outputs 1 to 7 are connected to the seven PI request lines. When a device requires service, it requests a break by gating on the decoder, grounding the line corresponding to the number in the register.

a Priority Chain

The 21 flip-flops in the priority chain are shown in Figure 8-3. Each channel is controlled by a column of three flip-flops, PIOi, PIRi, and PIHi, where i is the channel number. The setting of each flip-flop represents a stage in the interruption process for a given channel in the sequence: channel on, request made, break held. Between the rows of PIO and PIR flip-flops are the logic gates through which breaks are requested by a device or the program; between the PIR and PIH rows is the priority-determining chain of level gates which is activated by the input PI ACTIVE at the left. All other signals at the left control state changes in the flip-flops by rows (for the generation of these pulses, see b below). For program control over the system, the individual channels 1 to 7 are selected by ones in bits 29-35 (IOB29-35) of CONO.

Since the control sequence for all channels is identical, let us consider channel 4. The program selects this channel by a 1 in CONO bit 32 and turns the channel on or off by setting or clearing PIO4. To request a break on channel 4, a device must ground the PI4 line, pin V of IOC3. At specific times during its operations, the processor strobes the request lines by generating PIR STB. This signal sets PIR4 if the channel is on and a device has requested a break on it. The program, however, may bypass the PIO flip-flops by selecting a channel and setting the associated PIR flip-flop directly (PIR ← IOB (1)).

Several PIR flip-flops may be set simultaneously, but the PI system honors only the highest priority request (i.e., for the lowest numbered PIR flip-flop that is set), by asserting one of the PI REQ outputs shown at the top of the figure. The selection of a request output is made by the chain of level gates between the PIR and PIH flip-flops. The 1 state of PI ACTIVE enables the highest priority stage of the chain at the left. If a break is neither requested nor held on channel 1, the first stage enables the second. Similarly, if stage 2 is enabled and an interrupt is neither requested nor held on channel 2, stage 3 is enabled, and so on to each successive stage. For example, a request is honored on channel 4 in the following manner: If the request and hold flip-flops for the first three channels are all 0, stage 4 is enabled at 2N11R. If there is currently no break held on channel 4, PIH4 is 0, satisfying AND gate 2N11R,S and producing a ground at 2N12S and 2N13R. The 1 state of PIR4 satisfies the upper AND gate asserting the request output PI REQ4, but negates input S of the lower AND gate disabling the rest of the chain to the right. After the break has been started, PI control generates PIH ←PI CH RQ, which sets PIH4. The 1 state of this flip-flop negates input 2N11S, disabling both PI REQ4 and the remainder of the chain. Furthermore, it also holds PIR4 in the 0 state, preventing any further request on the same channel while the break is in progress. These actions, however, have no effect to the left, so a request on a higher priority channel can interrupt the current break.

To dismiss a break when returning to the interrupted program, PI control generates PIH(0) ← PI OK(1), which clears the PIH flip-flop for the channel on which a break is currently being held. It does this by clearing all PIH flip-flops from the left, up to and including the first one that is 1. The enabling levels for the clear gates in the stages to the left require that both the PIH and PIR flip-flops be 0, but this condition is bound to be satisfied because setting any higher priority PIR flip-flop would have caused an interruption of the break in progress. Thus, the system always dismisses the current break which is of highest priority and returns to a lower priority break, or if there is none, to the main program.

b PI Control

Figure 8-4 shows the circuits through which the program controls the priority interrupt system, and which governs the sequence of operations in the priority chain when a device requests a break. When an IOT with device code 004 appears in the program, the AND gate in B6

enables the selection level for the PI system. In CONI, the status level from IOT control gates PI ACTIVE and the PIO flip-flops onto the I/O bus (the bus gates are shown in Figure 8-5, upper right). CONO bit 23 gates the clear pulse to clear the entire system by generating PI RESET (B4). This pulse clears PI ACTIVE, clears the PIO flip-flops directly, and generates other pulses which clear the PIR and PIH flip-flops (A1, 3). The following set pulse in CONO triggers PI CONO SET, which performs the various operations specified by bits 24-28. Bit 28 turns on the PI system by setting PI ACTIVE (C7); bit 27 turns it off. The other three control bits perform operations on channels selected by ones in bits 29-35: bits 25 and 26 turn the selected channels on and off (A8); bit 24 requests a break on the selected channels (A3).

The remaining logic in Figure 8-4 controls operations associated with the requesting, execution, and dismissal of a break (most events are listed in the flow charts for the instruction and execute cycles, Figures 4-4 and 4-5). When any device grounds a request line, the inverted level from the PIR input gating lights an indicator located at 2L20 (right of the figure). The initial time pulses in both the instruction and address cycles check for an interrupt request by generating PI SYNC (B3), which triggers PIR STB (B5), provided the processor is not in a PI cycle. The PIR strobe sets the PIR flip-flop in any on channel over which a device has requested a break. As soon as one or more PIR flip-flops are set, the priority chain asserts one and only one PI REQ level. The nets in the lower left of Figure 8-4 generate a level PI RQ if any request is honored, and encode the channel number into binary. PI RQ allows a delayed PI SYNC (5.2a) to generate IATO (A5) which places the processor in a PI cycle by setting PI CYC (C6). The processor then returns to the instruction cycle, and using the binary-encoded channel number, retrieves from memory the instruction contained in location 40 + 2n (see 7.1b).

Since the most common interrupt for an I/O device is one that merely executes a single block IOT that takes the place of a whole subroutine, let us consider this type first. If the indexing of the pointer word overflows, IOT T0A sets PI OV when the processor switches from the block to the data instruction. If the block is not complete, both a hold and a restore level are generated (upper left). In the execute cycle, ET0 holds the break by setting PIH, only to have ET1 immediately dismiss it; then ET10 ends the PI cycle by clearing PI CYC. If the block is complete (PI OV(1)), the break is not held, and after completing the data instruction, the

processor performs the instruction in location 41 + 2n (which must be a JSR to the break routine). The sequence of events is now the same as it would have been had the first instruction not been an IOT. The break is held at ET0; ET10 dismisses the PI cycle and clears PI OV. The routine must then terminate with the appropriate restore instruction to dismiss the break (B2).

When the program performs a block transfer, PIR STB is generated at BLT T4 during the processing of every word in the block. If a request is discovered, there is no IATO; the processor instead terminates the incomplete block as though it were finished (6.6a), and then returns to the instruction cycle to finish the remainder as a new BLT with initial addresses one greater than those last used. However, since a request is waiting, the instruction cycle is interrupted and the processor does not restart the block until all requests have been honored.

8.3 PROCESSOR I/O CONTROL

The interface between the arithmetic processor and its own IOT control section is primarily a system of flags and enable flip-flops that allows the processor to check its own status and permits processor actions to request sequence breaks through the priority interrupt system. Certain data transfers may also be triggered through the interface. An IOT instruction selects the processor in the same manner as any other device: when the device code 000 appears in IR3-9, the diode net at the left in Figure 8-5 generates the level CPA which gates the various command signals from IOT control. For DATAI, CPA gates the contents of the console data switches into AR (this transfer does not use the I/O bus and the gate is shown with the key logic, Figure 5-1); a processor DATAO loads the memory protection and relocation registers from the bus (7.1c). The condition commands, gated by CPA, control and sense the flip-flops shown in Figure 8-5 plus the user mode flag (5.5) and the PC change and overflow flags in the AR logic (6.2e). CPA CONO SET regulates the flip-flops according to information on the bus; CPA STATUS gates the flip-flops onto the bus. Extra bus lines are used for conditions out so that the set pulse may set or clear individual bits; some flip-flops are set only by external conditions and CPA CONO SET can only clear them. A complete list of the bus lines and the CONO actions and status bits associated with them is included in the IOT flowchart (Figure 4-13). In addition to the enable flip-flops and flags, CPA includes a PI

assignment register PIA which is loaded from IOB33-35. Various flag conditions gate the PIA decoder to request an interrupt on the PI line addressed by the register.

The first two flip-flops at the left indicate errors: the left one is set when a user program attempts to use a protected area of memory; the right one, whenever the processor requests access to memory but specifies an address which fails to elicit a response within 100 μsec (7.3). Either error flip-flop triggers an interrupt when set. The next pair of flip-flops provide a means of synchronizing computer operations to real time. A filament transformer in the power control drives a pulse generator that sets the clock flag 60 times per second. If the enable flip-flop is on, the flag causes a break every time it is set. The other two enable flip-flops allow the PC change or overflow flag in AR control to request an interrupt. There is also an overflow flag for the pushdown list: if there is a carry out of AR0 in a pushdown or pullout instruction, ET10 sets PDL OV causing a break. Both types of instructions are represented by the level that gates a switch of MB and AR at ET10. AR CRY0(1) represents overflow for pushdown, underflow for pullout.

8.4 I/O INTERFACE LOGIC

Every peripheral device attached to PDP-6 must have a control unit that acts as an interface between the device and the IOT control section of the arithmetic processor. A control unit is connected to the processor via the in-out bus, but an automatic control unit for a high-speed device may also be connected to the memory bus for direct transfers of data to and from memory. The basic control unit includes a data buffer and a control section. The data buffer contains a number of bits appropriate to the device (up to a maximum of 36) and is connected to both the device and the I/O bus data lines. The control section handles receipt and transmission of control signals for both the processor and the device.

The data and control connections from a control unit to its device depend upon the organization of the device and are described with the appropriate equipment. All control units are connected to the processor via the in-out bus, whose configuration is shown in Figure 7-10, and which includes the following lines:

1 Power On Line - This line is at -15 volts whenever computer power is on. It is connected to the remote terminal of the local/remote switch in the power control for each device and normally turns on all peripheral equipment when the computer is turned on.

1 Reset Line - This line supplies negative pulses to clear the control registers and data buffers of all equipment attached to the bus. The line is pulsed when computer power goes on or when the operator presses the I/O reset key on the processor console; the program may also generate a reset.

36 Data Lines, IOB0-35 - These lines transfer all data and control information, with ones asserted at ground. For output, the lines connect AR to the device control registers and data buffers through capacitor-diode gates; for input, the status registers and data buffers are connected to AR through Type 4657 Diode-Gate Bus Drivers (connection must be made by a 4657 or equivalent circuit that places no dc load on the bus).

7 PI Request Lines, PI1-7 - A device requests an interrupt on one of seven priority channels by placing a ground on the appropriate line.

14 In-Out Selection Lines, IOS3-9 - These seven pairs of lines are derived from the outputs of IR3-9, which contains the device code in an IOT instruction. Assertion is at ground for both ones and zeros. A device control is connected to only seven of the lines, one from each pair. The configuration of these connections determines the selection code for the device, which then responds to only those IOT commands that are accompanied by the appropriate code.

6 Command Lines - These lines provide two pairs of negative pulses for output, two negative levels for input. Both pulse pairs include a clear and a set: one pair loads conditions from the bus into the control register, the other loads the data buffer. The two input levels gate status and data onto the bus.

Every control unit includes control and status registers, although these registers usually overlap and may even be identical. That is, all control flip-flops can usually be sensed by the program as status, and all status flip-flops within the control unit can usually be governed by CONO. Some devices have additional status signals which can be sensed by the status checking instructions but cannot, in general, be affected by the program directly.

The control/status registers for all devices include the same basic elements. As an example, consider the control/status register for the paper tape reader, Figure 8-6. The basic elements are a 3-bit priority interrupt assignment register PIA and two control flip-flops, BUSY and FLAG. These elements have the same names in most control units but are distinguished by an appropriate equipment prefix. The tape reader has an additional control flip-flop that governs the data mode, i.e., whether data is to be read from the tape in binary or alphanumeric. The punch register contains the same bits; whereas the Teletype has two busy flip-flops and two flags because it is actually two devices—one input, the other output. Even in large scale in-out systems such as magnetic tape, the control connections to the processor are the same as in the tape reader, and the control and status registers differ only in complexity, not in kind. Every device has a PIA register, but it may have a command register associated with the busy flip-flop, several flags, and a number of status bits for error and other conditions. CONO can provide a maximum of 18 control bits; control information on a larger scale such as full addresses and word counts must be supplied by a DATAO. A maximum of 36 bits can be checked by a single status instruction; if more are necessary, additional device codes must be used.

Selection of a device and the timing of all transfers of data, initial conditions, and status between the processor and the peripheral control units are described in detail in 8.1 (which the reader is strongly advised to study before investigating any of the peripheral equipment). At the left in Figure 8-6 are the connections to the processor for PI requests, IOT commands, and device selection. Every device is connected to the six command lines that govern the transfer of conditions out, data in and out, and status in. The pulses provide output clear and transfer functions while IOT control gates AR onto the bus; the levels gate input onto the bus while IOT control pulses the information into AR at the processor. All devices require the condition and status commands; however, only a bidirectional device uses both types of data commands.

Although all devices are connected to the six command lines, to have any effect on a control unit, the commands must be gated in by a selection level that is generated by the net in the lower left of the figure. Every control contains such a diode net, wired to the seven IOS lines that define the device selection code. The control unit then responds to IOT commands only when the appropriate code appears in the instruction. Note that the reset bypasses the selection gate in triggering IC CLR and clears the control registers for all devices.

In the normal sequence of operation for any device, the program first provides the initial conditions by means of a CONO, whose pair of command pulses generates the IC pulses within the control unit. The first pulse usually clears all the control and status flip-flops; the second pulse loads the assigned channel number into PIA and sets up the other control bits as necessary. The control information is supplied over the in-out bus by the effective address calculated for the CONO: bits 33-35 always provide the priority assignment; the configuration for other bits depends upon the device and is listed in the flowchart of the input-output operations (Figure 4-14).

The 1 states of the status bits are separately ANDed with the status level, which is generated by any of the three conditions-in instructions. The AND gate outputs are applied, in general, to the same bus lines over which the equivalent initial conditions are sent to the control register. The outputs of individual bits in the right of the figure provide control levels for the various operations within the control unit. The outputs of the three PIA bits, however, are applied to a gated binary-to-octal decoder. Outputs 1 to 7 of this decoder are connected to the request lines of the priority interrupt system. A group level at pin P enables the decoder, placing a ground on the output corresponding to the number contained in PIA, i.e., requesting a break on the channel assigned to the device. The 0 decoder output is not connected, so it corresponds to no PI assignment.

In addition to governing information transfers, the data commands also perform certain control functions so that an entire block of data can be processed after giving the initial conditions only once. CONO must assign a PI number and provide the necessary data mode information, but it need not necessarily set BUSY. A common procedure for output is that each DATAO not only loads the buffer but also sets BUSY causing the device to operate. When a control unit has completed the transfer of data from buffer to device, its completion

signal clears BUSY and sets FLAG. The latter action enables the PIA decoder requesting an interrupt. The processor then responds with another DATAO which provides new data, clears FLAG, and again sets BUSY. Following the last word in the block, the program must provide an extra CONO to clear FLAG. For input, the initial conditions must include setting BUSY to make the control unit retrieve information from the device; the completion of the transfer from device to buffer then clears BUSY and sets FLAG for an interrupt. Then the DATAI that gates the buffer onto the bus also clears FLAG and sets BUSY causing the control unit to retrieve more data. The turnoff of the DATAI level clears the buffer in preparation for the next word. To end a block, the program must follow the last DATAI with an immediate CONO to clear BUSY, preventing the retrieval of an extra word.

8.5 STANDARD IN-OUT EQUIPMENT

Small scale in-out devices most commonly used with PDP-6 are paper tape reader and punch, Teletype keyboard-printer, and card reader. This section describes the control units for these four devices; control units for other peripheral equipment are described in separate publications. The remaining Chapter 8 logic drawings are in four groups of two or three each for the standard devices. In each group, the first drawing shows the control/status register and control connections to the processor via the I/O bus as described in 8.4. The other drawing or drawings in each group show the control connections to the device, and the data buffer with its connections to both the device and the I/O bus.

a Paper Tape Reader

In addition to PIA, BUSY, and FLAG, the control register for the tape reader (Figure 8-6) contains a data mode flip-flop B. There is also an extra status bit which indicates that the reader motor is on. Since the reader is an input device, for initial conditions CONO must provide a PI assignment, clear FLAG, set BUSY to cause the device to retrieve the first data from tape, and set up B according to the desired mode. Alphanumeric mode (B=0) reads a single 8-bit character from tape and makes it available over bits 28-35 of the bus with hole 1 in IOB35. Binary mode (B=1) reads holes 1-6 of only those characters in which hole 8 is punched and assembles six such characters into a 36-bit word. The first character encountered is made available over bits 0-5 of the bus with hole 1 on IOB5.

Figures 8-7 and 8-8 show the 36-bit buffer, which is made up of six Type 4221 6-Bit Shift Registers. A single shift register module contains one buffer bit from 0 to 5 and every sixth bit after it; output from a single hole position is always read into a single module. For example, hole 1 is associated with that part of the buffer containing bits 5, 11, 17, 23, 29, and 35 (upper right, Figure 8-7). The strobe generated by the feed hole reads hole 1 into bit 35 (the presence of a hole is indicated by a negative level), hole 2 into bit 34, etc. In alphanumeric mode, B(0) causes the strobe to load holes 8 and 7 directly into bits 28 and 29 at the same time that the other six holes are loaded into the least significant bits of the 4221 Registers. Thus in reading a single 8-hole character, the 4221 Registers are not used as shift registers at all. But in binary mode, the first strobe loads the six data holes into the least significant bits of the register, and each subsequent strobe shifts the previously read character one place to the left in parallel at the same time that it reads in a new one. The characters are counted in an extra 4221 Shift Register in the upper left of Figure 8-7. Every strobe sets the right bit SR6 and also shifts left those ones that have already been strobed in. The sixth strobe sets SR36 indicating that the entire 36-bit word has been assembled.

Retrieval of information from tape is controlled by the logic shown in the lower left of Figure 8-8. When the operator turns on the reader from the console, the level MOTOR ON generates a pulse that clears BUSY to ensure that the reader does not go into operation inadvertently. This start clear pulse also sets FLAG to cause an interrupt when the motor goes on, and the same action is also produced through a pulse generator in Figure 8-6D6 when the motor goes off. When a CONO for the reader appears in the program, the first command pulse clears the control register. The second command pulse must then set BUSY, whose 0-to-1 transition generates PTR CLR (Figure 8-8C4) to clear the buffer and the character counter SR. BUSY(1) also engages the clutch and releases the brake (lower left), placing the tape in motion. (The operator may duplicate this action by means of the console tape feed switch but no reading occurs.) The feed hole signal is connected to two pulse generators—one through an inverter—so that a leading edge pulse is generated on the transition-to-ground when the feed hole is encountered and the trailing edge generates a second pulse on the negative return. If BUSY is 1, the leading pulse triggers a 400-μsec delay whose termination generates the strobe that loads the buffer. This is timed to occur

when the holes are centered with respect to the photocells. In alphanumeric mode, every feed hole strobes all eight data holes into buffer bits 28-35. The trailing edge pulse then sets FLAG and clears BUSY: the former enables the PIA decoder, the latter releases the clutch and engages the brake, stopping the tape. (The trailing edge pulse is gated by BUSY(1) to prevent the tape feed from setting FLAG.) When the processor responds, DATAI gates the buffer onto the bus and clears FLAG. The termination of DATAI sets BUSY to restart the reader and the flip-flop transition again clears the buffer and SR by generating PTR CLR.

In binary mode, every feed hole triggers the delay but the terminating pulse generates a strobe only if hole 8 is punched. The strobe loads holes 1-6 into the least significant bits of the shift registers and sets SR6. Six strobes fill the buffer and shift a 1 into SR36, which allows the trailing edge pulse to set FLAG and clear BUSY. Retrieval of the word by the processor and restart of the reader are the same as for alphanumeric mode.

b Paper Tape Punch

The punch has the same control bits as the reader: a 3-bit PIA register, BUSY, FLAG, and a data mode flip-flop B (Figure 8-9). B determines whether punching is in alphanumeric or binary mode; unlike the reader, however, either mode punches just one 8-bit character per DATAO and the only difference between modes is in the character format. Alphanumeric mode (B=0) punches the character supplied by bits 28-35 of the bus, with IOB35 controlling hole 1. Binary mode (B=1) always punches hole 8, skips hole 7, and punches holes 6-1 according to the characters supplied by IOB30-35. Also unlike the reader, the program rather than the operator turns on the punch motor and there is a wait of 1 second after turnon to allow the motor to reach speed. To reduce wear, the motor goes off whenever the program does not call the punch for 5 seconds. Because the program must turn on the motor, the initial conditions may vary. However, in normal circumstances CONO supplies a PI assignment, sets up B according to the desired mode, and may set FLAG if the programmer wishes to handle an entire block including the first character through the PI system. Then the clear for every DATAO clears FLAG and sets BUSY to trigger the punch cycle for the character supplied by the instruction; when the cycle is complete, a done pulse clears BUSY and sets FLAG to request a priority interrupt by gating on the PIA decoder.

Figure 8-10 shows the data buffer with associated gates, punch solenoid drivers, and timing circuits that govern the motor and punching cycle. Whenever the program sets BUSY or the operator feeds tape from the console (either of which generates GO), the 4303 Integrating Delay in D6 is set and remains set as long as it is triggered at intervals shorter than the delay period. The 1 state of this delay turns on the punch motor by enabling the Type 823 SCR Driver (A1) and holds the motor on for 5 seconds after the final GO. The gate to the driver is inhibited, however, by the power clear enable from the key logic so that the punch cannot go on when computer power is turned on. The delay 1 state is also ANDed with the off level from a second delay to generate SPEED, which is in turn ANDed with BUSY(1) to generate READY. The second delay is triggered by the turnon of the first, so the assertion of the two signals by GO is delayed 1 second if the motor is off. Both signals indicate that the motor is up to speed and punching may proceed, but the busy condition in the ready signal prevents the tape feed from setting the flag and causing an interrupt.

The circuits in the upper left synchronize punching to the period of the motor and drive mechanism. A reluctance pickup provides a sync mark through a pulse generator whose output, gated by READY, triggers a 5-msec delay. The delay level output, ANDed with BUSY(1), gates the data buffer 1 outputs to enable the drivers for the punch solenoids. At the end of the punch interval, the solenoids release and the delay terminating pulse, DONE, clears BUSY and sets FLAG. Note that the input from the tape feed key operates the feed hole solenoid driver directly, provided the motor is up to speed. This allows the operator to generate tape leader: the tape feed key enables the speed level after the appropriate delay but does not generate the ready level, so it punches feedholes without punching the contents of the buffer.

The data mode flip-flop controls the character format through the gates in the lower left. In binary, B(1) provides a hard ground through the 4113 Diode Gates to hold PTP8 set and PTP7 clear regardless of the action of the DATAO clear and set pulses on the remainder of the buffer. In alphanumeric, the 4113 outputs float, and punching of holes 7 and 8 is determined solely by the contents of the corresponding buffer bits.

c Keyboard-Printer

The Teletype is actually two separate and distinct devices with a common PI channel and device code. Signal names in the logic drawings use three prefixes: TTY for elements common to both devices, TTI and TTO for signals unique to the keyboard and printer, respectively. In addition to PIA, the control register (Figure 8-11) includes two pairs of control flip-flops, TTO BUSY, TTO FLAG, TTI BUSY, and TTI FLAG. In order to allow the program to control the devices separately, the first pulse in the CONO pair clears only PIA; both clearing and setting of the other control bits are handled directly by CONO bits (of course the IOB reset does clear the control bits in addition to PIA). In regular operation, the control bits are handled by the data instructions and signals derived from flag state changes in the transmitter and receiver. For output, the DATAO clear sets TTO BUSY and clears TTO FLAG; the subsequent setting of TTO DONE in the transmitter clears TTO BUSY and sets TTO FLAG. For input, TTI BUSY is set by the 0-to-1 transition of TTI ACTIVE when the operator strikes a key. When the entire character has been received, TTI DONE is set, clearing TTI BUSY and setting TTI FLAG. The response by the program with a DATAI then clears TTI FLAG. Setting either flag gates on the PIA decoder to request an interrupt. During a prolonged idle interval, PIA should be left clear to prevent interrupts due to inadvertent keyboard manipulation. When setting up PIA following an idle period, the CONO should also clear TTI FLAG to prevent readin of a character that may have been typed accidently.

Data transmission between the processor and the Teletype control unit is in 8-bit characters over bus lines 28-35. IOB35 corresponds to the first character bit and the eighth bit (IOB28) is always 1. Between the control unit and the keyboard-printer, data transmission is in the form of 11-unit characters which are presented serially at 110 bits per second, so one complete character requires exactly 100 msec. Character transmission always begins with a start impulse (space), followed by the eight data bits in order—with ones represented by marks—and transmission is terminated by a stop impulse (2 marks). An idle line marks continuously.

At the right in Figure 8-12 are a crystal oscillator and a countdown chain that generate a pair of clocks to regulate the transmission and receipt of data. The higher speed clock for input provides greater time resolution: this allows sampling the keyboard data near the center of each bit to prevent ambiguity during level changes. Almost all of the remaining logic in the figure is included in two modules, a 4706 Teletype Receiver and a 4707 Teletype Transmitter. Each includes an 8-bit data buffer with associated control circuits and flags. Both buffers are shift registers (the output buffer is actually included within a 10-bit shift register).

TTO (Figure 8-12, upper half)

The transmitter module receives no initial reset, so the program or the operator should always send a rubout character after computer power turnon. After every transmission, the data buffer is left clear in readiness for the next character. When a DATAO triggers a transmission cycle by setting BUSY, it also clears DONE and then loads the character into the buffer and sets ENABLE. The stop timer is a 4-counter that holds at 4 until enabled by a 1 count applied to a terminal separate from the clock input. The 4 count ANDed with ENABLE(1) conditions the set gate for ACTIVE so the next clock sets it. ACTIVE(1) enables $\div 2$, and the transition clears OUT LINE, placing a start impulse (0) on the printer line. Clocks 2 and 3 then set and clear $\div 2$, whose clear transition generates the first shift pulse. Shift 1 clears ENABLE, shifts the 1 that was in ENABLE to TTO8, and moves the entire character left one unit, placing the first character bit in OUT LINE and out to the printer. The seven succeeding shifts, at clocks 5, 7, ..., 15, 17, move the rest of the character out to the printer one bit at a time; they also fill the register with a 1 followed by zeros immediately after the eighth data bit. When TTO1-8 contain 10 000 000, the AND gate in A6 is satisfied, enabling the ACTIVE clear gate. The shift at clock 19 then clears ACTIVE and shifts left again, clearing the register and setting OUT LINE to feed a stop impulse to the printer. ACTIVE(0) disables $\div 2$ to inhibit further shifts, enables the stop timer by loading in a 1 count, and sets DONE, which clears BUSY and sets FLAG. It also holds OUT LINE on to keep the line marking.

The next character cycle cannot begin until the ACTIVE set gate is enabled again. Clock 22 asserts the 4 count: if the new DATAO has already set ENABLE or does so before the next clock, ACTIVE is set at clock 23 and printing continues at maximum rate; if not, ACTIVE(1) does not occur until the clock following DATA SET. Thus the stop is held on the line for four clock periods or two character units (minimum) as required by the printer; and for maximum printing rate, DATAO should appear within this interval. The left and right sections of the output timing chart show, respectively, the flip-flop conditions for relatively late and normal DATAO arrival within the maximum rate interval. Levels shown at the left also apply to first-character or slow-speed printing except that the 4 count would be up already.

TTI (Figure 8-12, lower half)

At the beginning of an input cycle, the data buffer contains the previous character; other conditions are as shown at the left end of the input timing chart. The negative start impulse from the keyboard distributor asserts SPACE, which enables the ACTIVE set input through the AND gate in D8. ACTIVE(1), produced by clock 1, enables ÷ 8; the transition clears ÷ 8, clears LAST UNIT, sets BUSY, and generates SET, which sets all flip-flops in the buffer. LAST UNIT(0) enables the shift one-shot, so each 4 count asserted by ÷8 generates a shift pulse. The first shift, at clock 5, clears DONE and loads the start impulse into TTI8 as a 0; successive shifts read in the character bits and move the 0 left. When TTI1 contains the 0, the last shift fills the buffer and sets DONE and LAST UNIT. The DONE transition sets FLAG and clears BUSY; when the program responds, the DATAI gates the buffer onto the bus and clears both FLAG and DONE. LAST UNIT(1) inhibits the one-shot but enables the lower ACTIVE clear gate; no more shifts are generated within the cycle but the next 4 count clears ACTIVE, inhibiting ÷8 and enabling the upper input to the AND gate for the ACTIVE set input. The next character cycle begins when a start impulse satisfies the lower input to this gate.

The AND gate in D7 connected to the upper ACTIVE clear input prevents activation of the unit by momentary noise pulses at the data input. The start impulse must prevail through clock 5; otherwise ~TTI SPACE satisfies the gate (the buffer was set at clock 1) so the first shift clears ACTIVE to reset the device.

d Card Reader

In addition to PIA and FLAG, the control register for the card reader (Figure 8-13) contains START instead of the usual BUSY, a data mode flip-flop B, an error flip-flop DATA MISSED, and an auxiliary flag, FLAG ALL. All of these but START are available for status checking, and the reader provides four additional status signals. Since START is cleared as soon as the reader starts a card cycle, the busy status is the OR of START(1) and CARD CYCLE, which is derived from the CCL signal from the reader. The negation of this signal, CARD DONE, is also available separately as a status bit (27) as are the other four reader status signals, FEED CHECK, CRL, CREL, and EOF (bits 23-26). CRL indicates that the card reader is not ready for operation, CREL indicates a validity check or read check error in the reader (for a full explanation of reader conditions, refer to card reader operation, 3.2d). The EOF signal indicates that the card hopper is empty and its assertion triggers the pulse generator in B8 to set FLAG. The PG is also triggered whenever a feed check error occurs.

Both modes determined by B assemble standard 36-bit computer words from six 6-bit characters, but the type of information and the number of columns varies. Binary mode (B=1) reads all 12 bits in a single column and assembles three columns into one word. Although 12 bits are taken from the column, it is read as two 6-bit characters. The data lines are designated CRXL where X has the values 1, 2, 4, 8, A, B, and CR1L is the least significant bit. Each column read requires two strobes: the first reads the lower half (holes 4-9) with hole 9 corresponding to CR1L; the second reads the upper half (holes 12, 11, 0, 1, 2, 3) with hole 3 corresponding to CR1L. In alphanumeric (Hollerith) mode, the reader converts the 12 bits in a column to a 6-bit code which is then read by a single strobe. This mode therefore assembles six columns into a single word. Either mode, however, requires six strobes for six characters.

Since the reader is an input device, for initial conditions CONO must provide a PI assignment, set START, and set up B according to the desired mode. Usually the program begins a card in binary because the first column contains information about the card format. Setting START causes the reader to read an entire card even though the program can retrieve no more than six columns at a time. To retrieve a single column at a time, CONO sets FLAG ALL, allowing every column read to set FLAG, which otherwise is set for every six

characters. FLAG(1) requests a priority interrupt in the usual manner, and the DATAI response by the program clears FLAG but does not affect START, which is cleared when the card cycle begins. If the program does not respond quickly enough after FLAG is set, the buffer is cleared and DATA MISSED is set, informing the program that there are still six characters in search of a reader. Completion of the entire card sets FLAG even though the buffer may not contain a complete word (a card has 80 columns). The program must provide a new CONO to read the next card.

Figures 8-14 and 8-15 show the 36-bit buffer, which is made up of six Type 4221 6-Bit Shift Registers. Each shift register module contains a buffer bit from 0 to 5 and every sixth bit after it; output from a single data line is always read into a single module. For example, CR1L is associated with that part of the buffer containing bits 5, 11, 17, 23, 29, and 35 (Figure 8-14, upper right). The strobe generated for a character reads CR1L into bit 35 (a 1 is asserted negative), CR2L into bit 34, etc. The first strobe loads the six data bits into the least significant bits of the register and each subsequent strobe shifts the previously read character one place to the left in parallel at the same time that it reads in a new one. The characters are counted in an extra 4221 Shift Register in the upper left of Figure 8-14. Every strobe sets the right bit SR6 and also shifts left those ones that have already been strobed in. The sixth strobe sets SR36 indicating that the entire 36-bit word has been assembled.

Retrieval of information from a card is controlled by the logic shown in the lower left of Figure 8-15. The first CONO pulse clears the entire buffer and character counter (C4) at the same time that it clears the control register. When START is set, SCCL is asserted (D2) causing the reader to process an entire card. When the reader starts the card, CCL asserts CARD CYCLE which clears START but continues to assert the busy status. For alphanumeric mode, CBIL is negated causing the reader to translate each column from the 12-bit Hollerith code to a 6-bit character; if B is 1, however, signals from the hole positions are applied directly to the data lines. The reader indicates that data is ready by sending a column strobe pulse to the control unit, which responds by asserting a level that causes a validity check in alphanumeric mode only. If the column does not contain a valid Hollerith character, the data lines are cleared so all zeros are read; and if the operator has pressed the validity check button on the reader, it stops and must be reset manually. The CSP from the

reader also triggers the PA in D1 to generate a strobe that loads the character into the buffer and counts it in SR. The PA output also triggers a delay which asserts CBHL for 20 μsec. This signal has no effect in alphanumeric, but in binary it causes the reader to put the upper half of the column on the data lines in place of the lower half, and the terminating pulse from the delay triggers a second strobe. There are thus six strobes for six characters whether the unit reads six columns or only three. The delay terminating pulse also indicates the presence of a column in the buffer and sets FLAG if FLAG ALL is 1. If full words are being assembled, the fifth strobe sets SR30 which then allows the sixth strobe to set FLAG. When the program responds, DATAI gates the buffer onto the bus, and the turnoff of the gating level triggers the clear to prepare the buffer for the next character. Each pulse from the 20-μsec delay also triggers a 2.2-msec delay. If this terminates while FLAG is still 1, indicating that the data has not been retrieved, DATA GONE clears the buffer and sets DATA MISSED. Finally when the card is finished, CCL from the reader is negated, generating CARD DONE, which sets FLAG and again triggers the DATA GONE delay.

The logic also includes a gate that allows the program to read any column twice. If the program gives a CONO during the card cycle, the set pulse for the initial conditions, which may change the mode, also triggers a short delay whose termination triggers the same PA that is usually triggered by the column strobe. The CONO thus generates a strobe or strobe pair that rereads the column in either the same or the opposite mode. This feature is usually used in conjunction with FLAG ALL to determine what mode to read in a card, based on information contained in the first column. The program usually starts the card in binary to inspect the first column. It may then clear FLAG ALL and reread the first column as part of a full word either in binary or in alphanumeric. The program may also use this feature to reread the third column of a binary card so that the final 78 columns are read in groups of three. If a CONO given during a card cycle sets START, the next card will be read as soon as the present one is finished.

CHAPTER 9

MAINTENANCE

This chapter discusses preventive and corrective maintenance for the arithmetic processor, tape reader, punch, keyboard-printer and card reader. Since neither the processor nor its memories can operate in isolation, the information herein must be used in conjunction with maintenance information in the memory manual. Except for the exhaustive preventive maintenance procedures for the basic in-out equipment, the maintenance is discussed at the system logic level. Circuit troubleshooting and repair are described in PDP-6 Circuits; specific lubrication and adjustment procedures as well as corrective maintenance for the in-out devices are included in the following manufacturer's manuals:

Digitronics Perforated Tape Reader Model 3500

Teletype Bulletin 215B: High Speed Tape Punch Set (BRPE)

Teletype Bulletin 281B: Model 35 Send-Receive Teletypewriter Set (KSR), Vols 1 and 2

Teletype Bulletin 1187D: Parts, Model 35 Send-Receive Teletypewriter Set (KSR)

Burroughs B122 Card Reader Technical Manual (B122.51)

The first section of the chapter discusses operation of the equipment for maintenance purposes and describes those controls and indicators not used in the course of normal system operation. The second section discusses maintenance programs and includes a list of those for the equipment described in this manual, plus the fast and core memories. The third section includes preventive maintenance schedules and a description of marginal check procedures. Finally, corrective maintenance includes troubleshooting procedures and a description of the construction of a diagnostic and exercise program loops. Special tools and test equipment required for the performance of the various procedures are listed below; except for DEC equipment, suggested commercial brands are given for purposes of specification only, and do not constitute exclusive endorsement.

Multimeter	Triplett Model 630-NA; Simpson Model 260
Dual-channel oscilloscope	Tektronix 580 series, preferably with delayed sweep trigger facilities
System module extender	DEC Type 1954
System module puller	DEC Type 1960
Paper tape gauge	DEC Type 18467
1-inch width gauge	DEC Type 0001
Tape reader cleaning kit	Digitronics MS-133
Feeler gauges	Any quality set, 1-25 mils
0-10 pound spring scale	Chatillon 719-10
0-20 pound spring scale	Chatillon 719-20
Tape punch maintenance kit	Teletype Bulletin 1124B lists all special punch tools; order with discretion
Punch lubricants	Teletype KS7470 oil; Mobilgrease #2
KSR-33 lubricants	Teletype KS7471 grease (installations without punch add KS7470 oil)
Card reader lubricants	Burroughs S15821-26 and S64960-2 oils; S15821-32 and -39 greases
Lint-free cloths	Cheese cloth or equivalent
Cotton swabs	Q-tips or equivalent
Cleaning fluids	Dupont Freon TF; denatured alcohol
Test cables and probes	Low-capacity probes for the oscilloscope; alligator clips; etc.
Super Filter Kote (aerosol)	Research Products Corp., Madison, Wisconsin

Also have standard hand tools including Phillips screwdrivers and a complete set of Allen wrenches. The card reader requires a drift punch, a brass drift rod, a plastic-headed hammer, and a pair of 18-inch water-pump pliers. Standard cleansers for the bay exteriors, etc. should also be available. At installations that include magnetic tape equipment, consult the tape transport manual for additional housekeeping necessities.

9.1 OPERATION FOR MAINTENANCE

Chapter 3 discusses normal operation of the processor and basic in-out equipment, and explains the use of the controls and indicators that are regularly available to the operator. There are many additional controls, primarily for maintenance. They include controls for ac line and for marginal checking, and maintenance switches for the logic.

a Power Controls

Most power controls and all power supplies are mounted on the plenum doors at the rear of the bays; all their switches and indicators face outward, directly behind the exterior double doors. The main power control, usually a Type 835, is at the bottom of bay 4 (the left bay of the console viewed from the rear). It has a red light that is on whenever the external ac line is plugged in, a ganged pair of ac circuit breakers, and a LOCAL/OFF/REMOTE toggle switch. Switching to LOCAL turns on power; when the switch is in REMOTE, power is controlled from the console. The total time that power has been on is registered by an elapsed time indicator at the lower left on the in-out panel (Figure 3-3). In some processors the main control is a Type 829 which has an additional switch for a delayed output that is not used. This type of control also supplies a power clear enable directly to the power clear clock; so in a system using the 829 the integrating delay shown in the lower right of Figure 5-1 is not present. Turning on the main power control supplies ac to the fans and the power supplies.

Located in the upper part of the bay 3 plenum door is a secondary power control that supplies ac for the motors in the reader, punch, and Teletype. This control is usually a Type 834, otherwise Type 811; both have controls and indicators identical to those on the 835. When this secondary control is in REMOTE, it goes on whenever the power supplies are turned on by the main control. Also at the top of bay 3 is a panel containing six ac convenience outlets.

As explained in 3.2, power to the reader and Teletype motors is controlled by the operator. The ac line to the punch is controlled by a Type 823 (mounted directly on the punch motor) through which the logic turns the punch on and off. For maintenance purposes the logic may be bypassed by means of an ON/OFF toggle switch located behind the right end of the chad box. The card reader can go on only if the operator has turned on its main power switch located on the left side of the stacker. Then it turns on with system power in the same way as the 834: a −15 vdc turnon signal is supplied through the I/O bus. If system power is off, the reader may be turned on and off independently by the POWER ON and POWER OFF buttons on the reader console.

b Marginal Check Controls

At the top of the bay 4 plenum door is a Type 734 Power Supply whose floating output may be varied from 0 to 20 volts. Its controls are on the front of the console at the bottom of the in-out panel (Figure 3-3). The 3-position switch controls the polarity of the supply output; the large knob controls the output voltage amplitude, and the dc voltmeter indicates the magnitude of the output (a similar meter is mounted directly on the supply).

In every logic mounting panel (except the lower panel of a pair containing double-height modules) pins A to D are connected to the power and ground lines. To allow submodular marginal checking, there are two independent +10 volt power lines, +10A and +10B; one is bussed to all A pins in a panel, the other to all B pins. All D pins are grounded. The C pin on every module receives −15 volts, but all the C pins are bussed together only if the panel contains no pulse amplifiers. The C pins of the pulse amplifiers in a mounting panel are separately bussed to permit independent application of negative marginal check voltage to them (in the processor the pulse amplifiers are module types 1607, 1609, 4606, 6603, 6609); the remaining modules in the panel always receive the fixed −15 volts.

At the left end of each logic panel are three toggle switches. When all three switches are down, all the modules in the panel receive the normal fixed voltages. Turning the polarity switch to +10 MC and pushing up the top or middle toggle switch applies the output of the variable power supply to the +10A or +10B bus respectively on the panel. Turning the polarity switch to −15 MC and pushing up the bottom toggle switch applies the marginal check

voltage to the C pins of the pulse amplifiers in the panel. Turning the polarity switch OFF applies normal voltage to all power lines regardless of the setting of the toggle switches. While marginal checking a panel, the toggle switches need not be turned off when switching from one polarity to the other because the polarity switch applies correct fixed voltages to all unselected lines (i.e., those of opposite polarity) even if the individual panel switches are left on. However, this interlock and the center-off polarity switch position are provided only as a convenience and should not be used as a substitute for turning off the toggle switches. At the completion of marginal check procedures the technician should turn off all three marginal check toggle switches on every mounting panel.

Marginal check voltages may also be applied to the photo amplifiers in the tape reader. Two switches, one for the eight code holes, the other for the feed hole, are located below the console shelf in the upper right corner of mounting panel 4K. When both switches are in the +10V position (left), all amplifiers receive the normal fixed +10 volts. Turning the polarity switch to +10 MC and pushing the FEED HOLE switch to the right (also labeled +10 MC) applies the output of the variable supply to the amplifier for the feed hole photodiode output. Similarly the CODE HOLE switch allows application of marginal voltage to the amplifiers for the information holes.

c Maintenance Switches

Mounted on brackets between the mounting panels on the wiring side are five toggle switches that allow the technician to alter the normal operation of the processor as an aid in troubleshooting the logic. All of these switches are off when down. One switch was mentioned in Chapter 3 because it must be used when depositing a readin loader in the bottom of core. The RIM MAINT switch is mounted just above 2M1,2, and it is shown in the lower left of Figure 5-2. Turning the switch on grounds the 0 output of the RIM SBR flip-flop so that it cannot be cleared no matter what location is used for instruction retrieval. The processor thus stays in the readin mode indefinitely, and a memory checkerboard may be run that includes the readin area. Also shown on Figure 5-2 is the REPEAT BYPASS switch which is mounted above 1M22, 23. While this switch is on, KT0A triggers the repeat delay, so the key cycle can be repeated even when the chain does not include KT4.

Two of the switches provide interruptions in the normal flow of events in the arithmetic and shift-count subroutines. Shown in the upper right of Figure 6-9 and mounted above 1H18, 19 is ART3 MAINT, which prevents the carry completion in an AR subroutine from generating the return pulse ART3; the return is instead simulated by KT2. A technician may single step through a shift-count by using SCT MAINT, which is mounted above 2C7,8 and shown at the left in Figure 6-16. With this switch on, neither SCT0 nor SCT1 can trigger the next step in the subroutine; instead each step is triggered by KT2.

The last switch may be used for troubleshooting the memory from the processor. This is SPLIT CYCLE OVERRIDE, shown at the right in Figure 7-9 and mounted above 1N15, 16. Ordinarily if the MEMORY STOP key or the ADDRESS STOP switch is on, AT4 sets the split cycle sync flip-flop. Then if the fetch cycle should generate a fetch-and-pause request, the memory subroutine makes only a read request to memory; so a processor that is undergoing trouble-shooting procedures does not hold the memory during a stop. But in troubleshooting the memory it may be desirable to stop it between the read and write parts of a cycle. The override switch accomplishes this objective by disabling the set gate to the flip-flop.

Other than the power and marginal check controls mentioned in a and b above, there are no special maintenance switches for the reader, punch, or Teletype. The card reader, however, has switches inside the hopper and stacker that generate NOT READY when the former is empty or the latter full, and these may be operated manually for maintenance. Under the reader cover is a local/remote toggle switch, LOCAL RUN. Raising this switch (local) causes the reader to feed cards continuously until the hopper is empty, and then stop with FEED CHECK on.

Besides the switches there are many maintenance indicators. This category includes at the top of bay 1 most of the indicators which display the states of all SBRs and many control flip-flops and levels. The PI REQUEST lights on the console indicate those channels on which requests have been synchronized, but the signals arriving at the logic over the I/O bus are displayed by a set of lights mounted in front of module connector 2L20. The top light is for the PI1 line.

d Single Step Operation

By using the STOP and CONTINUE keys for instructions, the operator may single step a program from one instruction to the next. Similarly, for maintenance the technician may single step from one memory call to the next by using the STOP and CONTINUE keys for memory. In either case the STOP key should be latched on, and each restart may be done manually by pressing the CONTINUE key; or the single stepping can be performed automatically by latching on the CONTINUE key, turning on the REPEAT switch, and setting the SPEED controls to the desired time interval. Each time the processor stops following a memory call, the MEM STOP light is on, and at the top of bay 1 a light must be on for the SBR that is awaiting the subroutine return. The meaning of the information displayed by the indicators on the console and the bay indicator panels can be determined by following the flow charts as the processor proceeds. Using REPEAT, the function associated with any initiating key can be repeated; all require that KT4 set the repeat delay, and the required switch settings are discussed at the end of Chapter 5.

By using the maintenance switches discussed in c above, the processor can be operated in even finer steps. Turning on ART3 MAINT inhibits the return pulse from the AR subroutines; so the processor stops every time an addition, subtraction, index, or negation is performed. The immediate results of the subroutine can then be seen by observing the lights for the arithmetic registers on bay 2, and the point at which the stop occurred is indicated by one of the SBR lights on bay 1. To observe the operation of the shift counter and the effect of shift pulses applied to AR and MQ, turn on SCT MAINT. The processor then stops prior to the shift-count subroutine and also following every step in it. After either an ART3 or an SCT stop, the processor should be restarted by pressing MEMORY CONTINUE. The attendant key cycle supplies the necessary pulse for simulating the subroutine pulse, but triggers no other operations because there is no memory stop.

To produce an automatic restart after every stop, turn on REPEAT BYPASS and latch on MEMORY CONTINUE; this causes every KT0A to retrigger the repeat delay, and the level from the key causes the delay termination to retrigger the key cycle. The ART3 and SCT switches can be left on together, so the processor stops after every AR subroutine and every step in a shift-count; if MEMORY STOP is left on too, the processor will also stop after every memory subroutine. MEMORY CONTINUE supplies the restart for all three types of stop.

While the processor is single stepping, the technician should follow the operations in the flow charts and compare the information given by them with the state of the processor as displayed in the lights. The point at which the processor has stopped is indicated by the lights on bay 1. The indicators display a great deal more information about the workings of the processor at stops within an instruction than they can provide when the processor is stopped only at the end of an instruction. For example, the indirect light I is always off at the completion of an instruction, but when single stepping through an address cycle with AR and memory stops, the light may sometimes be on. For particulars always refer to the flow charts.

9.2 MAINTENANCE PROGRAMS

MAINDEC (maintenance DEC) programs permit self-testing of the PDP-6 for check-out, preventive maintenance, or diagnosing equipment malfunctions. Each MAINDEC package consists of program tapes and a reference manual. All manuals have the same format: an abstract; a section containing operator information; another suggesting applications of the programs; and a third describing the programs.

The first section is all that the operator need use when running a MAINDEC. It lists the required tapes, specifies the usage of the console switches, and gives detailed instructions for loading and starting the programs. The final part specifies how errors are indicated to the operator (e.g., programmed halts or typeouts), provides information as to the cause of the error, and tells the operator how to repeat or restart the program. The second section suggests various applications including procedures for using the MAINDEC with marginal checking. It specifies the panels and margin levels that apply to each program. The third section contains detailed program descriptions including octal and symbolic listings; flow charts are included where appropriate. Each description explains what the program is testing and in what manner, and furnishes information to aid in updating or modifying the program.

The following MAINDECs are applicable to the basic hardware (consult the DEC program library for the current list of all MAINDECs for PDP-6).

Test	MAINDEC
Instruction Test	
Part 1	601-1
Part 2	601-2
Part 3	601-3
Part 4	601-4
Part 5	601-5
Micro Checkerboard	602
Memory Address Test	603
Clock Test	604
Memory Speed Test	605
Memory Retention after Power Failure	606
Memory Overlap	607
Power Failure Test	608
Reader Binary Test	610
Reader Alpha Test	611
Punch Test	612
Memory Data Test	613
Teleprinter Test	614
Memory Checkerboard	
Low 64 x 4K	662-1
Hi 4 x 4K	622-2
16 x 16K	622-3
16 x 16K Interleave	622-4
Protect and Relocate Test	623
Card Reader Test	641
Fast Memory Test	662

9.3 PREVENTIVE MAINTENANCE

This section discusses preventive maintenance schedules and use of marginal check during PM procedures, and lists recommended procedures for the arithmetic processor and the basic in-out equipment. Preventive maintenance consists of tasks performed prior to initial operation of the system and periodically during its operating life to ensure that it is in satisfactory operating

condition. Faithful performance of these tasks forestalls possible future failure by discovering progressive deterioration and correcting minor damage at an early stage. The tasks consists of mechanical checks, including cleaning and inspection; marginal checks, which aggravate borderline circuit conditions or intermittent failures to make them easy to detect; and checks of specific elements such as power supplies, in-out interface circuits, drivers, and sense elements.

a Schedules and Margins

Preventive maintenance procedures should be performed according to strict schedules. Recommended intervals for PM checks are based on elapsed operating time as well as calendar schedules. For convenience in scheduling normal operations, and to minimize malfunctions, large-scale PM procedures that occur at long intervals should be staggered. Each user should set up a schedule for his entire installation that distributes the total PM task evenly over the longest interval recommended. In every operating shift, a specific period should be assigned to performance of the scheduled PM program for that shift. Without an appropriately designed program of this type the PM function would necessitate system unavailability for two or three days every month.

Marginal checking utilizes the MAINDEC diagnostic programs to test the functional capabilities of the system with module operating voltages biased within specified margins above or below nominal levels. Failures brought about by over- or under-biasing are detected by the program, which provides a printout or other visual indication helpful in locating the source of the malfunction. Marginal components can then be replaced during scheduled preventive maintenance. After such preventive replacement, or when no marginal components exist, operating voltages are then biased beyond the specified margins, and the margins at which circuits fail are recorded in the maintenance log. By plotting bias voltages obtained during each scheduled PM, progressive deterioration is easily observable and expected failure dates are predictable. Accurate information acquired over a long period provides a basis for planning future preventive replacement. These plots are also useful in locating marginal or intermittent components such as deteriorating transistors.

Raising the operating voltage above +10 volts increases the transistor cutoff bias that must be overcome by the previous driving transistor; therefore low-gain transistors fail. Lowering the voltage below +10 volts reduces transistor cutoff bias and noise rejection, and thus provides a test for high-leakage transistors and simulates high temperature conditions (to check for thermal runaway). Raising or lowering the −15 volt supply (this is done only to pulse amplifiers) increases or decreases the output pulse amplitude. Each panel has a separate negative bus to pulse amplifiers; the line to pin C for logic level modules comes directly from the power supply, whereas the line to pulse amplifiers comes through the marginal check switch. The +10 margins are normally +5 and +15 volts; the −15 margins are −7 and −8 volts. The −15 volt supply must never be made more negative than −18 volts or damage to the logic can result.

For every system in the PDP-6 the user should keep preventive maintenance voltage charts on which are plotted the failure margins for each major section of logic while running the appropriate MAINDEC. DEC supplies a standard form for such plots; each page has three pairs of charts for +10A, +10B, and −15 marginal checks. The ordinate is the margin level as read from the meter on the supply; the abscissa is the time (nine spaces are provided below each chart for the date). At each check-out, two points are plotted: the voltage level which is too high for normal operation, and that which is too low. Thus a chart shows the change in time of the voltage region over which a particular section of the logic operates properly. When equipment malfunction is induced by abnormal margin levels (e.g., the region being too narrow), troubleshooting procedures (9.4) should be performed to diagnose the abnormality. At the top of each chart, spaces are provided for entering the page number in the maintenance log where such dysfunction is explained.

b Arithmetic Processor PM

The procedures in this section apply only to the arithmetic processor; procedures for the in-out equipment are given in c below.

Daily Operator Maintenance

1. Run all five parts of the Instruction Test (MAINDEC 601) without margins. Log all error halts, noting the cause if known.

2. Check that all cooling fans of the system are running and that cooling air flows freely through the filters.

3. Replace any noncritical components such as indicators, fuses, etc; note any replacement in the log.

The remaining procedures are to be performed by trained personnel only.

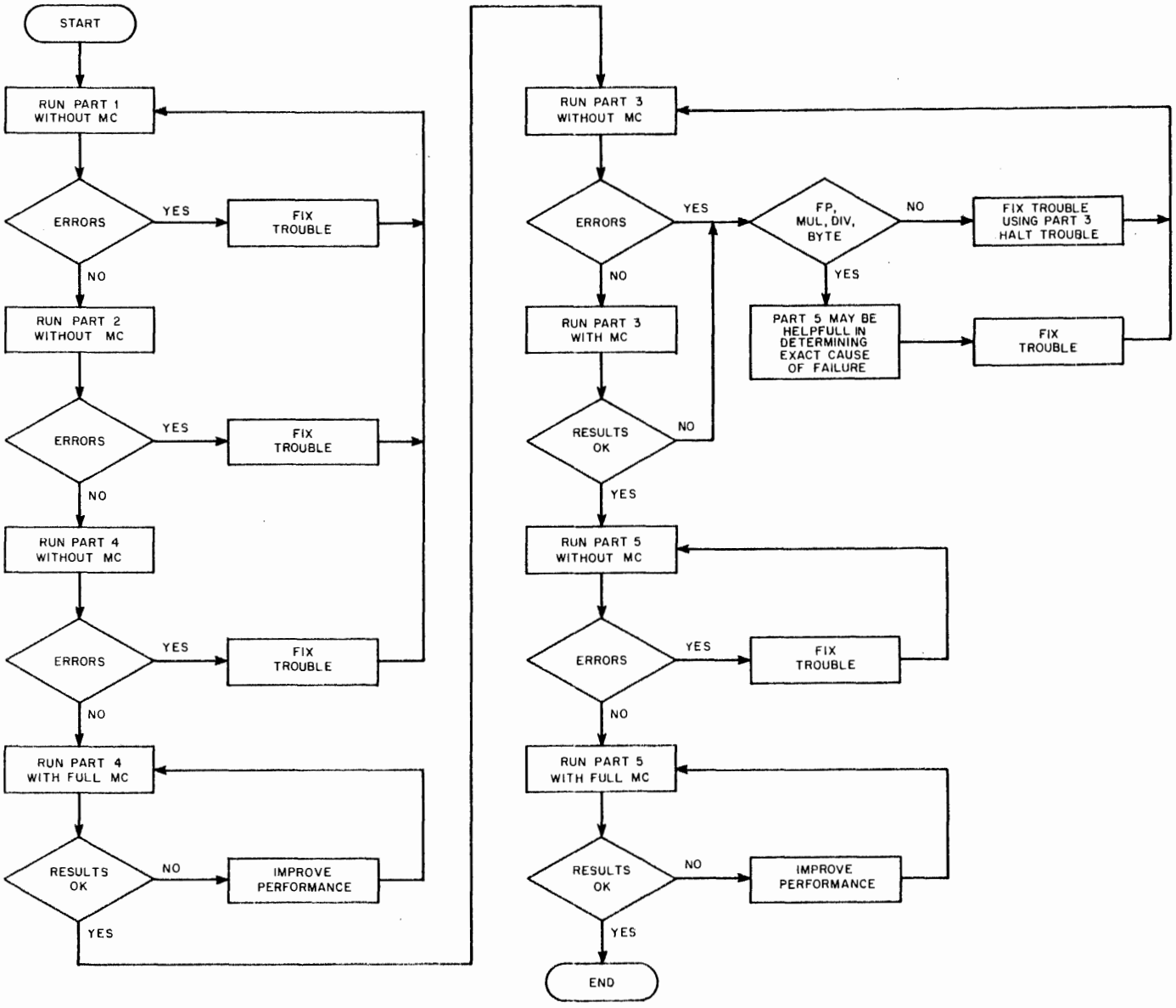


Figure 9-1 Processor Marginal Check Flow

Weekly

1. Check the operator log; note any malfunctions which have occurred in the machine, and take corrective measures if necessary.

2. If trouble noted in the log cannot be reproduced by normal trouble-shooting methods, contact the operator who made the note; using his program, try to reproduce the trouble. If the system can be kept operational by use of moderate margin voltages on certain small sections of logic, do so. Do not fail to make an appropriate entry in the log describing any provisional measures adopted, and notify the scheduling authority that diagnostic down time will be required to rectify the trouble.

Every 1000 Hours

1. Run all five parts of the Instruction Test (MAINDEC 601) with margins, using the flow diagram (Figure 9-1) and plotting the results on the preventive maintenance voltage charts. In case of failure at abnormal margin levels, note all circumstances in the maintenance log and cross reference the log to the chart by page number.

2. Change and clean the air filters at the bottom of every bay in the entire system using the following procedure:

   a. Loosen the two thumb screws holding the fan and filter housing to the floor of the bay.

   b. Remove the housing. Take the filter out of the housing and install a clean one.

   c. Replace the housing containing the clean filter and tighten the two thumb screws.

d. Clean the dirty filter by flushing it thoroughly with hot tap water in a direction opposite to that of air flow. When all dust and lint is removed shake out excess moisture.

e. Stand the filter on one end for 10 to 15 minutes to allow remaining moisture to evaporate. If the flush water is sufficiently hot, the filter should dry completely in about 15 minutes.

f. Spray the filter with aerosol Super Filter Kote or an equivalent product. The spray serves both as a dirt capturing medium and as a detergent which helps wash out trapped dust and lint during the next reverse flushing.

c In-Out Equipment PM

This section includes preventive maintenance procedures for the tape reader, punch, keyboard-printer, card reader, and DEC logic associated with them. Procedures for all four devices are grouped together under the recommended PM intervals.

The paper tape equipment requires accurately punched tape. A DEC standard paper tape gauge is supplied with the system; use it for all paper tape measurements. A paper tape with the correct specifications is shown on the next page.

The most important dimension is the lateral distance from feedhole to inner edge: it must be .392 ± .002 inch. When examining the tape on the gauge with the naked eye, this dimension must be exact—the smallest visible error will exceed the tolerance. The reader has a wide tolerance to variations in longitudinal hole spacing, but mechanical readers in general do not. To ensure that tape punched by the processor will be readable on mechanical readers such as the Teletype Model 35 or the Flexowriter, longitudinal hole spacing must be within ±5 mils accumulated error in 6 inches. The 5 mil tolerance is about one eighth the diameter of the feedhole. The output of every punch in the system should be checked once every day to ensure that the base dimensions are within tolerance. It is much easier to punch the tape accurately than to try to jockey the reader adjustments to read a wide tolerance of tape dimensions.

The punch can be adjusted to produce accurate lateral spacing on any given tape whose width is within 7 mils of the nominal 1 inch. Once it is so adjusted, any tape of 1 ± .007 inches will punch properly. If possible, make adjustments while punching the narrowest tape (.993 inch).

The following procedures are recommended for the in-out equipment.

Daily Operator Maintenance

Tape Reader –

1. Using Digitronics cleaning kit MS-133, clean the read head, tape guides, roller bearings, capstan, and brake.

2. Run the reader tests (MAINDECs 610, 611) without margins. During the tests, check that the tape is laterally positioned to ride along the inner edge of the tape guide; the photoelectric read head is designed to accept tape in this position. Log all error halts, noting the cause if known.

3. Remove the exciter lamp cover and check that the lens is clear and the photodiodes clean; replace the bulb if there is any sign of yellowing.

4. Check that the lamp cover springs are adjusted so as to just touch the top surface of the tape.

Tape Punch –

1. Run the punch test (MAINDEC 612) without margins. Log all error halts, noting the cause if known.

2. Using the DEC standard tape gauge, check that the punch output conforms to the following standards (see Figure 9-2).

   a. Feed hole to inside edge: .392 ± .002 inch.

b. ±5 mils accumulated longitudinal error in 6 inches.

c. Width: 1 ± .007 inch.

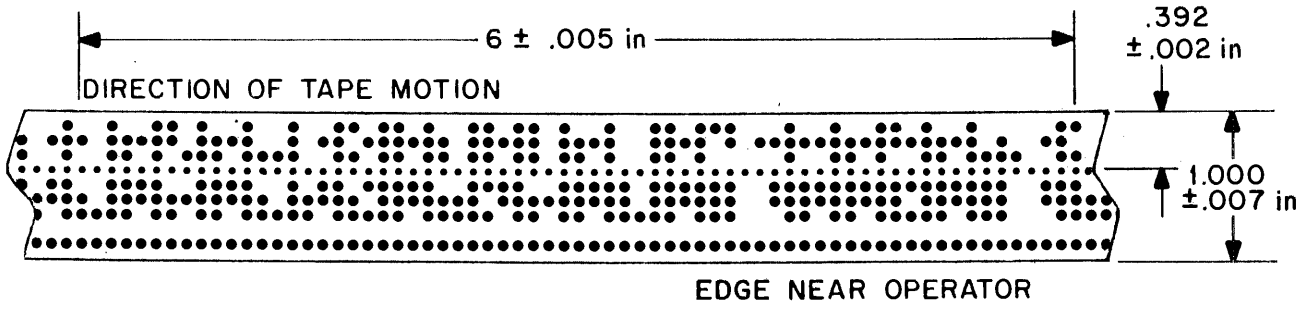


Figure 9-2 Paper Tape Dimensions

3. Check for "fuzzy" holes. This indicates a misadjusted or worn out die block.

4. Empty the chad box (the chad box should also be emptied every time a new box of tape is installed).

5. Clean the punch by blowing dust off the die block. Do not use alcohol or other solvents near the feed pawl or die block since such solvents remove the light lubricating film.

Teleprinter –

1. Inspect and clean the platen and paper guides as necessary (in general, the platen needs cleaning only if typing has run off the page or the printer has been run without paper).

2. Remove lint and other fouling material from the ribbon guides and replace the ribbon if necessary.

3. Run the Teleprinter Test (MAINDEC 614) without margins. Log all error halts, noting the cause if known.

Card Reader –

1. Remove the exciter lamp assembly. Clean the solar cells, the feed head, and the hopper area with a soft bristle brush.

CAUTION

Use only Freon as a cleaning agent for the solar cells. Other cleaning fluids may damage the potting in the assembly.

2. Run the Card Reader Test (MAINDEC 641) without margins. Watch for unusual wear on the test deck. Log all malfunctions with probable cause if known.

Every 160 Hours (Monthly)

Tape Punch –

1. Remove the punch from the console and perform the complete lubrication procedure outlined in 5 of Teletype Bulletin 215B. Be sure that no oil or grease accumulates between the armatures and magnet pole faces or between contact points.

2. Inspect the general condition of all moving parts and tightness of wiring connections. Make sure nuts and screws that lock the adjustments are tight, but do not apply sufficient torque to disturb the adjustment. See that all contact points meet squarely. Rotate the main shaft slowly in the normal direction (clockwise as viewed from the front) and activate all movable elements. Check for freedom of movement.

3. Run the Punch Test (MAINDEC 612) with margins. Plot the failure points on the PM voltage chart; log abnormal failure margins and cross reference the log to the chart by page number.

Every 330 Hours

Tape Reader –

1. Carefully perform the daily PM paying particular attention to the position of the tape as it proceeds through the reader, and making sure that it travels against the inner edge of the tape guides. Use an accurately punched paper tape, i.e., one which is well within the dimensional tolerances in Figure 9-2.

2. Carefully check the alignment of read head photodiodes with the holes in the accurate tape (with lamp cover still removed). Turn on the reader and thread the tape through the guides but not through the clutch or brake. Using a line with all holes punched, center the feedhole directly over the feedhole diode. The diodes for holes 1 and 8 must lie directly under the centers of those holes. If this alignment is incorrect or if skew is apparent, correct it by shimming either the read head or the tape guides.

3. Check the exciter bulb voltage: 8.5 ± .1 volts.

4. Check the pinch roller to capstan gap clearance with clutch de-energized. The proper clearance is 10 to 12 mils; if the clearance is insufficient, dirt may be limiting the full travel of the clutch solenoid.

5. With no tape in the reader, turn it on and lift the READER FEED key to engage the clutch as if reading tape. The correct clutch setting is that which allows the pinch roller just barely to engage the highest point on the rotating capstan, causing a slight jittering sensation in the finger when held against the roller. If the roller does not contact the capstan, or runs continuously with it, adjust the clearance according to Section 4-13 of the Digitronics manual. Severe jitter indicates excessive capstan runout, which should be tested as follows:

a. Turn the reader off, pull it from the console on its extension slides, and remove the drive belt from both capstan and motor pulley.

b. Turn on the reader and lift READER FEED to engage the clutch. Using 1- and 2-mil feeler gauges check the clearance between capstan and pinch roller for various angular positions of the capstan, turning it by hand. If runout is excessive, the capstan must be replaced.

c. Turn the reader off and replace the belt.

6. Using a spring tension scale, check that the force necessary to unseat the activated pinch roller is from 3 to 4 pounds. If adjustment is necessary use the procedure in Section 4-15 step 2 of the Digitronics manual.

7. Check for ridges on the capstan around the area where the edges of the tape ride. Such ridges can cause tape skew; they may be removed by applying a fine file to the capstan while the reader is running. Do not allow rubber dust to collect on the clutch block.

8. Check brake tension using a short piece of paper tape with a loop in one end. Insert the free end of the tape between the brake solenoid and its armature; use a spring scale hooked into the loop to measure the tension required to move the tape with the brake engaged. The force must be from 3-1/2 to 7 pounds.

Teleprinter –

1. Lift the cover and check everywhere for effects of vibration: loose nuts, screws, retaining clips, etc.

2. Check the selector magnet coils for signs of overheating. Make sure there is no lubricant or other fouling material under the armature. If necessary,

insert a piece of bond paper between the pole and armature to soak up any lubricant. Make sure no lint is left.

Card Reader – Sections and figures referenced below are in Burroughs Card Reader Technical Manual B112.51.

1. Check exciter lamp brilliancy according to 3.16; adjust if necessary. The procedure requires a deck of cards punched with a repetitive pattern of all ones in twelve columns followed by all zeros in the next five. This step and step 2 must be performed whenever photo amplifiers are changed in the reader.

2. Using the same test deck, check the leading edge timing of the strobe pulse with respect to the read pulse for all twelve rows (Section 3.17).

3. Check operation of the strobe-8 counter according to Section 3.18 (performance can be checked only by adjusting the counter).

4. Check response of card-detect solar cell CD1 according to Section 3.19 (the procedure requires a deck of cards having a 4 punch in column 80).

5. Check performance of the translator for valid Hollerith codes (Section 2.3), using a deck prepared by punching each valid code in alternate columns of a card.

6. Test performance of the validity check circuits (Section 2.4), using a 50-card deck in which column 40 of each card contains one of the invalid punches (all invalid combinations are listed at the bottom of Table A4-3).

7. Using Burroughs S64960-2 oil, lubricate the feed-slide/gib bearing surfaces (Figure 3.4-1, right), the feed-knife pivot (Figure 3.4-1, C), and the feed-slide fork/actuator-arm pivot post bearing surfaces (Figure 4.5-1, top right).

8. Run the Card Reader Test (MAINDEC 641) with margins. Plot the failure points on the PM voltage chart; log abnormal margins and cross reference the log to the chart by page number.

Every 1000 Hours

Tape Reader –

1. Perform the daily and 330-hour PM schedules. Tape must travel along the inner edge of the tape guides.

2. Check all power supply voltages including the exciter lamp voltage.

3. Using the same length of accurately punched test tape with alternate lines of ones and zeros, run the Reader Tests (MAINDECs 610, 611) with margins applied to the read amplifiers and reader control logic. Run the reader at top speed (400 characters per second). Make no adjustments unless the tape being read has lateral dimensions well within the tolerances in Figure 9-2, and travels against the inner edge of the tape guides.

4. Observe the feedhole signal at pin 18M in the panel containing the reader logic (beneath the processor console) with one vertical input of a dual-channel oscilloscope. Adjust the feedhole output for a width of 1 msec.

5. Apply a +9 volt margin to the code hole amplifiers, and adjust the read amplifier cards while observing both the code hole and feedhole signals with the scope. The trailing edge of the feedhole should lead the trailing edge of the code holes by 300 to 500 μsec; code hole width should be approximately 1.8 msec when it satisfies this trailing edge timing requirement. A read amplifier that exhibits extreme changes in gain with moderate rotation of the adjustment potentiometer should be replaced with a less sensitive card.

6. Using MAINDEC 610, run the test tape at all speeds up to 400 characters per second with code hole margins from 3 to 17 volts and feedhole margins

from 5 to 15 volts. Amplifiers should be touched up as necessary to obtain these margins. Error halts occurring with a splice on the test tape between clutch and brake can usually be disregarded since these are generally caused by poorly made splices. Plot the margin levels on the PM voltage chart; if adjustments are required, log the prior and post adjustment margin levels and cross reference the log to the chart by page number.

Tape Punch –

1. Check the Teletype Maintenance Kit for parts which may require periodic replacement.

2. Run the Punch Test (MAINDEC 612) with margins. Plot the failure points on the PM voltage chart; log abnormal failure margins and cross reference the log to the chart by page number.

Teleprinter –

1. Clean the distributor with Freon and a cotton swab.

2. Perform the complete teleprinter lubrication procedure as prescribed by Teletype Bulletin 281B, Volume 1 (there are four applicable sections; see the Bulletin table of contents).

3. Run the teleprinter test (MAINDEC 614) with margins. Plot the failure points on the PM voltage chart; log abnormal failure margins and cross reference the log to the chart by page number.

Every 2000 Hours

Tape Reader –

1. Check all electrical connections for electrical and mechanical security.

2. Inspect all moving parts for wear.

Card Reader – Sections and figures referenced below are in Burroughs Card Reader Technical Manual B122.51; part numbers specify Burroughs lubricants.

1. Check the following adjustments:

   a. Top and bottom feed roller clearance and feed-roller pivot arm spring tensions (Section 3.1).

   b. Exciter lamp bracket to read-bed plate clearance; light spot registration; side plate to insulation block clearance (Section 3.2).

   c. Top and bottom hopper throat blade clearance (Section 3.3).

   d. Feed-knife to card-column-80 clearance (Section 3.4).

   e. Clutch collar to actuated trip-arm clearance; actuated trip-arm to rear solenoid core clearance; disengaged trip-arm/clutch-collar latching ridge overlap (Section 3.5).

   f. Gap between card deck and front card guide, and slide to left gib clearance (Section 3.6).

   g. Stacker-stop-actuator arm position (Section 3.7).

   h. Reluctance-pickup strobe output pulse duration; pickup tip protrusion and orientation (Section 3.8).

   i. Clutch-reset and feed-test circuit breaker timing (Section 3.9).

2. With S64960-2 oil, lubricate the feed-roll arm pivots (Figures 4.3-1,2), the circuit breaker contact-arm pivots (Figure 4.4-1), the clutch trip-arm pivot (Figure 3.5-1), and the stacker-switch actuator shaft (Figure 3.7-1).

3. Using S15821-39 grease, lubricate the feed roll shaft gears (Figure 4.3-3), the feed cams (Figure 4.5-1), and the circuit breaker cams (Figure 4.4-1).

4. Using S15821-32 grease, repack the clutch housing and spring (Section 4.5-1).

5. Using S15821-26 oil, lubricate the drive motor bearings.

6. Check the following bearings for wear as evidenced by excessive bearing noise under rotation; replace worn bearings using the procedures of Section 4.

a. Feed-roller shaft bearings

b. Cam-follower roll bearings

c. Pulley bearings (at the clutch assembly)

d. Clutch shaft bearings

e. Card-feed actuator-arm shaft bearings

9.4 CORRECTIVE MAINTENANCE

PDP-6 is constructed of highly reliable, solid state modules, and the possibility of failure during the first year after installation is exceedingly remote. The maintenance staff should use this period to acquire total familiarity with the system and all its documentation — approaching the system in ignorance guarantees frustration. This section contains no fault classification system with specific remedies; instead it outlines a broad plan of attack for isolating malfunctioning hardware. When diagnosis is complete, proper remedial action is manifest. Diagnosis depends on logical thinking, common sense, secure knowledge of the system, and an organized step-by-step procedure.

Do not attempt to memorize the prints; instead concentrate on associating the drawing number mnemonic with the logic section portrayed. When system logic is well understood, the logic drawings and flow charts are generally sufficient reference documentation during troubleshooting. Logic drawings are cross-referenced to the manual by section number (it appears with the figure number in the lower right) allowing recourse to the text in cases of extreme difficulty. In addition to the DEC prints and manuals, there are manufacturer's manuals for the in-out devices. Familiarity with the contents of these manuals avoids time wasted in futile searches for information.

When confronting a malfunction in the machine, the following plan of attack should be employed:

1. Initial investigation: gather all available information on the problem.

2. Preliminary check: see if the malfunction presents obvious physical symptoms.

3. Console troubleshooting: use pertinent diagnostic programs, maintenance controls, marginal check procedures, etc., to localize the problem within a particular section of the logic.

4. Logic troubleshooting: complete the diagnosis by isolating the malfunction within a particular module, supply, or control.

5. Remedial action.

6. Validation of remedy: ensure that proper system function is restored.

7. Log report: make a complete record of diagnosis, remedy, and validation.

Troubleshooting, which is discussed in a below, includes the first four steps above. The diagnosis is expected to suggest a remedy (step 5), but the remedy may require attendant procedures; such situations are treated in b. The final two steps are discussed in c.

a Troubleshooting

This section details the diagnostic procedure to explain how faults may be isolated to individual modules; troubleshooting within modules is discussed in PDP-6 Circuits.

Initial Investigation

Before commencing troubleshooting procedures, explore every possible source of information. Ascertain all possible information concerning any unusual function of the machine prior to the fault and all symptoms evident when the fault occurred, such as the type of program in progress, conditions reflected by console indicators, etc. Search the maintenance log to determine whether the present fault or a similar one has occurred before or whether any cyclic history of

related malfunction exists, to determine how similar past conditions were corrected. Examine the PM voltage charts for any steadily narrowing operating voltage regions under marginal check. If a deteriorating module produces failures seemingly related to the present trouble, diagnosis may be simplified. The more information the technician can gather, the more rapidly he can make a diagnosis. Attempting to troubleshoot the system without first exploring every available source of information usually just wastes time.

Preliminary Check

Troubleshooting begins with a check for physical symptoms of malfunction. If a very large portion of system logic is inoperative, inspect the physical and electrical security of power sources, cables, connectors, etc. Be sure that power supplies provide proper voltages. Check the condition of all filters for free flow of air; a clogged filter may allow the temperature within a bay to rise sufficiently to cause marginal semiconductors to fail. This preliminary check is useful more often for catastrophic malfunctions than for intermittent ones. Except for cable and module connections, most intermittent failures are due to cold-soldered joints or faulty circuit components. Unless the malfunction is isolated with certainty to within two or three modules by the initial investigation, it is poor strategy to start checking arbitrary modules; more sophisticated troubleshooting procedures must be used. Nevertheless, the preliminary check often discloses troubles which would otherwise require considerable diagnostic work. Few things are more annoying than the discovery of a simple power supply failure after complex, time-consuming troubleshooting procedures.

Console Troubleshooting

In many cases the initial investigation discloses an appropriate line of attack, but does not in itself pinpoint the location of a malfunction. Troubleshooting from the console localizes a malfunction within a small section of machine logic, and is usually accompanied by use of the MAINDECs (with or without marginal checking, depending on the nature of the malfunction). Intermittents caused by weakened components can almost always be aggravated and thus transformed to relatively consistent malfunctions by use of marginal checking with an appropriate MAINDEC (Section 9.2), selected on the basis of information derived from the initial investigation. Intermittent connections, however, are substantially more difficult to locate.

If the malfunction does not show up in the first run without margins, perform the operation in which the trouble was initially observed, using the same user program. In-out equipment faults frequently give indications similar to those caused by processor malfunctions. Since the entirely solid state processor is inherently more reliable, check the peripheral equipment first. Faulty ground connections between periphery and processor are common sources of trouble.

If a malfunction is caused by loading conditions unique to a user program, it can generally be detected by a MAINDEC run with margins. Malfunctions caused by differences in loading conditions show up at narrower margins than those listed in the last few plots on the PM voltage charts. If the MAINDEC discovers the error during application of marginal check, do not restart the computer; use the error halt listings to diagnose the trouble.

Console troubleshooting procedures for locating catastrophic malfunctions, or those made consistent through use of margins, should be directed toward discovering a pattern of consistency among the errors. Although some ingenuity may sometimes be necessary to discover it, all malfunctions display consistency in the errors they produce; separate error diagnoses must intersect at a common fault point.

Logic Troubleshooting

Logic troubleshooting is detail work performed on a small section of the logic after a malfunction has been detected but before the source has been identified. The two main parts of troubleshooting are to reproduce the malfunction and then to identify the source. A common technique for reproducing malfunctions is the use of diagnostic loops, though at times it may be necessary to reproduce the exact conditions under which the failure first occurred, e.g., by repeating the program that was running at the time. For an intermittent fault it may be useful to try aggravation techniques while a diagnostic loop is being run.

In general a diagnostic loop is a small string of instructions that causes the computer to perform some operation and checks that it is executed correctly; usually the error is indicated by having the computer halt. If the operation is executed correctly, the program starts over; in this manner the computer runs until a malfunction is encountered. The technician may design his own diagnostic loop and toggle it in from the console, but it is usually easier

to modify the maintenance program. MAINDEC 601 consists entirely of small diagnostic routines seldom containing more than ten instructions. Each routine halts the processor on an error by using JRST 4 with the address pointing to the first loaction of the next routine. Any routine may be converted into a diagnostic loop by replacing the instruction following the halt with a jump to the beginning of the routine.

Aggravation is intended to convert intermittent malfunctions to catastrophic ones. The technique involves either marginal check voltages or vibration (weak component detection by marginal checking is part of console troubleshooting described above). As long as reasonable care is used to avoid inflicting permanent damage, maintenance personnel should not hesitate to twist and probe at connections, cables, plugs or modules. All connections in the system are designed for excellent reliability; so plugs, cables, or sockets should be impervious to any reasonable amount of pulling or flexing. Although PDP-6 systems are tested before leaving the factory, nevertheless, a poorly soldered connection occasionally shows up; this type of fault usually appears as an intermittent and is sometimes very difficult to locate. Poorly soldered connections are more likely to appear in mounting panel or cable connections than within modules. Well hidden malfunctions of this type are occasionally located by tapping suspect modules with the plastic handle of a screwdriver or other such harmless implement.

Once the malfunction can be reproduced, the next step is to identify the source and correct it. Three common techniques are substitution, observing panel indicators, and signal tracing. Substitution is simply replacing a suspect module to see whether a malfunction is thereby cured. Whenever possible, swap modules rather than substituting a spare, e.g., swap counter or register bits. If the malfunction moves to the new location, the trouble is in the exchanged module. If the malfunction has not moved, the trouble is likely to be elsewhere, probably in the logic that feeds the module. Regardless of the outcome of the swap the good module should be returned to its original location – it is a desirable maintenance practice to keep each module in a specific place in the system.

Panel indicators may be used to detect malfunctions by following machine state changes. Using REPEAT or REPEAT BYPASS and the SPEED controls, a program loop can be run at slow speed by pausing after each instruction, memory access, AR subroutine, or even every step in a shift-count. A complete discussion of single step operation is presented in 9.1d.

Repetitive memory access can be produced without using any instructions by repeating an examine or deposit operation from the console. A single instruction may be repeated by setting it in the DATA switches and using REPEAT with the EXECUTE key. A single instruction in memory can be repeated by using START or READ IN. Complete information on the repetition of key functions is given in the final paragraph in 5.1.

Signal tracing requires a scope. With the machine running in an exercise loop, synchronize the oscilloscope sweep to one of the main sequence time pulses or to a pulse that should generate the suspect pulse or event. (The flow charts are very helpful for selecting an appropriate pulse. All pulses and levels are available at module output pins; all pin locations are shown on the logic drawings, and every signal name is prefixed by a mnemonic indicating the source logic for it.) In order that events may be viewed on the scope, the exercise loop should not halt the machine when an error occurs (to change a diagnostic loop into an exercise loop, replace the skip with a jump). It is difficult to pinpoint intermittents using signal tracing, so an effort should be made to change intermittent faults into catastrophic ones which can be located far more easily. When constructing an exercise loop keep it simple and use a minimum of instructions. For example, a simple loop can be obtained by starting the processor at a memory location that contains a jump to itself, or by repeating key functions.

Once the malfunction has been isolated and is presumably corrected, be particularly careful during validation to ensure that the module or component replaced was really at fault, and thus that the malfunction was actually cured by the replacement.

b Repair

Once a malfunctioning module has been located, repair should be made immediately by substitution, as intensive system scheduling requirements preclude use of the processor to aid in submodular troubleshooting. It is expected that defective modules will be returned to DEC for repair or replacement; those who wish to undertake module repair may refer to procedures given in PDP-6 Circuits. When replacing a module, make sure the new one is properly prepared to replace the old: in some cases adjustments must be made or internal connections must be made or broken.

Internal Connections

Most inverter, diode, capacitor-diode, and decoder modules have provision for internal jumpering to connect clamped loads at the output inverter collectors. These modules are shipped with all clamped loads connected; to prepare a replacement module for insertion in any given location the unused jumpers must be disconnected (e.g., with wire cutters). Some modules have jumpers for other purposes; for example some flip-flops, decoders, etc., have jumpers to select the connections to the module pins, and some delay lines require selection of the tap to choose the delay duration. Whether jumpers must be cut or soldered in depends upon the type of module. The utilization module lists (UML, see Appendix 1) give the jumper configurations in all locations for modules in which jumpers may be used. The jumper codes as they appear on the UML depend upon the type of module and the way in which the jumpers are used; these codes are explained in the circuit manual.

Delays

Most delay modules contain distributed-constant delay lines that cannot be adjusted. There are, however, three types of adjustable delays, 1304, 4301, and 4303. Whenever one of these modules is replaced, the new module must be adjusted to give the correct time interval. The 1304 and 4301 may be adjusted as follows:

1. Set up an exercise loop that repeatedly triggers the delay.

2. Set up an oscilloscope to observe the duration of the level output at pin J of the module (−3 volts during the delay interval, ground otherwise).

3. Set the scope sweep to the calibrated position, and select a per-centimeter sweep that displays the entire duration of the level output. The output is adjusted to the required duration by means of a screwdriver trimpot accessible through a hole in the rear of the module frame.

The 4303 has flip-flop type outputs. Use the above method but connect the scope at pin W to observe the negative output, or at pin U to observe the ground output. The delays for

power clear and punch motor turnoff are not critical, so the replacement may not need adjustment at all; the delay period for detecting a nonexistent memory should not exceed 100 μsec. The 4303 in the repeat logic is adjusted by means of the speed controls: the selector chooses one of five external capacitors or none at all (the smallest is internal); the potentiometer varies the available charge current for the selected capacitor.

Tape Reader

When replacing any reader circuit card having a trimpot adjustment, perform the entire 1000-hour PM checkout (and all checks subordinate to it). This action not only provides for the required card adjustment but also serves as a validation procedure for the repair.

Card Reader

Replacing any photoamplifier package in the card reader requires complete readjustment of the exciter lamp brilliance and of the strobe pulse coincidence, using the procedures of Sections 3.16 and 3.17 of the card reader manual. Replacement of any mechanical part that involves an adjustment requires performance of the adjustment procedure as part of the reassembly; the replacement procedures of Section 4 in the reader manual should be followed by appropriate calibration procedures from Section 3.

c Validation and Log Entry

Following replacement of any electrical component in the system, tests should be performed to ensure correction of the fault condition and to trim up final adjustments or signal levels affected by the replacement. Validation normally requires performance of the PM procedure most applicable to that portion of the system in which the fault occurred. If repair or replacement is made in an area not normally checked during preventive maintenance, the appropriate diagnostic program (MAINDEC) should be run, or an alternate operational test should be devised to ensure proper operation. It is strongly suggested that the entire preventive maintenance task be performed for the malfunctioning system component. This action ensures detection of faults that may have been masked by the just-corrected malfunction and may reveal a failure mechanism, perhaps still extant elsewhere, that is likely to cause recurrence of the fault. If the entire PM procedure can be performed while the equipment is down and available, it need not be scheduled for the whole PM interval.

Corrective maintenance activities are not complete until they are recorded in detail in the maintenance log. Include all data indicating symptoms given by the fault, the method of fault detection, the component at fault, and any comments that might be helpful in future diagnosis. Suggest improvements in PM procedures if indicated. A complete and detailed write-up of diagnostic procedures used to locate a real dog is particularly important. Although some details may seem insignificant and obvious, they should nevertheless be included since examination of the log will be a major part of the initial investigation for a future malfunction. Reduce down time by recording everything.

APPENDIX 1

ENGINEERING DRAWINGS

Reduced copies of engineering drawings are included in Volume 2 of this manual. Chapter 4 explains the drawing numbers and type codes that identify engineering drawings for all components in a PDP-6 system and details the notation and conventions used in the block schematics and flow charts, which are the basis for learning and maintaining the equipment. However, there are many other engineering drawings used primarily for reference in maintenance. For a given system component, every drawing number has the type number of that component following the size letter; drawings that apply to the entire system are identified by "6" as the type number. In the following discussion all drawings mentioned are D size unless otherwise specified; the type of drawing in each case is identified by the code, and individual drawings are specified by the number or mnemonic code that follows the drawing serial number.

The master drawing list (MDL) for the entire system is an A-6 drawing with no individual drawing number. It lists the MDLs for all system components; these are also A size and identified only by the equipment type number and drawing serial number. Each MDL lists all drawings for the system component by title, number, code, revision letter, and number of sheets. The only other drawing for the entire system is a module list ML; in this instance the system module list SML, which lists the module requirements for all system components (this drawing is discussed further in Appendix 2).

The 166 MDL lists first the SD, FDs, and BSs that accompany this manual. Detailed information about the modules that make up the processor is shown in the utilization module lists UML, of which there are ten, numbered 14 to 23. Each UML shows three mounting panels, each divided into 25 sections representing the plug-in locations. Above each location is the type number and jumper code for the module occupying it. Each location is further partitioned according to the individual circuits (flip-flops, pulse amplifiers, inverters) in the module. Circuits are identified by logical function, using the same signal names that appear on the block schematics. For example, a pulse amplifier is labeled by naming the output pulse; an inverter or diode gate, by the output of the net in which it is used.

Of the various cable diagrams, CD, the one that shows the logical signals on the in-out and memory busses is included in the manual (IOMB, Figure 7-10). Another CD that is closely associated with the logic is CONS, which shows the connections to all operating keys and switches on the console. This drawing shows the generation of logic levels from the keys and switches and the connections to the main power control including the 60-cycle signal for the clock flag. Note how the on position of the key-locked console disabling switch holds system power on but disrupts the console power that enables the keys and switches. Note also that the NO and NC labels on switch contacts bear no invariant relationship to the normal position of the switch; these are merely the labels molded onto the switch case for physical identification of connections.

A pair of drawings CBL-2 and CBL-3 show all levels from the logic to indicators and switch registers on the console and on the bay indicator panels. CBL-1 shows the locations of all plugs and jacks throughout the four bays; CH shows the configuration of the main cable harness and its many branches. There are also a large number of cable lists CL that show the connections made from cables to module connector pins and identify the signals carried on the various cable lines.

All connections made among the module connector pins are shown in a series of wiring diagrams WD. Each drawing shows three mounting panels and is identified by the panel letters. Because of the complexity of the wiring in the processor, each set of three panels is shown in two drawings, one showing for example the grounds and pulses, the other the levels and flip-flops outputs (there are also other combinations).

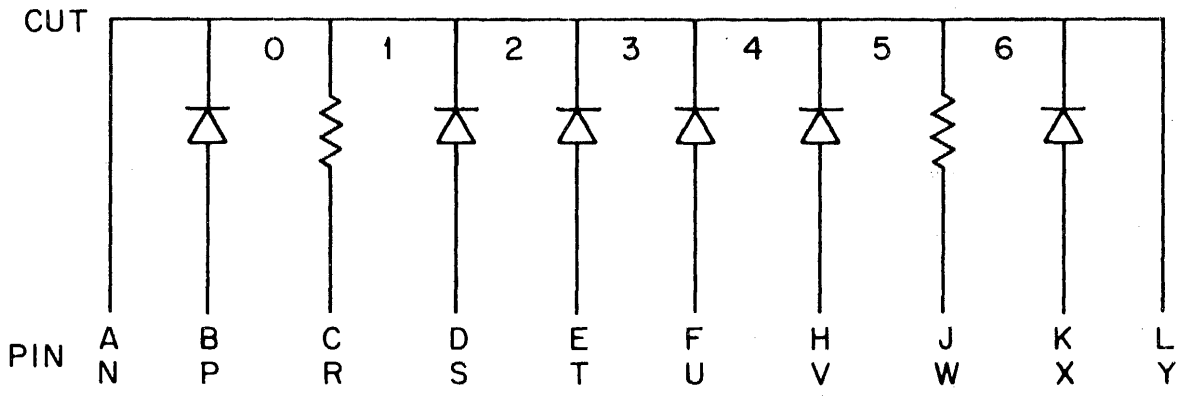
Power wiring is shown in the PW drawings, ACPW and DCPW. Each shows the location of power controls and power supplies on the inside of the rear plenum doors viewed from the front, i.e., as though one were looking right through the mounting panels from the wiring side. ACPW shows all ac connections from the power controls and fans (for machines using 240 volts, 50 cycles, drawing 50ACPW is substituted). The interlock circuit in the lower right is associated with the plenum doors in bays 3 and 4 and may be used for example in the high-voltage line to a display mounted in the console. The interlock switches have three positions including an override the technician may use during maintenance. DCPW shows all of the dc wiring from the supplies to the terminal strips. All wiring is color coded +10 red, −15 blue, ground black. As shown in

the lower left, all terminal strips and the terminals on the 728 and 734 Supplies use the same color code as the wiring; the terminals on the 778 Supply are different but the wires connected to them use the standard coding. The strip between the pair of representations for each bay is a power connector bracket, whose layout and wiring are shown at the lower left and right. These are mounted horizontally about halfway up the sides of the bays. The smaller horizontal strips in the right block for each bay are actually vertical and located at one end of every mounting panel. The green and yellow terminals and wires carry the positive and negative variable voltages for marginal check (the wires are color coded identically). In the upper right are the connections to the marginal-check control panel and the marginal-check bus. This bus carries the variable voltages for marginal checking to other equipment at the site and also carries the −15-volt turnon signal for the memories. A wiring diagram, MCLT, shows the connections from the controls on the marginal-check panel to the variable power supply at the rear of the same bay. Connections to the elapsed time meter are at the right.

Many processor drawings, such as the PWs, also present information about the in-out control units included in this manual. However, each device has its own UML, WD, and various other cable drawings as listed in the appropriate MDL. For the card reader there are a number of Burroughs logic schematics not included in the card reader manual. These include drawings H-1182627 to H-1182640, H-11900107, H-11900115, H-11877446, H-11877453, and H-11877461. The last three are lists of equations for the reader logic; the others are computer-printed block schematics. These schematics show signal flow and give pin numbers and part designations of electrical components, plug-in units, and connectors, all of which are shown by rectangles. Wires are indicated by dashed lines; a "0" at an intersection point of two wires represents a common connection. The information is laid out on a rectangular coordinate system, with all inputs to the drawing in the input column at the left, all outputs at the right. The columns between are numbered and all horizontal rows are identified by a letter and a number. All input signals are identified by at least two rows of information, the top row giving the signal name, the bottom row indicating its source by page and row. Supplementary data may be included between the top and bottom rows. Output information is identified in the same way, giving the destination of the signal by page and row.

All boxes are eight character positions wide. The first two positions are used exclusively for the pin connections of input lines, and in all but the bottom row, the final two positions are

used for pin connections of output lines. At the top center of each box is a part designation, and the bottom row gives the location. Section 2.2 of the card reader manual explains the system used to designate the locations of all electrical parts. The location of a plug-in unit is given in the form AAB5L6, which means that the unit is in rack A, panel A, row B, position 5 and that its key pin (the one larger than the others) plugs into row L, column 6 of the socket. The part designation is usually the initial letters of the words describing the part or some standard symbol such as K for relay; some of the part designations are given in Burroughs Section 2.5, and wherever the meaning of the part designation cannot be deduced, the location designation can be used to find and thus identify the part. If the box represents a pluggable unit, such as a photo amplifier, low-speed switch, or diode stick, its location also appears in Table 2.5-1. A schematic and description of the part can be found in Chapter 6. The majority of the plug-in units are diode sticks which are not described, but are all identical and are wired as follows:



A diode stick is designated in the drawings by D followed by a letter or number. It may be cut to form two or three diode gates: the top, center, or bottom part of the stick is represented by the box if a T, C, or B follows the D. A number indicates the position of the cut or cuts made (no cut is indicated by NC). Replacements are sent uncut, and the technician must cut them to conform with the configurations required for any given location. In addition to pin connections, the remaining rows may contain other information. Row 2 often indicates the part of the logic the circuit is used in. For a diode stick, rows 3 and 6 give information about the resistors at positions 1 and 6 and the gates associated with them: an R in position 3 indicates

that the resistor is tied to −12 volts; a numeral in position 4 indicates the number of diodes used in the gate; A or O in position 5 or 6 indicates the use of the gate as logical AND or OR. Information contained in other rows depends upon the element represented; for example in a box representing a relay, NO or NC in row 5 indicates normally open or normally closed.

APPENDIX 2

SPARES

For a large-scale system such as PDP-6, even moderate downtime is generally far more expensive than the cost of maintaining a fully adequate stock of spares. The system module list SML lists the quantities of all modules, power supplies and power controls used in all system components (i.e., processors, memories, in-out control units) available in PDP-6. However, individual systems may differ considerably in quantity and type of system components, so DEC Field Service supplies each user with a list geared to his particular system. This list gives the total quantity and number of spares recommended for every module, power supply, and power control. Field Service also supplies spares lists of electrical components needed for repair of DEC modules and lists of electrical and mechanical spares for the various in-out devices. It is further recommended that the user keep a spare tape reader, tape punch, and keyboard-printer, and stock the following miscellaneous items for the system.

Quantity	Item	Part Number	Vendor
1	Switch for bat-handle key	Telever Switch 16006 Cat. No. S-302 Reworked per DEC drawing MA-C-01510	Switchcraft Inc.
5	Subminiature toggle switch SPDT	6AT1	Micro Switch
1	Lockout switch DPDT	1575-L	Arrow Hart
2	Rotron fan	53E168 Type CFG	Rotron Mfg. Co.
2	Rotron filter	34-X1431	Rotron Mfg. Co.
10	Indicator lamp	MC48-639B	Transistor Electric
1	Howard fan	12-80-15	Howard Industries

APPENDIX 3

GLOSSARY

It is not intended that use of this glossary shall be substituted for reading Chapter 4 or for using the flow charts when following any sequence of events in the logic drawings. This glossary gives the meanings of all prefix codes and all terms used in the signal names for the processor and the four in-out devices. Only a few complete signal names are included because all names are composites of standard terms, and their meanings are usually quite obvious to anyone who knows the meanings of the terms and has read 4.2. No generating conditions are given for logical functions because these are listed in the flow charts. The processor flags are listed and described in 2.1c, and all inputs to the logic from the console are discussed in 3.1a. With each prefix code below are listed in parentheses the numbers of the figures to which the code applies.

A (5-3)	Address cycle.
AC	Accumulator.
AC0	Accumulator whose address is 0.
AC2	Accumulator following the one addressed by an instruction.
ACBM (5-9)	Accumulator bit modify; the logical compare instructions, which test bits of AC as specified by a mask (either E or C(E)) and may modify the masked bits.
ACCP (5-9)	Accumulator compare, i.e., those arithmetic compare instructions that compare AC with either E or C(E).

ACCP ET AL TEST ACCP ETC COND	ACCP ETC COND is the skip or jump condition as specified by bits 7 and 8 in ACBM, ACCP, or MEMAC. Since bit 6 of the instruction code specifies whether the skip or jump is to be made on the presence or absence of the condition specified by bits 7 and 8, the condition signal is exclusive ORed with IR6 to generate ACCP ET AL TEST, which is sensed by the program control logic.
ACK	Acknowledge.
ADDR	Address.
A LONG	Flip-flop set whenever any operations are performed in the calculation of an address.
AOBJP	Add one to both and jump if positive.
AOBJN	Add one to both and jump if negative.
AR (6-4, 5, 6, 7, 8, 9, 10)	Arithmetic register and associated control logic, special inputs, subroutines, and flags.
AR COM CONT	AR complement control flip-flop; set at the beginning of subtraction or decrementing in AR to indicate that result must be complemented following the addition or incrementing used.
AR CRY COMP	AR carry complete.
AR CRY0	Flip-flop set by any carry out of AR0.

AR CRY1	Flip-flop set by any carry out of AR1.
AR = FP HALF	Level asserted when the fractional part of AR as a floating-point number is of magnitude 1/2.
AR OV SET	Condition that determines overflow in AS or MEMAC.
AR SBR	Level that causes the execute cycle to pause at ET4 for an AR subroutine.
AS (5-10)	Add-subtract.
ASH	Arithmetic shift.
ASHC	Arithmetic shift combined.
B	Binary; buffered.
BLK	Block.
BLKI	Block in.
BLKO	Block out.
BLT (6-18)	Block transfer.
BLT DONE	Level that terminates a block transfer either permanently because the block is complete or temporarily to handle an interrupt.
BLT LAST	Inhibits AC storage when block transfer is complete.
BOOLE (5-10)	Boolean instructions.

BOTH	Mode in which result is deposited in both AC and memory.
CAO	Character add one (= IBP).
CBHL	Level to the card reader causing it to place the upper half of a binary column on the output lines.
C( )	Contents of.
C(E)	Contents of location specified by effective address.
CFAC (6-17)	Computer floating-arithmetic connection (subroutine interface).
CH (6-19, 20)	As a prefix, character operations (byte manipulation); also channel.
CHG	Change.
CH INC CH INC OP CH ∿INC OP	These three levels control the first part in a character operation. CH INC is asserted during the first part of any instruction that calls for incrementing the pointer. The other two levels are mutually exclusive within the first part of a character operation. CH INC OP causes incrementing of the pointer; CH ∿INC OP inhibits incrementing either because the instruction does not require it or the first part of an incrementing character operation is being repeated following a priority interrupt between the two parts.

CL	Clear.
CLK	Clock.
CLR	Clear.
CMC	Core memory control; prefix for control signals from memory (core or fast) over the bus.
COM	Complement.
COMP	Complete.
CONI	Conditions in.
CONO	Conditions out.
CONSO	Conditions in and skip if one.
CONSZ	Conditions in and skip if zero.
CONT	Control; continue.
CPA (8-5)	Processor I/O interface. This is the interface at the other end of the bus from IOT control; through it the processor, via IOT control and the bus, controls itself as a device.
CR (8-13, 14, 15)	Card reader.
CREL	Signal from the card reader that indicates a validity or read check error.
CRL	Signal from the card reader that indicates it is not ready for operation.

CRY	Carry.
CYC	Cycle.
DATAI	Data in.
DATAO	Data out.
DC (6-20)	Deposit character.
DCA	Dc adder.
DEP	Deposit.
DIR	Direct, i.e., C(E) rather than E.
DIV	Divide.
DN	Do nothing.
DPC	Deposit character (= DPB).
DPCI	Index pointer and deposit character (= IDPB).
DS (6-25, 26)	Divide subroutine.
E (5-5)	As a prefix, execute cycle; also effective address.
E LONG	Level that causes processor to use second half of execute cycle.
EOF	End of file.
EX (5-13)	As a prefix, executive mode; also examine.
EXCH	Exchange.

EXEC	Execute.
F (5-4)	As a prefix, fetch cycle; also flip-flop.
FA (6-22)	Floating add.
FAD	Floating add.
FC(C(ACLT))	Fetch the contents of the location addressed by the number in ACLT.
FC(C(ACRT))	Fetch the contents of the location addressed by the number in ACRT.
FD (6-22)	Floating divide.
FDV	Floating divide.
FE (6-14, 15)	Floating exponent.
FM (6-22)	Floating multiply.
FMC	Fast memory control.
FMP	Floating multiply.
FP (6-23)	As a prefix, floating-point exponent calculate subroutine; also floating-point instructions.
FP/CH	IR decoder output for the floating-point instructions and character operations.
FS (6-19)	Floating scale.
FSC	Floating scale.

FSB	Floating subtract.
FWT (5-9)	Full-word transfer.
HWT (5-9)	Half-word transfer.
I (5-3)	Instruction cycle.
ILL	Illegal.
ILLEG	Illegal.
INC	Increment.
INH	Inhibit.
INST	Instruction.
INT	Interrupt.
IO	In-Out.
IOB (8-1,2)	In-out bus; IOBi (i = 0, ..., 35) is data line i on the bus.
IOS (5-7)	In-out select.
IOT (8-1)	In-out transfer.
IOT A	IR decoder output for IOT (replaced by a UUO in user mode).
IR (5-7, 8)	Instruction register and associated decoding nets.
JFCL	Jump on flag and clear.
JP (5-10)	Jump and pushdown.

JRA	Jump and restore accumulator.
JRST	Jump and restore.
JRST A	IR decoder output for JRST (replaced by a UUO in user mode if a halt or PI dismiss).
JSA	Jump and save accumulator.
JSP	Jump and save program counter.
JSR	Jump to subroutine.
K (5-2)	Key cycle.
KEY (5-1, 2)	Key.
KEY MANUAL	Level generated when any initiating key is pressed.
LC (6-20)	Load character.
LDC	Load character (= LDP).
LDCI	Increment pointer and load character (= ILDP).
LSB	Least significant bit.
LSH	Logical shift.
LSHC	Logical shift combined.
LT	Left.
MA (7-1, 2, 3)	Memory address.
MAI (7-2)	Memory address interface.

MAJ	Majority gate.
MARK	Teletype signal representing a 1.
MAS	Address switch.
MA SW	Address switch.
MB (6-1, 2, 3, 4; 7-6)	Memory buffer and associated control logic, special inputs, and data interface with memory bus.
MC (7-8, 9)	Memory control.
MD	Multiply-divide.
MEM	Memory; as a mode, result is stored only in E.
MEMAC (5-10)	Memory and accumulator modification and test instructions; these test AC against zero for a jump, and may or may not increment AC, or test C(E) against zero for a skip, and may or may not increment C(E).
MI (7-7)	Memory indicators.
MISC BITS	Miscellaneous bits.
MOVN, M	Move negative or magnitude.
MOV, S	Move or move and swap.
MP (6-21)	Multiply.
MQ (6-7, 11,12, 13)	Multiplier quotient register and associated control logic and special inputs.

MQ36	Extra MQ bit for use in multiply.
MR (5-1, 2)	Master.
MR CLR	Pulse that prepares processor for each main sequence or console function.
MR START	Pulse that clears entire computer at power turnon or when IO RESET is pressed.
MS (6-24)	Multiply subroutine.
MSB	Most significant bit.
MUL	Multiply.
MULT	Multiply.
NEGATE	Form the arithmetic 2's complement.
NR (6-27)	Normalize return.
NXT	Next.
OP	Operation.
OV	Overflow.
PC (5-11, 12)	Program counter and control.
PDL OV	Pushdown list overflow.
PI (8-3, 4)	Priority interrupt.
PIA	Priority interrupt assignment.
PICH	PI channel address.

PIH	PI hold.
PIO	PI on.
PIR	PI request.
POP	Pullout.
POPJ	Pullout and jump.
PR (7-4)	Protection.
PSE	Pause.
PTP (8-9, 10)	Paper tape punch.
PTR (8-6, 7, 8)	Paper tape reader.
PUSH	Pushdown.
PUSHJ	Pushdown and jump.
PWR	Power.
PWR CLR	Pulses that clear computer at power turnon.
RD	Read.
REL	Relocation.
REM	Remainder; floating-point mode that stores low-order half of result in AC2.
REQ	Request.
RIM	Readin mode.
RLA (7-5)	Relocation adder.

RLR (7-5)	Relocation register.
ROT	Rotate.
ROTC	Rotate Combined.
RPT	Repeat.
RQ	Request.
RS	Restart.
RST	Restore.
RT	Right.
RUN (5-1)	Run.
S (5-6)	Store cycle.
SBR	Subroutine; also a mnemonic for subroutine card 1260.
SC (6-14, 15, 16)	Shift counter.
SEL	Select.
SH (6-20)	Shift.
SH AC2	Double-length shift.
SHC (6-7)	Shift connection.
SPACE	Teletype signal for 0.
SR	Shift register.
ST	Start.

STATUS	Any IOT instruction that examines status, i.e., CONI, CONSO or CONSZ.
STB	Strobe.
SW (5-1)	Switch.
SWAP	Interchange the left and right halves of a word.
SUB	Subtract.
T	Time.
TST	Test.
TTI (8-11, 12)	Teletype input.
TTO (8-11, 12)	Teletype output.
TTY (8-11, 12)	Teletype.
UUO (5-10)	Programmed operator (unused op code); a UUO is performed when one is programmed or when it replaces an illegal user instruction.
UUO A	IR decoder output for UUO.
WR	Write.
XCT (5-10)	Execute.
2XX	Level indicating that IR contains an octal code beginning with 2.
25X	Level indicating that IR contains an octal code beginning with 25.

APPENDIX 4

INSTRUCTION CODES

Octal	Mnemonic	Octal	Mnemonic	Octal	Mnemonic
000		172	FDVM	242	LSH
⋮	UUO	173	FDVB	243	
⋮		174	FDVR	244	ASHC
077		175	FDVRL	245	ROTC
100		176	FDVRM	246	LSHC
⋮		177	FDVRB	247	
⋮		200	MOVE	250	EXCH
131		201	MOVEI	251	BLT
132	FSC	202	MOVEM	252	AOBJP
133	IBP	203	MOVES	253	AOBJN
134	ILDB	204	MOVS	254	JRST
135	LDB	205	MOVSI	255	JFCL
136	IDPB	206	MOVSM	256	XCT
137	DPB	207	MOVSS	257	
140	FAD	210	MOVN	260	PUSHJ
141	FADL	211	MOVNI	261	PUSH
142	FADM	212	MOVNM	262	POP
143	FADB	213	MOVNS	263	POPJ
144	FADR	214	MOVM	264	JSR
145	FADRL	215	MOVMI	265	JSP
146	FADRM	216	MOVMM	266	JSA
147	FADRB	217	MOVMS	267	JRA
150	FSB	220	IMUL	270	ADD
151	FSBL	221	IMULI	271	ADDI
152	FSBM	222	IMULM	272	ADDM
153	FSBB	223	IMULB	273	ADDB
154	FSBR	224	MUL	274	SUB
155	FSBRL	225	MULI	275	SUBI
156	FSBRM	226	MULM	276	SUBM
157	FSBRB	227	MULB	277	SUBB
160	FMP	230	IDIV	300	CAI
161	FMPL	231	IDIVI	301	CAIL
162	FMPM	232	IDIVM	302	CAIE
163	FMPB	233	IDIVB	303	CAILE
164	FMPR	234	DIV	304	CAIA
165	FMPRL	235	DIVI	305	CAIGE
166	FMPRM	236	DIVM	306	CAIN
167	FMPRB	237	DIVB	307	CAIG
170	FDV	240	ASH	310	CAM
171	FDVL	241	ROT	311	CAML

Octal	Mnemonic	Octal	Mnemonic	Octal	Mnemonic
312	CAME	367	SOJG	444	EQV
313	CAMLE	370	SOS	445	EQVI
314	CAMA	371	SOSL	446	EQVM
315	CAMGE	372	SOSE	447	EQVB
316	CAMN	373	SOSLE	450	SETCA
317	CAMG	374	SOSA	451	SETCAI
320	JUMP	375	SOSGE	452	SETCAM
321	JUMPL	376	SOSN	453	SETCAB
322	JUMPE	377	SOSG	454	ORCA
323	JUMPLE	400	SETZ	455	ORCAI
324	JUMPA	401	SETZI	456	ORCAM
325	JUMPGE	402	SETZM	457	ORCAB
326	JUMPN	403	SETZB	460	SETCM
327	JUMPG	404	AND	461	SETCMI
330	SKIP	405	ANDI	462	SETCMM
331	SKIPL	406	ANDM	463	SETCMB
332	SKIPE	407	ANDB	464	ORCM
333	SKIPLE	410	ANDCA	465	ORCMI
334	SKIPA	411	ANDCAI	466	ORCMM
335	SKIPGE	412	ANDCAM	467	ORCMB
336	SKIPN	413	ANDCAB	470	ORCB
337	SKIPG	414	SETM	471	ORCBI
340	AOJ	415	SETMI	472	ORCBM
341	AOJL	416	SETMM	473	ORCBB
342	AOJE	417	SETMB	474	SETO
343	AOJLE	420	ANDCM	475	SETOI
344	AOJA	421	ANDCMI	476	SETOM
345	AOJGE	422	ANDCMM	477	SETOB
346	AOJN	423	ANDCMB	500	HLL
347	AOJG	424	SETA	501	HLLI
350	AOS	425	SETAI	502	HLLM
351	AOSL	426	SETAM	503	HLLS
352	AOSE	427	SETAB	504	HRL
353	AOSLE	430	XOR	505	HRLI
354	AOSA	431	XORI	506	HRLM
355	AOSGE	432	XORM	507	HRLS
356	AOSN	433	XORB	510	HLLZ
357	AOSG	434	IOR	511	HLLZI
360	SOJ	435	IORI	512	HLLZM
361	SOJL	436	IORM	513	HLLZS
362	SOJE	437	IORB	514	HRLZ
363	SOJLE	440	ANDCB	515	HRLZI
364	SOJA	441	ANDCBI	516	HRLZM
365	SOJGE	442	ANDCBM	517	HRLZS
366	SOJN	443	ANDCBB	520	HLLO

Octal	Mnemonic	Octal	Mnemonic	Octal	Mnemonic
521	HLLOI	571	HRREI	641	TLC
522	HLLOM	572	HRREM	642	TRCE
523	HLLOS	573	HRRES	643	TLCE
524	HRLO	574	HLRE	644	TRCA
525	HRLOI	575	HLREI	645	TLCA
526	HRLOM	576	HLREM	646	TRCN
527	HRLOS	577	HLRES	647	TLCN
530	HLLE	600	TRN	650	TDC
531	HLLEI	601	TLN	651	TSC
532	HLLEM	602	TRNE	652	TDCE
533	HLLES	603	TLNE	653	TSCE
534	HRLE	604	TRNA	654	TDCA
535	HRLEI	605	TLNA	655	TSCA
536	HRLEM	606	TRNN	656	TDCN
537	HRLES	607	TLNN	657	TSCN
540	HRR	610	TDN	660	TRO
541	HRRI	611	TSN	661	TLO
542	HRRM	612	TDNE	662	TROE
543	HRRS	613	TSNE	663	TLOE
544	HLR	614	TDNA	664	TROA
545	HLRI	615	TSNA	665	TLOA
546	HLRM	616	TDNN	666	TRON
547	HLRS	617	TSNN	667	TLON
550	HRRZ	620	TRZ	670	TDO
551	HRRZI	621	TLZ	671	TSO
552	HRRZM	622	TRZE	672	TDOE
553	HRRZS	623	TLZE	673	TSOE
554	HLRZ	624	TRZA	674	TDOA
555	HLRZI	625	TLZA	675	TSOA
556	HLRZM	626	TRZN	676	TDON
557	HLRZS	627	TLZN	677	TSON
560	HRRO	630	TDZ	7-00	BLKI
561	HRROI	631	TSZ	7-04	DATAI
562	HRROM	632	TDZE	7-10	BLKO
563	HRROS	633	TSZE	7-14	DATAO
564	HLRO	634	TDZA	7-20	CONO
565	HLROI	635	TSZA	7-24	CONI
566	HLROM	636	TDZN	7-30	CONSZ
567	HLROS	637	TSZN	7-34	CONSO
570	HRRE	640	TRC		

APPENDIX 5

TELETYPE CODE

The 8-bit codes are listed below. An asterisk indicates a code that has no effect on the Model 35. Alternate characters are listed in parentheses. The characters actually contain only seven information bits. The eighth bit may be used for parity, but currently all machines are set up so that the eighth bit is a mark, and thus the codes generated from the keyboard are 200_8 greater than the corresponding ASCII codes.

Octal Code	ASCII Character	Key Combination	Remarks
200	NULL	SHIFT CTRL P	Null.
201*	SOM	CTRL A	Start of message.
202*	EOA	CTRL B	End of address.
203*	EOM	CTRL C	End of message.
204	EOT	CTRL EOT	End of transmission; shuts off TWX machines.
205	WRU	CTRL WRU	"Who are you?" Triggers "Here is...," at remote station.
206*	RU	CTRL RU	"Are you...?"
207	BELL	CTRL BELL	Rings the bell.
210*	FE	CTRL H	Format effector.
211	HT	CTRL TAB	Horizontal tab.
212	LF	LINE FEED	Line feed.
213	V TAB	CTRL VT	Vertical tab.
214	FF	CTRL FORM	Form feed.
215	CR	RETURN	Carriage return.
216*	SO	CTRL N	Shift out.
217*	SI	CTRL O	Shift in.
220*	DC0	CTRL P	Device control reserved for data linc escape.

Octal Code	ASCII Character	Key Combination	Remarks
221	DC1	CTRL Q	Turns reader on.
222*	DC2	CTRL TAPE	Turns punch on.
223	DC3	CTRL XOFF	Turns reader off.
224*	DC4	CTRL ~~TAPE~~	Turns punch off.
225*	ERR	CTRL U	Error.
226*	SYNC	CTRL V	Synchronous idle.
227*	LEM	CTRL W	Logical end of media.
230*	S0	CTRL X	Separator, information.
231*	S1	CTRL Y	Separator, data delimiter.
232*	S2	CTRL Z	Separator, words.
233*	S3	SHIFT CTRL K	Separator, groups.
234*	S4	SHIFT CTRL L	Separator, records.
235*	S5	SHIFT CTRL M	Separator, files.
236*	S6	SHIFT CTRL N	Separator, miscellaneous.
237*	S7	SHIFT CTRL O	Separator, miscellaneous.
240	Space	Space bar	
241	!	SHIFT !	
242	"	SHIFT "	
243	#	SHIFT #	
244	$	SHIFT $	
245	%	SHIFT %	
246	&	SHIFT &	
247	´(')	SHIFT ´(')	
250	(	SHIFT (	
251	)	SHIFT)	
252	*	SHIFT *	
253	+	SHIFT +	
254	,	,	
255	-	-	
256	.	.	

Octal Code	ASCII Character	Key Combination	Remarks
257	/	/	
260	Ø	0	Zero, prints with a slash
261	1	1	
262	2	2	
263	3	3	
264	4	4	
265	5	5	
266	6	6	
267	7	7	
270	8	8	
271	9	9	
272	:	:	
273	;	;	
274	<	SHIFT <	
275	=	SHIFT =	
276	>	SHIFT >	
277	?	SHIFT ?	
300	↖(@)	SHIFT ↖ (@)	
301	A	A	
302	B	B	
303	C	C	
304	D	D	
305	E	E	
306	F	F	
307	G	G	
310	H	H	
311	I	I	
312	J	J	
313	K	K	
314	L	L	

Octal Code	ASCII Character	Key Combination	Remarks
315	M	M	
316	N	N	
317	O	O	
320	P	P	
321	Q	Q	
322	R	R	
323	S	S	
324	T	T	
325	U	U	
326	V	V	
327	W	W	
330	X	X	
331	Y	Y	
332	Z	Z	
333	[	SHIFT K	
334	← (\)	SHIFT L	
335	]	SHIFT M	
336	∧ (↑)	SHIFT ∧ (↑)	
337	_ (←)	SHIFT _ (←)	
340-373*			Lower case letters; codes cannot be generated from keyboard and should not be used in programs for reasons of compatability.
374*	ACK		Acknowledge; code cannot be generated from keyboard and should not be used in programs for reasons of compatability.
375*	ⓘ	ALT MODE	May be used for any desired control purpose.
376*	ESC		Escape; code cannot be generated from keyboard and should not be used in programs for reasons of compatability.

Octal Code	ASCII Character	Key Combination	Remarks
377*	DEL	RUB OUT	Delete
		REPT	Causes any other key that is struck to repeat continuously until REPT is released.
		LOC LF	Local line feed.
		LOC CR	Local carriage return.
		BRK RLS	Not connected.

APPENDIX 6

CARD READER CODE

6-bit Code	Character	Column Punch	6-bit Code	Character	Column Punch
00		Any invalid	40	-	11
01	1	1	41	J	11 1
02	2	2	42	K	11 2
03	3	3	43	L	11 3
04	4	4	44	M	11 4
05	5	5	45	N	11 5
06	6	6	46	O	11 6
07	7	7	47	P	11 7
10	8	8	50	Q	11 8
11	9	9	51	R	11 9
12	0	0	52		11 0
13	= [#]	8 3	53	$	11 8 3
14	' [@]	8 4	54	*	11 8 4
15		8 5	55		11 8 5
16		8 6	56		11 8 6
17		8 7	57		11 8 7
20	Space	None	60	+ [&]	12
21	/	0 1	61	A	12 1
22	S	0 2	62	B	12 2
23	T	0 3	63	C	12 3
24	U	0 4	64	D	12 4
25	V	0 5	65	E	12 5
26	W	0 6	66	F	12 6
27	X	0 7	67	G	12 7
30	Y	0 8	70	H	12 8
31	Z	0 9	71	I	12 9
32		0 8 2	72		12 0
33	,	0 8 3	73	.	12 8 3
34	([%]	0 8 4	74	) [□]	12 8 4
35		0 8 5	75		12 8 5
36		0 8 6	76		12 8 6
37		0 8 7	77		12 8 7

Invalid Punch Combinations

0 1 11	0 1 12	1 2	2 3	3 5	5 6
0 2 11	0 2 12	1 3	2 4	3 6	5 7
0 3 11	0 3 12	1 4	2 5	3 7	5 9
0 4 11	0 4 12	1 5	2 6	3 9	6 7
0 5 11	0 5 12	1 6	2 7	4 5	6 9
0 6 11	0 6 12	1 7	2 8*	4 6	7 9
0 7 11	0 7 12	1 8	2 9	4 7	8 9
0 8 11	0 8 12	1 9	3 4	4 9	11 12
0 9 11	0 9 12				

*except 2 8 0

digital
EQUIPMENT
CORPORATION
MAYNARD, MASSACHUSETTS

Cambridge, Mass. • Washington, D. C. • Parsippany, N.J. • Rochester, N.Y. • Los Angeles
Palo Alto • Chicago • Ann Arbor • Pittsburgh
Denver • Huntsville • Orlando • Carleton Place
and Toronto, Ont. • Reading, England • Paris,
France • Munich, Germany • Sydney, Australia

5526 PRINTED IN U.S.A. 5-9/65

F-67 (166)

ARITHMETIC PROCESSOR 166
INSTRUCTION MANUAL

VOLUME 2

PDP-6

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

F-67(166)

PDP-6 ARITHMETIC PROCESSOR 166
INSTRUCTION MANUAL

VOLUME 2

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

COPY NO.

ENGINEERING DRAWINGS

ENGINEERING DRAWINGS (continued)

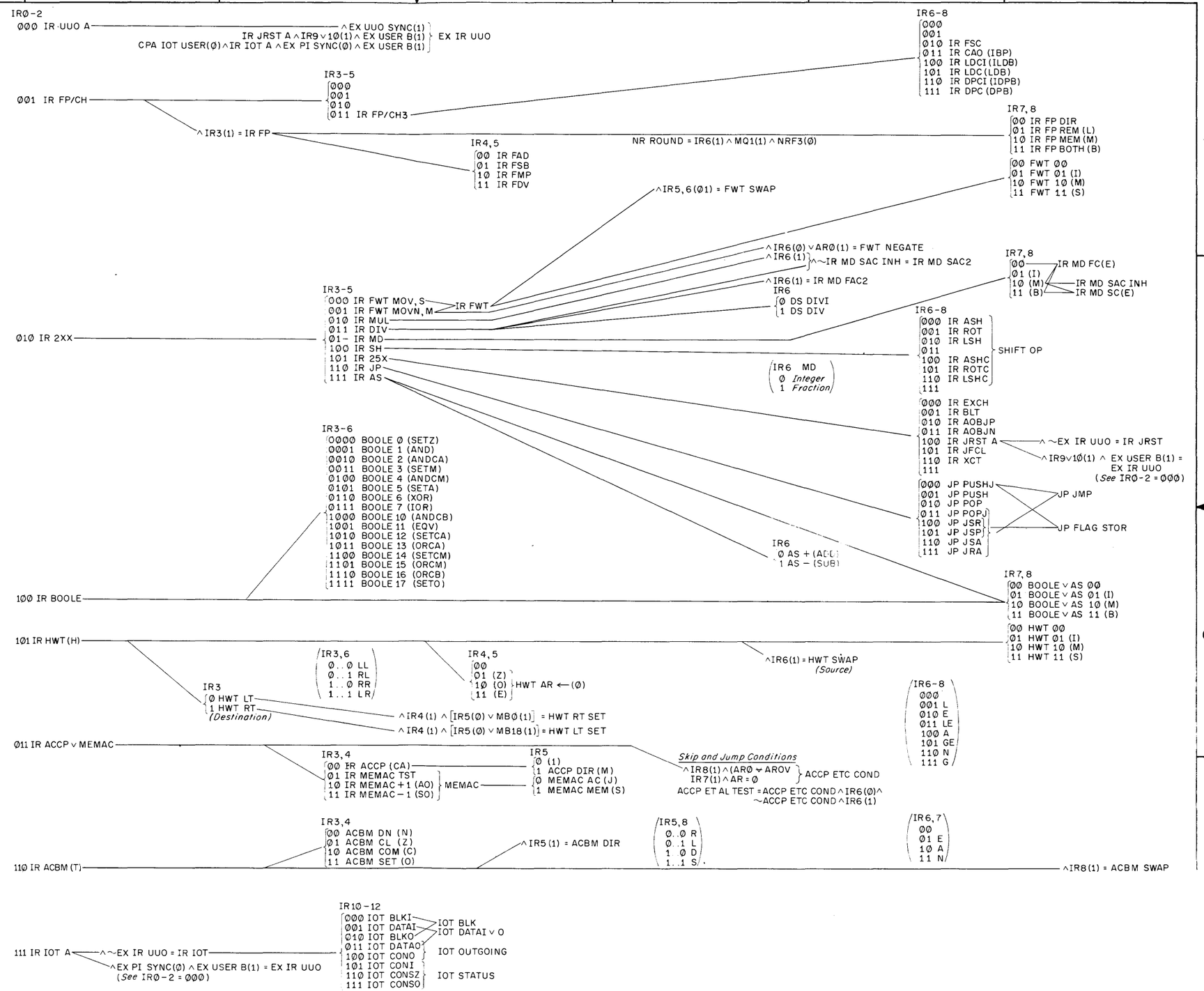
ENGINEERING DRAWINGS (continued)

System Block Diagram
(SD-D-166-0-BD)

System Block Diagram
(SD-D-166-0-BD)

LOGIC PREFIXES

A	5-3
ACBM	5-9
ACCP	5-9
AR	6-4,5,6,7,8,9,10
AS	5-10
BLT	6-18
BOOLE	5-10
CFAC	6-17
CH	6-19,20
CPA	8-5
CR	8-13,14,15
DC	6-20
DS	6-25,26
E	5-5
EX	5-13
F	5-4
FA	6-22
FD	6-22
FE	6-14,15
FM	6-22
FP	6-23
FS	6-19
FWT	5-9
HWT	5-9
I	5-3
IOB	8-1,2
IOT	8-1
IR	5-7,8
JP	5-10
K	5-2
KEY	5-1,2
LC	6-20
MA	7-1,2,3
MAI	7-2
MB	6-1,2,3,4;7-6
MC	7-8,9
MEMAC	5-10
MI	7-7
MP	6-21
MQ	6-7,11,12,13
MR	5-1,2
MS	6-24
NR	6-27
PC	5-11,12
PI	8-3,4
PR	7-4
PTP	8-9,10
PTR	8-6,7,8
RLA	7-5
RLR	7-5
RUN	5-1
S	5-6
SA	6-16
SC	6-14,15,16
SH	6-20
SHC	6-7
SW	5-1
TTI	8-11,12
TTO	8-11,12
TTY	8-11,12
UUO	5-10
XCT	5-10



Instruction Decoding
(FD-D-166-0-IRD)

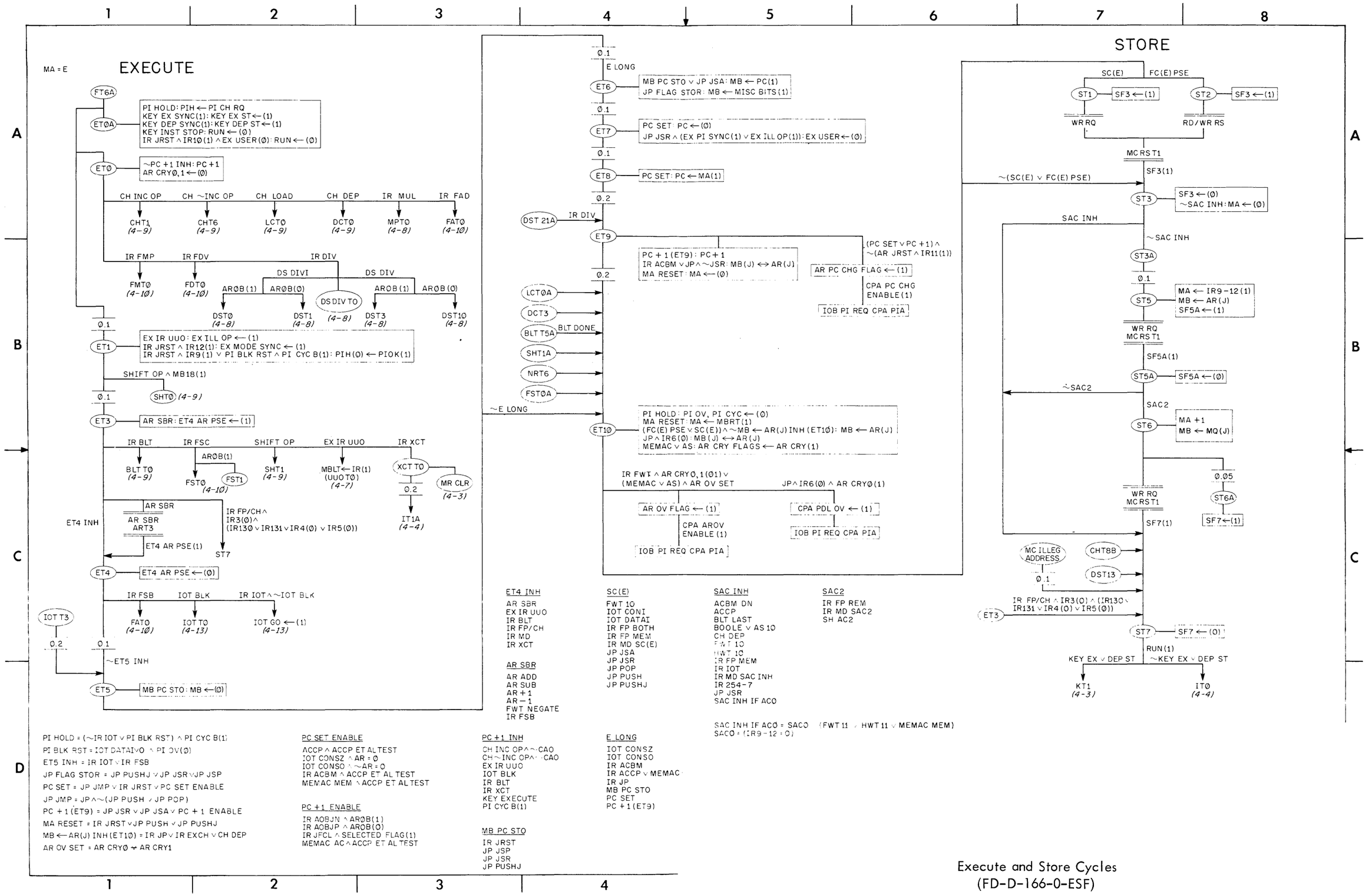
AP FIGURE 4-3

Key Cycle; Memory Subroutine
(FD-D-166-0-KMF)

Instruction, Address and Fetch Cycles
(FD-D-166-0-1AFF)

Instruction, Address and Fetch Cycles
(FD-D-166-0-1AFF)

Execute and Store Cycles
(FD-D-166-0-ESF)



Execute and Store Cycles
(FD-D-166-0-ESF)

HALF WORD TRANSFER
500–577

IR = 1Ø1 WXX YZZ
IR HWT = IRØ–2(101)
W *specifies destination half*
HWT LT = IR HWT ∧ IR3(Ø)
HWT RT = IR HWT ∧ IR3(1)
XX *specifies action on other half – do nothing, zero, one, extend*
HWT AR ← (Ø) = IR HWT ∧ [IR4(1) ∨ IR5(1)]
HWT RT SET = HWT LT ∧ IR4(1) ∧ [IR5(Ø) ∨ MBØ(1)]
HWT LT SET = HWT RT ∧ IR4(1) ∧ [IR5(Ø) ∨ MB18(1)]
Y *specifies source half*
HWT SWAP = IR HWT ∧ IR6(1)
ZZ *specifies mode*

IR HWT ∧ IR7,8		Action
ØØ	= HWT ØØ	✓[C(E), AC] → AC
Ø1	= HWT Ø1	✓[(Ø,E), AC] → AC
1Ø	= HWT 1Ø	✓[AC, C(E)] → E
11	= HWT 11	✓[C(E)] → E

Initial Registers
HWT ØØ: AR = AC MB = C(E)
HWT Ø1: AR = AC MB = Ø,E
HWT 1Ø: AR = AC MB = C(E)
HWT 11: AR = Ø,E MB = C(E)
MQ = Ø

Initial Gates
HWT ØØ: FC(E)
HWT 1Ø: FC(E) PSE
HWT 11: FAC INH, FC(E) PSE

ETØA
HWT 1Ø: MB(J) ↔ AR(J)
HWT 11: AR ← MB(J)

ET1
HWT SWAP: MB LT(J) ↔ MB RT(J)
HWT AR ← (Ø): AR ← (Ø)

ET3

ET4
HWT LT SET: AR LT COM
HWT RT SET: AR RT COM
HWT LT: AR ← MB LT(J)
HWT RT: AR ← MB RT(J)

ET5

ET10
HWT 1Ø ∨ 11: MB ← AR(J)

Final Gates
HWT 1Ø: FC(E) PSE, SAC INH
HWT 11: FC(E) PSE
HWT 11 SACØ: SAC INH
SACØ = (IR9–12 = Ø)

FULL WORD TRANSFER
200–217

IR = Ø1Ø ØØY YZZ
IR 2XX = IRØ–2(Ø1Ø)
YY *specifies instruction – MOVE, MOVS, MOVN, MOVM*
IR MOV, S = IR 2XX ∧ IR3–5(ØØØ)
IR MOVN, M = IR 2XX ∧ IR3–5(ØØ1)
IR FWT = IR MOV, S ∨ IR MOVN, M
FWT SWAP = IR FWT ∧ IR5,6 (Ø1)
FWT NEGATE = IR MOVN, M ∧ [IR6(Ø) ∨ ARØ(1)]
ZZ *specifies mode*

IR FWT ∧ IR7,8		Action
ØØ	= FWT ØØ	✓[C(E)] → AC
Ø1	= FWT Ø1	✓[Ø,E] → AC
1Ø	= FWT 1Ø	✓[AC] → E
11	= FWT 11	✓[C(E)] → E

Initial Registers
FWT ØØ: AR = Ø,E MB = C(E)
FWT Ø1: AR = Ø,E MB = ?,E
FWT 1Ø: AR = AC MB = Ø,E
FWT 11: AR = Ø,E MB = C(E)
MQ = Ø

Initial Gates
FWT ØØ: FAC INH, FC(E)
FWT Ø1: FAC INH
FWT 11: FAC INH, FC(E) PSE
FWT NEGATE: ET4 INH, AR SBR

ETØA
FWT ØØ ∨ 11: AR ← MB(J)
FWT Ø1 ∨ 1Ø: MB ← AR(J)

ET1
FWT SWAP: MB LT(J) ↔ MB RT(J)

ET3
FWT NEGATE: ET4 AR PSE ← (1) | AR NEGATE ART3 |

ET4
ET4 AR PSE ← (Ø)
FWT SWAP: MB(J) ↔ AR(J)

ET5

ET10
AR CRYØ,1(Ø1): AR OV FLAG ← (1)
FWT 1Ø ∨ 11: MB ← AR(J)

Final Gates
FWT 1Ø: SC(E), SAC INH
FWT 11: FC(E) PSE
FWT 11 ∧ SACØ: SAC INH
SACØ = (IR9–12 = Ø)

EXCH
250

IR = Ø1Ø 1Ø1 ØØØ
IR 2XX = IRØ–2(Ø1Ø)
IR 25X = IR 2XX ∧ IR3–5(1Ø1)
IR EXCH = IR 25X ∧ IR6–8(ØØØ)

AC ↔ C(E)

Initial Registers
AR = AC
MB = C(E)
MQ = Ø

Initial Gates
FC(E) PSE
MB ← AR(J) INH(ET1Ø)

ETØA
MB(J) ↔ AR(J)

ET1

ET3

ET4
AR PSE ← (Ø)

ET5

ET10

Final Gates
FC(E) PSE

ARITHMETIC COMPARE
300–377

IR = Ø11 VVW XYZ
IR ACCP ∨ MEMAC = IRØ–2(Ø11)
VV *specifies instruction type*

(IR ACCP ∨ MEMAC) ∧ IR3,4	
ØØ	= ACCP
Ø1	= MEMAC TST
1Ø	= MEMAC +1
11	= MEMAC –1

MEMAC = MEMAC TST ∨ MEMAC +1 ∨ MEMAC –1
X *specifies whether or not action is on condition YZ*
Y *specifies condition 'equals'*
Z *specifies condition 'less than'*
XYZ *together define relation* R

XYZ	R
ØØØ	*Never*
ØØ1	<
Ø1Ø	=
Ø11	≤
1ØØ	*Always*
1Ø1	≥
11Ø	≠
111	>

LOGICAL COMPARE
6ØØ–677

IR = 11Ø VVW XYZ
IR ACBM = IRØ–2(11Ø)
VV *specifies action on masked bits*

IR ACBM ∧ IR3,4	
ØØ	= ACBM DN
Ø1	= ACBM CL
1Ø	= ACBM COM
11	= ACBM SET

W *specifies source of mask – immediate or memory*
ACBM DIR = IR ACBM ∧ IR5(1)
X *specifies whether or not action is on condition Y, i.e 'equals zero'*
XY *together define relation* R

XY	R
ØØ	*Never*
Ø1	= Ø
1Ø	*Always*
11	≠ Ø

Z *specifies swapping mask selected by* W
ACBM SWAP = IR ACBM ∧ IR8(1)
WZ *together specify mask as* (Ø,E), (E,Ø), C(E), C(E) RT LT

AOBJP 252 / AOBJN 253

IR = Ø1Ø 1Ø1 Ø1X
IR 2XX = IRØ–2(Ø1Ø)
IR 25X = IR 2XX ∧ IR3–5(1Ø1)
X *specifies instruction*
IR AOBJP = IR 25X ∧ IR6–8(Ø1Ø)
IR AOBJN = IR 25X ∧ IR6–8(Ø11)

AC + 1ØØØØØ1 → AC
AOBJP ∧ (AC ≥ Ø): E → PC
AOBJN ∧ (AC < Ø): E → PC

IR AOBJP ∧ ARØB(Ø): PC SET
IR AOBJN ∧ ARØB(1): PC SET

Determination of Skip or Jump Condition
ACCP ETC COND = [IR7(1) ∧ (AR = Ø)] ∨ [(IR ACCP ∨ MEMAC) ∧ IR8(1) ∧ (ARØ ≠ AR OV)]
ACCP ET AL TEST = ACCP ETC COND ≠ IR6
ARØ ≠ AR OV = ARØ ≠ (AR OV SET ∧ ∼MEMAC)
AR OV SET = AR CRYØ ≠ AR CRY1

MEMAC
320–377

W *specifies whether* AC *or* C(E) *compared with zero, i.e whether action is jump or skip*
MEMAC AC = MEMAC ∧ IR5(Ø)
MEMAC MEM = MEMAC ∧ IR5(1)

MEMAC AC	MEMAC MEM
AC R Ø: E → PC	C(E) R Ø: PC + 1
MEMAC ±1: AC ±1 → AC	MEMAC ±1: C(E) ±1 → E
ACCP ET AL TEST: PC SET	∼ACØ ∧ MEMAC ±1: C(E) ±1 → AC
	SACØ = (IR9–12 = Ø)

ACCP
300–317

W *specifies whether* AC *compared with* E *or* C(E)
ACCP DIR = ACCP ∧ IR5(1)
IR5(Ø) ∧ [AC R (Ø,E)]: PC + 1
IR5(1) ∧ [AC R C(E)]: PC + 1

ACBM

R [*Mask* ∧ AC]: PC + 1
✓ [*Mask*, AC] → AC

Initial Registers
AR = AC
MB = Ø,E
MQ = Ø

AR = Ø,E
MB = C(E)
MQ = Ø

AR = AC
∼ACCP DIR: MB = Ø,E
ACCP DIR: MB = C(E)
MQ = Ø

AR = AC
∼ACBM DIR: MB = Ø,E
ACBM DIR: MB = C(E)
MQ = Ø

AR = AC
MB = Ø,E
MQ = Ø

Initial Gates
MEMAC ±1: ET4 INH
AR SBR
E LONG

FAC INH
FC(E) PSE
MEMAC ±1: ET4 INH
AR SBR
E LONG

ACCP DIR: FC(E)
ET4 INH
AR SBR
E LONG

ACBM DIR: FC(E)
E LONG

ET4 INH
AR SBR
E LONG

ETØA
AR ← MB(J)
ACBM SWAP: MB LT(J) ↔ MB RT(J)

ET1
MB ← AR(Ø)
ACBM COM: AR ← MB(∼)
ACBM SET: AR ← MB(1)

ET3
MEMAC +1: ET4 AR PSE ← (1) | AR +1 ART3 |
MEMAC –1: ET4 AR PSE ← (1) | AR –1 ART3 |
ET4 AR PSE ← (1) | AR SUB ART3 |
ET4 AR PSE ← (1) | AR +1 LT RT ART3 |

ET4
ET4 AR PSE ← (Ø)
ET4 AR PSE ← (Ø)
ET4 AR PSE ← (Ø)
MB(J) ↔ AR(J)
ET4 AR PSE ← (Ø)

ET5
AR COM

ET6
ACBM CL: MB ← AR(Ø)

ET7
PC SET: PC ← (Ø)
AR COM
PC SET: PC ← (Ø)

ET8
PC SET: PC ← MA(1)
PC SET: PC ← MA(1)

ET9
PC SET: AR PC CHG FLAG ← (1)
ACCP ET AL TEST: PC + 1 AR PC CHG FLAG ← (1)
ACCP ET AL TEST: PC + 1 AR PC CHG FLAG ← (1)
MB(J) ↔ AR(J) ACCP ET AL TEST: PC + 1 AR PC CHG FLAG ← (1)
PC SET: AR PC CHG FLAG ← (1)

ET10
AR OV SET: AR OV FLAG ← (1)
AR CRY FLAGS ← AR CRY(1)
AR OV SET: AR OV FLAG ← (1)
AR CRY FLAGS ← AR CRY(1)
MB ← AR(J)

Final Gates
FC(E) PSE
SACØ: SAC INH
SAC INH
ACBM DN: SAC INH

JUMP AND PUSHDOWN

260-267

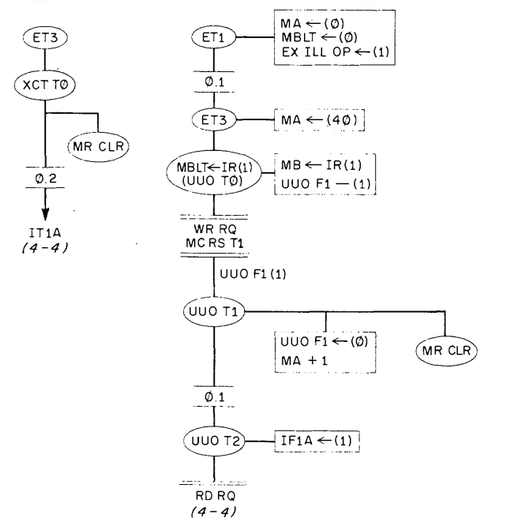
IR = 010 110 XXX
IR 2XX = IR0-2(010)
IR JP = IR 2XX ∧ IR3-5(110)
XXX *specifies instruction*

IR JP ∧ IR6-8:
000 = JP PUSHJ
001 = JP PUSH
010 = JP POP
011 = JP POPJ
100 = JP JSR
101 = JP JSP
110 = JP JSA
111 = JP JRA

JP JMP = IR JP ∧ ~(JP PUSH ∨ JP POP)
JP FLAG STOR = JP PUSHJ ∨ JP JSR ∨ JP JSP
~IR IOT ∧ PI CYC B(1): PI HOLD

	PUSHJ 260	PUSH 261	POP 262	POPJ 263	JSR 264	JSP 265	JSA 266	JRA 267
	AC+1000001→AC MISC BITS, PC→ C(C(ACRT)) E→PC PI ON OVERFLOW	AC+1000001→AC C(E)→C(C(ACRT)) PI ON OVERFLOW	C(C(ACRT))→E AC-1000001→AC PI ON UNDERFLOW	C(C(ACRT))→PC AC-1000001→AC PI ON UNDERFLOW	MISC BITS, PC→E E+1→PC	MISC BITS, PC→AC E→PC	AC→E E, PC→AC E+1→PC	C(C(ACLT))→AC E→PC
Initial Registers	AR = AC MB = Ø, E MQ = Ø	AR = AC MB = C(E) MQ = Ø	AR = AC MB = Ø, E MQ = C(C(ACRT))	AR = AC MB = Ø, E MQ = C(C(ACRT))	AR = Ø, E MB = ?, E MQ = 0	AR = Ø, E MB = ?, E MQ = Ø	AR = AC MB = Ø, E MQ = Ø	AR = AC RTLT MB = Ø, E MQ = C(C(ACLT))
Initial Gates	ET4 INH AR SBR E LONG MB ← AR(J) INH (ET1Ø)	FC(E) ET4 INH AR SBR E LONG MB ← AR(J) INH (ET1Ø)	FC(C(ACRT)) ET4 INH AR SBR E LONG MB ← AR(J) INH (ET1Ø)	FC(C(ACRT)) ET4 INH AR SBR E LONG MB ← AR(J) INH (ET1Ø)	FAC INH E LONG MB ← AR(J) INH (ET1Ø)	FAC INH E LONG MB ← AR(J) INH (ET1Ø)	E LONG MB ← AR(J) INH (ET1Ø)	FC(C(ACLT)) E LONG MB ← AR(J) INH (ET1Ø)
ETØA			MB ← MQ(J)	MB ← MQ(J)	PI HOLD: PIH ← PI CH RQ	MB(J)↔AR(J)	MBLT(J)↔MBRT(J)	MB ← MQ(J)
ET1				MA ← (Ø)				
ET3	ET4 AR PSE ← (1) ‖AR + 1 LTRT ART3‖	ET4 AR PSE ← (1) ‖AR + 1 LTRT ART3‖	ET4 AR PSE ← (1) ‖AR - 1 LTRT ART3‖	ET4 AR PSE ← (1) ‖AR - 1 LTRT ART3‖				
ET4	ET4 AR PSE ← (Ø)	ET4 AR PSE ← (Ø)	ET4 AR PSE ← (Ø)	ET4 AR PSE ← (Ø)				
ET5	MB ← (Ø)				MB ← (Ø)	MB ← (Ø)		
ET6	MB ← MISC BITS, PC(1)				MB ← MISC BITS, PC(1)	MB ← MISC BITS, PC(1)	MB ← PC(1)	
ET7	PC ← (Ø)			PC ← (Ø)	PC ← (Ø) EX PI SYNC(1) ∨ EX ILL OP(1): EX USER ← (Ø)	PC ← (Ø)	PC ← (Ø)	PC ← (Ø)
ET8	PC ← MA(1)			PC ← MA(1)	PC ← MA(1) EX ILL OP ← (Ø)	PC ← MA(1)	PC ← MA(1)	PC ← MA(1)
ET9	MB(J)↔AR(J) MA ← (Ø)	MB(J)↔AR(J) MA ← (Ø)	MB(J)↔AR(J)	MB(J)↔AR(J)	PC + 1 CHF7 ← (Ø)	MB(J) ← AR(J)	MB(J)↔AR(J) PC + 1	MB(J)↔AR(J)
ET10	MA ← MBRT(1) MB(J)↔AR(J) AR CRYØ(1): CPA PDL OV ← (1) IOB PI REQ CPA PIA		MB(J)↔AR(J) AR CRYØ(1): CPA PDL OV ← (1) IOB PI REQ CPA PIA		PI HOLD: PI OV, PI CYC ← (Ø)			
Final Gates	SC(E)	SC(E)	SC(E)		SC(E) SAC INH		SC(E)	

	JRST 254	JFCL 255	XCT 256	UUO 000-077
	IR = 010 101 100 IR 2XX = IR0-2(010) IR 25X = IR2XX ∧ IR3-5(101) IR JRST A = IR 25X ∧ IR 6-8(100) IR JRST = IR JRST A ∧ ~EX IR UUO	IR = 010 101 101 IR 2XX = IR0-2(010) IR 25X = IR 2XX ∧ IR3-5(101) IR JFCL = IR 25X ∧ IR 6-8(010)	IR = 010 101 110 IR 2XX = IR0-2(010) IR 25X = IR 2XX ∧ IR3-5(101) IR XCT = IR 25X ∧ IR6-8(110) EXECUTE C(E)	IR = 000 XXX XXX IR UUO A = IR0-2(000) EX IR UUO = IR UUO A ∧ EX UUO SYNC(1) ∨ IR JRST A ∧ EX USER B(1) ∧ IR9 ∨ 10(1) ∨ IR IOT A ∧ EX USER B(1) ∧ EX PI SYNC(Ø) ∧ CPA IOT USER(Ø) IR, E → 4Ø EXECUTE 41
	E → PC IR9(1): DISMISS PI IR10(1): HALT IR11(1): RESTORE MISC BITS IR12(1): ENTER USER MODE	SELECTED FLAG(1): E → PC CLEAR SELECTED FLAGS		
Initial Registers	AR = Ø, E MB = ?, E MQ = 0	AR = Ø, E MB = ?, E MQ = Ø	AR = Ø, E MB = C(E) MQ = Ø	AR = Ø, E MB = ?, E MQ = 0
Initial Gates	FAC INH E LONG	FAC INH E LONG	FAC INH FC(E) PC + 1 INH ET4 INH	FAC INH PC + 1 INH ET4 INH
ETØA	IR1Ø(1): RUN ← (0) IR11(1): AR FLAG CLEAR (*Clears* MISC BITS *at* ETØ)			
ET1	IR9(1): PIH(Ø) ← PIOK(1) IR11(1): AR FLAG SET (*Transfers* MB(1) *into* MISC BITS) IR12(1): EX MODE SYNC ← (1)			
ET3				
ET4				
ET5	MB ← (Ø)			
ET6	MB ← PC(1)			
ET7	PC ← (Ø)	SELECTED FLAG(1): PC ← (Ø)		
ET8	PC ← MA(1)	SELECTED FLAG(1): PC ← MA(1)		
ET9	MA ← (Ø)			
ET10	MA ← MBRT(1)	SELECTED FLAGS ← (Ø)		
Final Gates	SAC INH	SAC INH		



MISC BITS
0 AR OV FLAG
1 AR CRYØ FLAG
2 AR CRY1 FLAG
3 AR PC CHG FLAG
4 CHF7
5 EX USER - STORE ONLY
5 EX MODE SYNC - SET ONLY

SELECTED FLAGS
IR9(1) AR OV FLAG
IR1Ø(1) AR CRYØ FLAG
IR11(1) AR CRY1 FLAG
IR12(1) AR PC CHG FLAG

Pushdown and Jump Instructions
(FD-D-166-0-JPF)

BOOLEAN

400-477

IR = 100 XXX XYY
IR BOOLE = IR0-2(100)
XXXX *specifies Boolean function, decoded into* BOOLE 0-17 *as*
BOOLE N = IR BOOLE ∧ IR3-6(N)

In both instruction groups YY *specifies common mode*

(IR BOOLE ∨ IR AS) ∧ IR7,8 { 00 = BOOLE ∨ AS 00 ; 01 = BOOLE ∨ AS 01 ; 10 = BOOLE ∨ AS 10 ; 11 = BOOLE ∨ AS 11 }

Action

	BOOLE	AS
BOOLE ∨ AS 00:	f[AC, C(E)] ⟶ AC	AC ± C(E) ⟶ AC
BOOLE ∨ AS 01:	f[AC, (0,E)] ⟶ AC	AC ± (0,E) ⟶ AC
BOOLE ∨ AS 10:	f[AC, C(E)] ⟶ E	AC ± C(E) ⟶ E
BOOLE ∨ AS 11:	f[AC, C(E)] ⟶ AC, E	AC ± C(E) ⟶ AC, E

Initial Registers

AR = AC
BOOLE ∨ AS 00 ∨ 10 ∨ 11: MB = C(E)
BOOLE ∨ AS 01: MB = (0,E)
MQ = 0

Common Initial Gates

BOOLE ∨ AS 00: FC(E)
BOOLE ∨ AS 10 11: FC(E) PSE

Initial Gates AS *Only*

ET4 INH
AR SBR

ET0A	BOOLE 0,3,14,17: AR ⟵ (0) BOOLE 2,4,12,13,15: AR ⟵ COM	
ET1	BOOLE 6,11,14: AR ⟵ MB(∨) BOOLE 1,2,15,16: AR ⟵ MB(0) BOOLE 3,4,7,10,13: AR ⟵ MB(1)	
ET3		ET4 AR PSE ⟵ (1) AS+: AR ADD AS−: AR SUB ART3
ET4	BOOLE 4,10,11,14,15,16,17: AR ⟵ COM	ET4 AR PSE ⟵ (0)
ET5		
ET10	FC(E) PSE: MB ⟵ AR(J)	FC(E) PSE: MB ⟵ AR(J) AR OV SET: AR OV FLAG ⟵ (1) AR CRY FLAGS ⟵ AR CRY(1) AR OV SET = AR CRY0 ∨ AR CRY1

Final Gates

BOOLE ∨ AS 10: FC(E) PSE, SAC INH
BOOLE ∨ AS 11: FC(E) PSE

BOOLEAN FUNCTIONS

	0 SETZ	1 AND	2 ANDCA	3 SETM
ETOA	AR ⟵ (0)		AR ⟵ COM	AR ⟵ (0)
ET1		AR ⟵ MB(0)	AR ⟵ MB(0)	AR ⟵ MB(1)
ET4				

	4 ANDCM	5 SETA	6 XOR	7 IOR
ETOA	AR ⟵ COM			
ET1	AR ⟵ MB(1)		AR ⟵ MB(∨)	AR ⟵ MB(1)
ET4	AR ⟵ COM			

	10 ANDCB	11 EQV	12 SETCA	13 ORCA
ETOA			AR ⟵ COM	AR ⟵ COM
ET1	AR ⟵ MB(1)	AR ⟵ MB(∨)		AR ⟵ MB(1)
ET4	AR ⟵ COM	AR ⟵ COM		

	14 SETCM	15 ORCM	16 ORCB	17 SETO
ETOA	AR ⟵ (0)	AR ⟵ COM		AR ⟵ (0)
ET1	AR ⟵ MB(∨)	AR ⟵ MB(0)	AR ⟵ MB(0)	
ET4	AR ⟵ COM	AR ⟵ COM	AR ⟵ COM	AR ⟵ COM

ADD-SUBTRACT

270-277

IR = 010 111 XYY
IR 2XX = IR0-2(010)
IR AS = IR 2XX ∧ IR3-5(111)
X *specifies add or subtract*

IR AS ∧ IR6 { 0 = AS + ; 1 = AS − }

MULTIPLY-DIVIDE

220-237

IR = 010 01X YZZ
IR 2XX = IR0-2(010)
X *specifies operation*

IR 2XX ∧ IR3-5 { 010 = IR MUL ; 011 = IR DIV }

Y *specifies operand type:* IR6 *is gate in multiply flow; for divide*
DS DIVI = IR DIV ∧ IR6(0)
DS DIV = IR DIV ∧ IR6(1)

ZZ *specifies mode*

Levels for fetch and store according to operand type and mode

IR MD = IR 2XX ∧ IR3(0) ∧ IR4(1)
IR MD FC(E) = IR MD ∧ ~(IR7(0) ∧ IR8(1))
IR MD FAC2 = IR DIV ∧ IR6(1)
IR MD SC(E) = IR MD ∧ IR7(1)
IR MD SAC INH = IR MD ∧ IR7(1) ∧ IR8(0)
IR MD SAC2 = ~IR MD SAC INH ∧ (IR DIV ∨ IR MUL ∧ IR6(1))

IR MUL

Action

For IMUL(XY = 00) *store only low order half of product*
ZZ = 00: AC × C(E) ⟶ AC
ZZ = 01: AC × (0,E) ⟶ AC
ZZ = 10: AC × C(E) ⟶ E
ZZ = 11: AC × C(E) ⟶ AC; E

For MUL(XY = 01) *store entire product or high order half*
ZZ = 00: AC × C(E) ⟶ AC, AC2
ZZ = 01: AC × (0,E) ⟶ AC, AC2
ZZ = 10: AC × C(E) ⟶ E
ZZ = 11: AC × C(E) ⟶ AC, AC2; E

Initial Registers

AR = AC
ZZ = 00 ∨ 10 ∨ 11: MB = C(E)
ZZ = 01: MB = 0,E
MQ = 0

Initial Gates

ZZ = 00 ∨ 10 ∨ 11: FC(E)
ET4 INH

IDIV (XY = 10)

Action

ZZ = 00: AC ÷ C(E) ⟶ AC, AC2
ZZ = 01: AC ÷ (0,E) ⟶ AC, AC2
ZZ = 10: AC ÷ C(E) ⟶ E
ZZ = 11: AC ÷ C(E) ⟶ AC, AC2; E

Initial Registers

AR = AC
ZZ = 00 ∨ 10 ∨ 11: MB = C(E)
ZZ = 10: MB = 0,E
MQ = 0

Initial Gates

ZZ = 00 ∨ 10 ∨ 11: FC(E)
ET4 INH

DIV (XY = 11)

Action

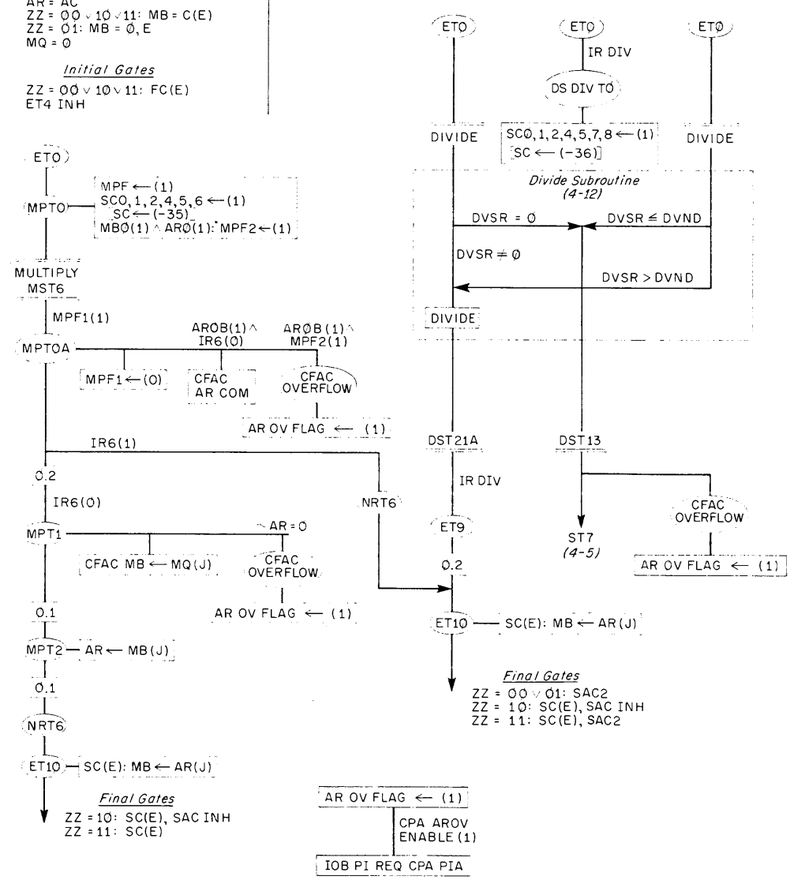
ZZ = 00: AC, AC2 ÷ C(E) ⟶ AC, AC2
ZZ = 01: AC, AC2 ÷ (0,E) ⟶ AC, AC2
ZZ = 10: AC, AC2 ÷ C(E) ⟶ E
ZZ = 11: AC, AC2 ÷ C(E) ⟶ AC, AC2; E

Initial Registers

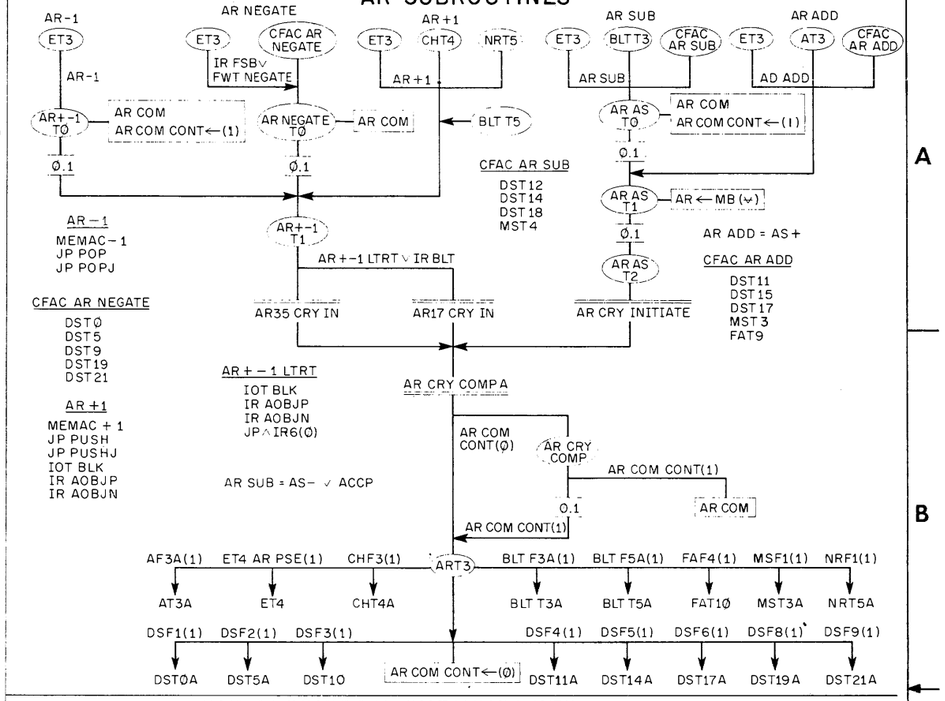
AR = AC
ZZ = 00 ∨ 10 ∨ 11: MB = C(E)
ZZ = 10: MB = 0,E
MQ = AC2

Initial Gates

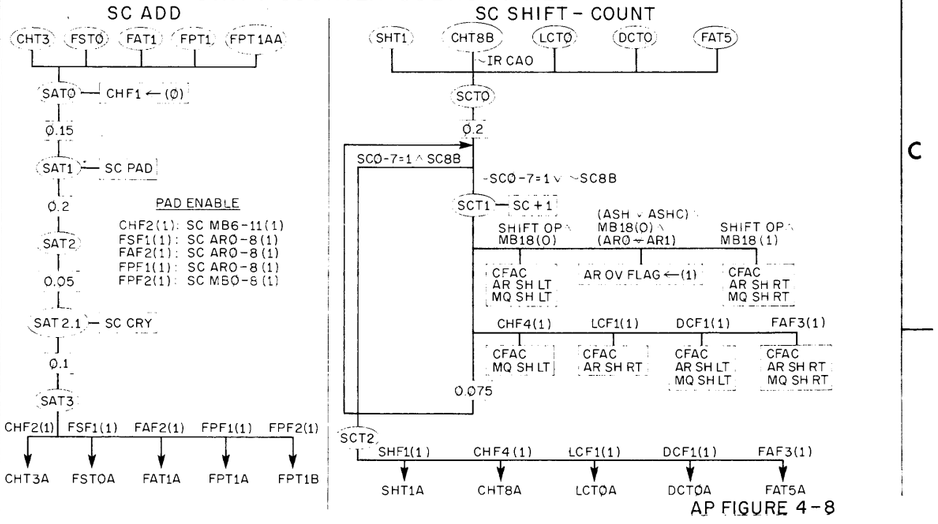
ZZ = 00 ∨ 10 ∨ 11: FC(E)
FAC2
ET4 INH



AR SUBROUTINES



SHIFT COUNTER SUBROUTINES



AP FIGURE 4-8

Fixed Point and Logical Instructions; AR and SC Subroutines
(FD-D-166-0-ALF)

CHARACTER OPERATIONS

133-137

IR = 001 011 XXX
IR FP/CH = IR0-2(001)
IR FP/CH3 = IR FP/CH ∧ IR3-5(011)
XXX *specifies instruction*

IR FP/CH3 ∧ IR6-8 =
- 011 = IR CAO
- 100 = IR LDC1
- 101 = IR LDC
- 110 = IR DPC1
- 111 = IR DPC

FIRST PART-CHF5(0)

CH INC = (IR LDCI ∨ IR DPCI ∨ IR CAO) ∧ CHF5(0)
CH INC OP = CH INC ∧ CHF7(0)
CH ~INC OP = [(IR LDC ∨ IR DPC) ∧ CHF5(0)] ∨ [CH INC ∧ CHF7(1)]

Pointer Format

P	S		I	X	Y
0-5	6-11	12	13	14-17	18-35

Fetch and increment pointer

Initial Registers
AR = 0,E
MB = C(E)
MQ = 0

Initial Gates
CH INC OP: FAC INH, FC(E) PSE
CH ~INC OP: FAC INH, FC(E)
~IR CAO: PC+1 INH
ET4 INH

SECOND PART-CHF5(1)

CH LOAD = (IR LDC ∨ IR LDCI) ∧ CHF5(1)
CH DEP = (IR DPC ∨ IR DPCI) ∧ CHF5(1)

CH LOAD
Fetch and load character

CH DEP
Fetch and deposit character

Initial Registers

AR = 0,E
MB = C(E)
MQ = 0...0 1...1 (S)

AR = 0,E
MB = C(E)
MQ = 0...0 1...1 (S)

Initial Gates

FAC INH
FC(E)
ET4 INH

FC(E) PSE
ET4 INH
MB ← AR(J) INH(ET0)

BLT

251

IR = 010 101 001
IR 2XX = IR0-2(010)
IR 25X = IR 2XX ∧ IR3-5(101)
IR BLT = IR 25X ∧ IR6-8(001)
BLT DONE = MQ0(0) ∨ PI RQ
BLT LAST = IR BLT ∧ MQ0(0)

S = *Source address*
D = *Destination address*
C(S) → D
(S,D) + 1000001 → (S,D)
Repeat until D = E

Initial Registers
AR = D,S *initially* AC RT LT
MB = 0,E
MQ = C(S) *initially* C(C(ACLT))

Initial Gates
FC(C(ACLT))
PC+1 INH
ET4 INH

SHIFT OPERATIONS

240-242, 244-246

IR = 010 100 XXX
IR 2XX = IR0-2(010)
IR SH = IR 2XX ∧ IR3-5(100)

IR SH ∧ IR6-8 =
- 000 = IR ASH
- 001 = IR ROT
- 010 = IR LSH
- 100 = IR ASHC
- 101 = IR ROTC
- 110 = IR LSHC

SHIFT OP = IR ASH ∨ IR ROT ∨ IR LSH ∨ IR ASHC ∨ IR ROTC ∨ IR LSHC
SH AC2 = IR ASHC ∨ IR ROTC ∨ IR LSHC
Shift E *mod* 256 *places*
~SH AC2: *f* [AC] → AC
SH AC2: *f* [AC, AC2] → AC

SHIFT OP AR, MQ CONNECTIONS
IR ASHC: SHC ASH

Initial Registers
AR = AC
MB = 0,E
~SH AC2: MQ = 0
SH AC2: MQ = AC2

Initial Gates
SH AC2: FAC 2
ET4 INH

Final Gate
SHAC2: SAC2

FLOATING POINT

140-177

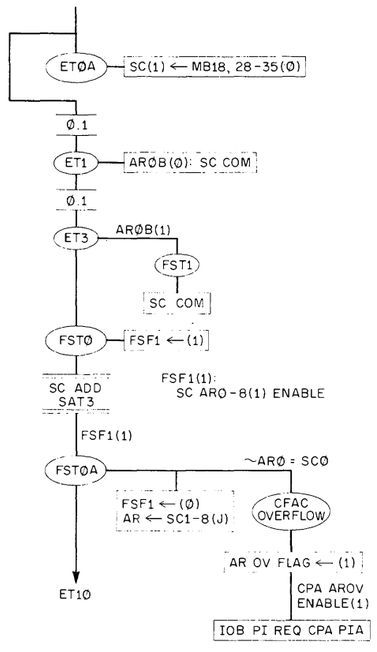
FSC

132

IR = ØØ1 Ø11 Ø1Ø
IR FP/CH = IRØ-2(ØØ1)
IR FP/CH3 = IR FP/CH ∧ IR3-5(Ø11)
IR FSC = IR FP/CH3 ∧ IR6-8(Ø1Ø)

AC × 2^E mod 256 → AC

Initial Registers: AR = AC; MB = C(E); MQ = Ø
Initial Gate: ET4 INH



IR = ØØ1 1XX YZZ
IR FP/CH = IRØ-2(ØØ1)
IR FP = IR FP/CH ∧ IR3(1)
Y specifies rounding, see Figure 4-11

XX specifies instruction
IR FP ∧ IR4,5 { ØØ = IR FAD; Ø1 = IR FSB; 1Ø = IR FMP; 11 = IR FDV }

ZZ specifies mode

IR FP ∧ IR7,8		*Action*
ØØ	= IR FP DIR	AC R C(E) → AC
Ø1	= IR FP REM	AC R C(E) → AC, AC2
1Ø	= IR FP MEM	AC R C(E) → E
11	= IR FP BOTH	AC R C(E) → E, AC

Operator R = +, −, ×, ÷

FAD

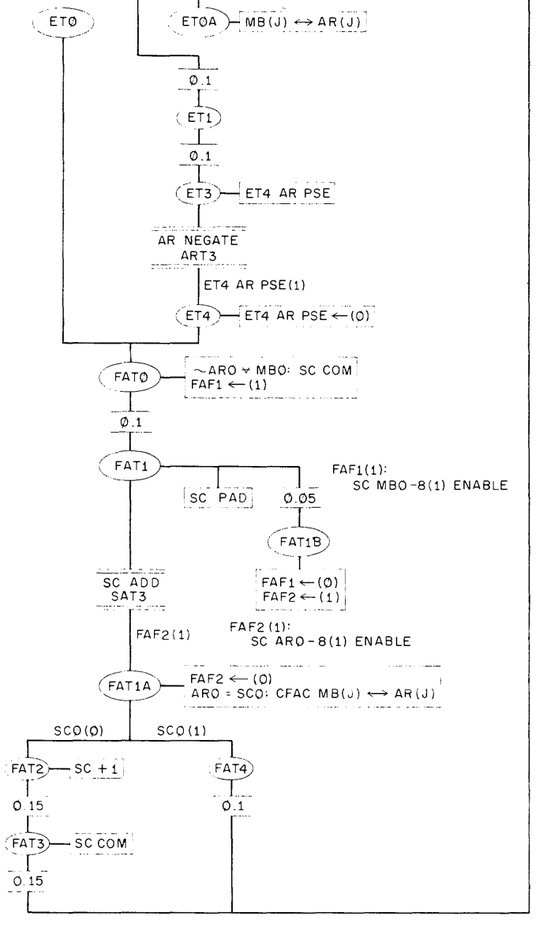
140-147

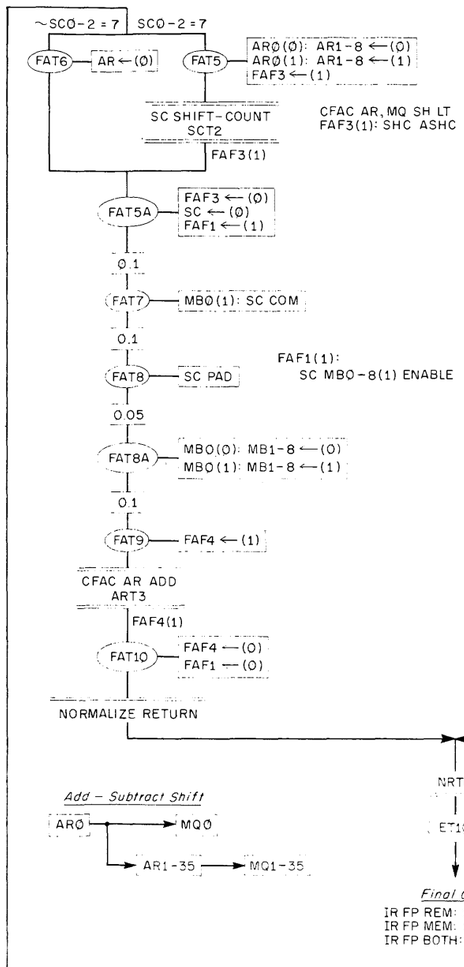
Initial Gates: FC(E); ET4 INH

FSB

150-157

Initial Gates: FC(E); ET4 INH; AR SBR; ET5 INH



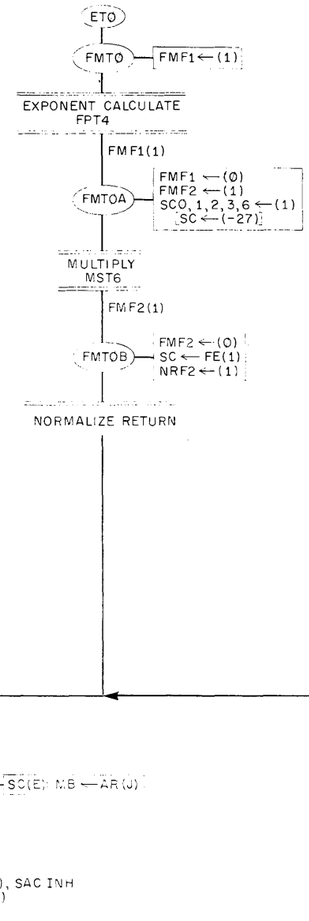


Initial Registers: AR = AC; MB = C(E); MQ = Ø

FMP

160-167

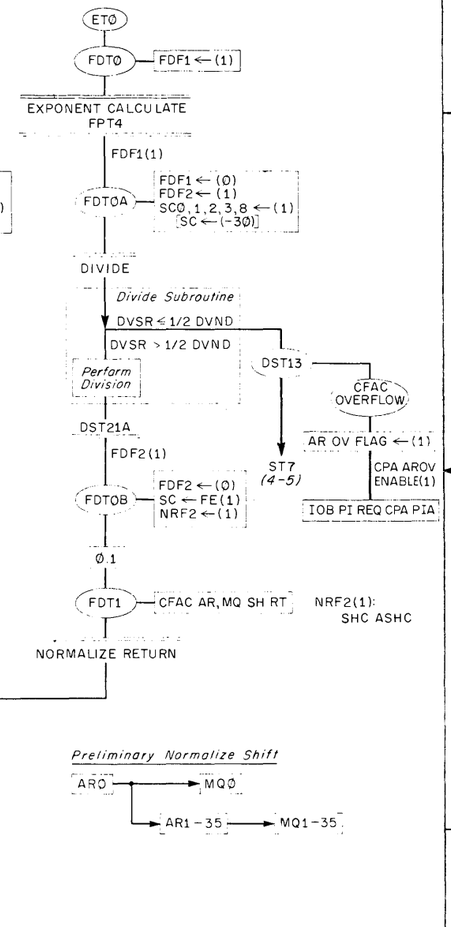
Initial Gates: FC(E); ET4 INH



FDV

170-177

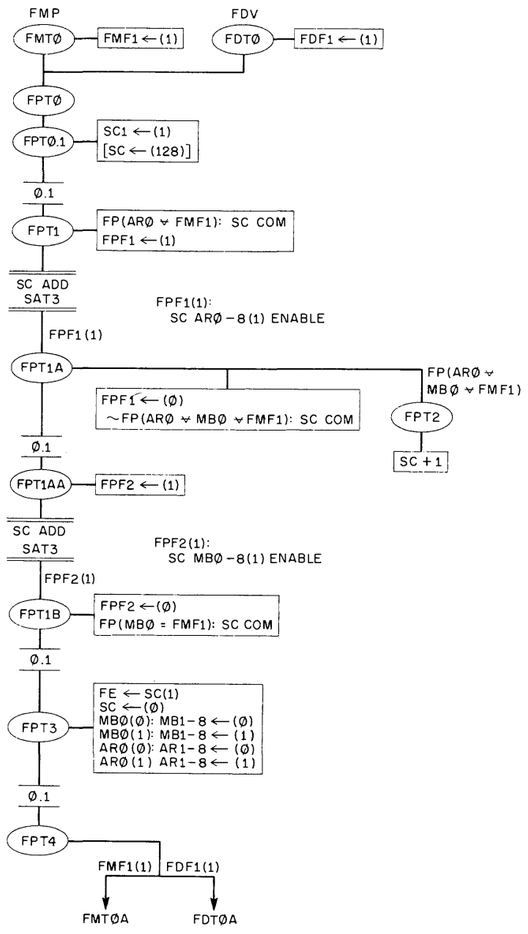
Initial Gates: FC(E); ET4 INH



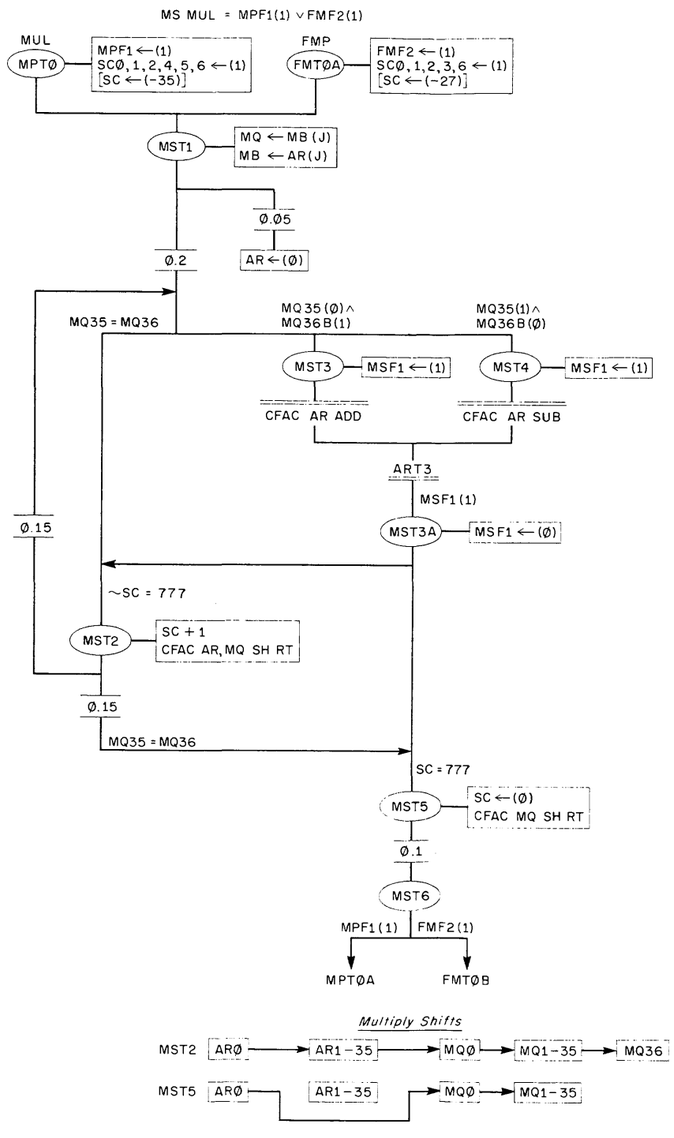
Exponent Calculate, Multiply and Normalize Return Subroutines
(FD-D-166-0-EMNF)

CODE FD | DRWG NO D-166-0-EMNF | REV LTR

EXPONENT CALCULATE

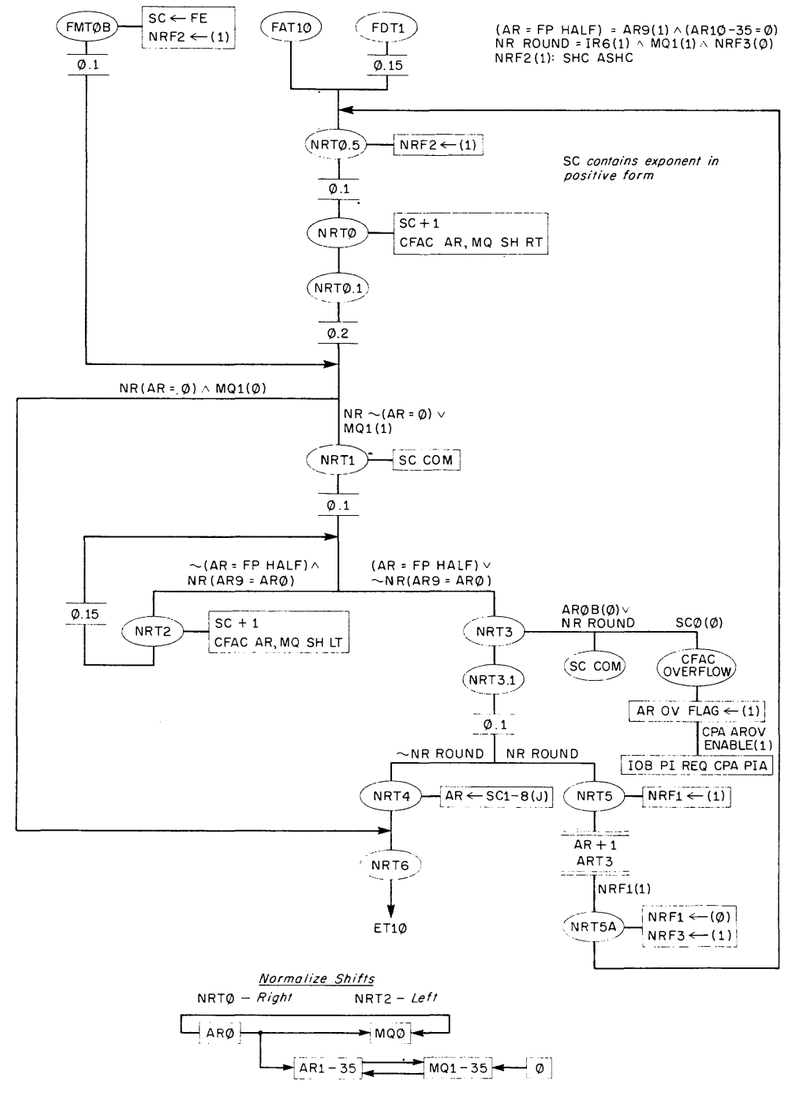


MULTIPLY



Multiply Shifts

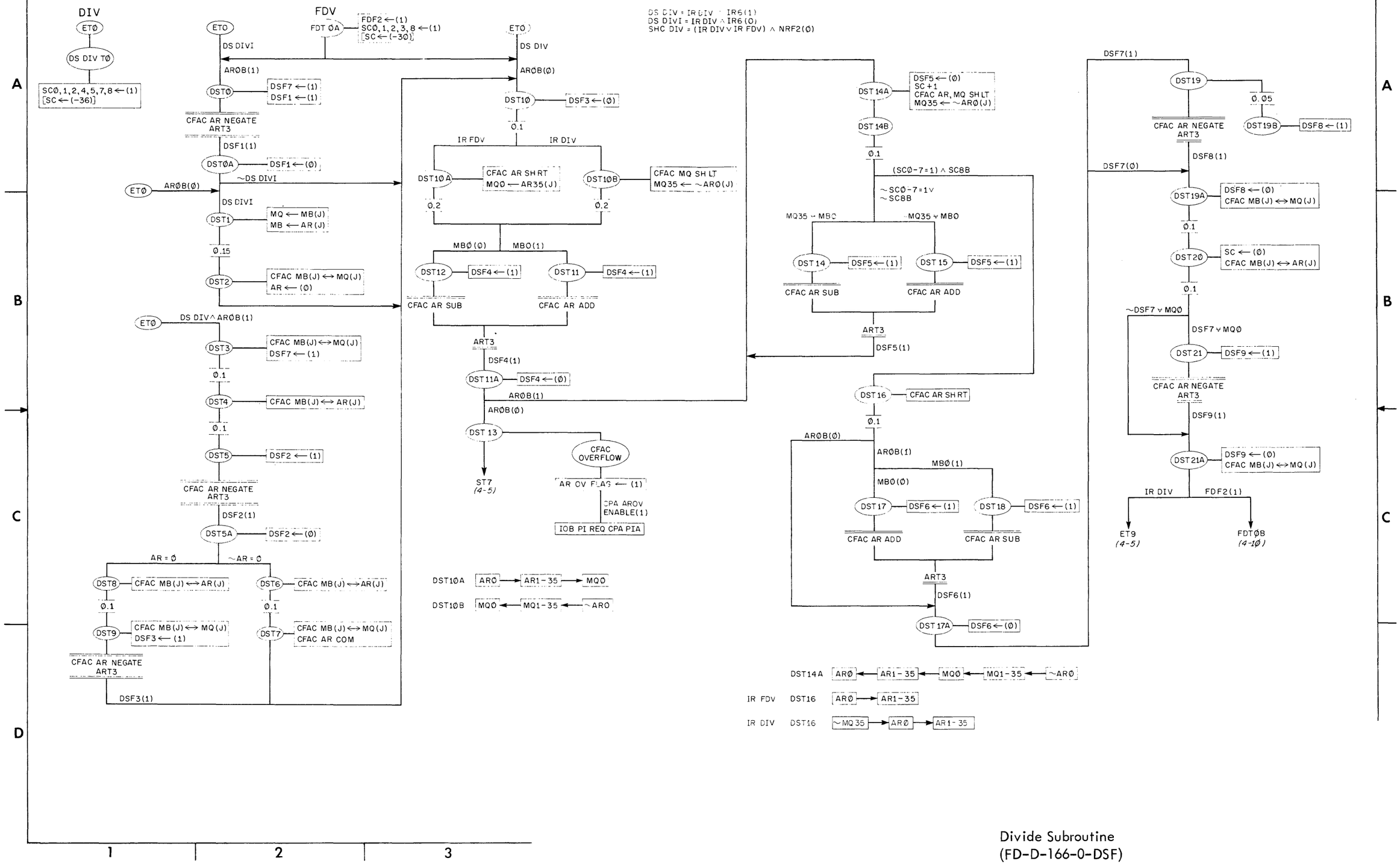
NORMALIZE RETURN



Normalize Shifts

Exponent Calculate, Multiply and Normalize Return Subroutines
(FD-D-166-0-EMNF)

DIVIDE



Divide Subroutine
(FD-D-166-0-DSF)

IN-OUT TRANSFER

IR = 111 XXX XXX XYY Y

IR IOT A = IRØ-2 (111)

IR IOT = IR IOT A ∧ ~EX IR UUO

YYY *specifies instruction*

IR IOT ∧ IR1Ø-12:
- ØØØ = IOT BLKI
- ØØ1 = IOT DATAI
- Ø1Ø = IOT BLKO
- Ø11 = IOT DATAO
- 1ØØ = IOT CONO
- 1Ø1 = IOT CONI
- 11Ø = IOT CONSZ
- 111 = IOT CONSO

XXX XXX X *specifies device.* IR3-9 *outputs thru bus drivers to* IOS3-9 *lines on* IO *bus. Each* IO *control unit contains a decoding net that allows proper code to select device by gating in command signals from bus*

Instruction Action

BLKI	C(E) + 1ØØØØØ1 → E, DEVICE BUFFER → C(C(ERT))
DATAI	DEVICE BUFFER → E
BLKO	C(E) + 1ØØØØØ1 → E, C(C(ERT)) → DEVICE BUFFER
DATAO	C(E) → DEVICE BUFFER
CONO	E → DEVICE CONTROL
CONI	STATUS → E
CONSZ	(STATUS ∧ E) = Ø: PC + 1
CONSO	(STATUS ∧ E) ≠ Ø: PC + 1

IOT BLK = IOT BLKI ∨ IOT BLKO
IOT OUTGOING = IOT DATAO ∨ IOT CONO
IOT STATUS = IOT CONI ∨ IOT CONSZ ∨ IOT CONSO
IOT DATAI ∨ O = IOT DATAI ∨ IOT DATAO
PI BLK RST = IOT DATAI ∨ O ∧ PI OV(Ø)
PI BLK RST ∧ PI CYC B(1): PI HOLD, PI RST
IOT CONSZ ∨ IOT CONSO: E LONG

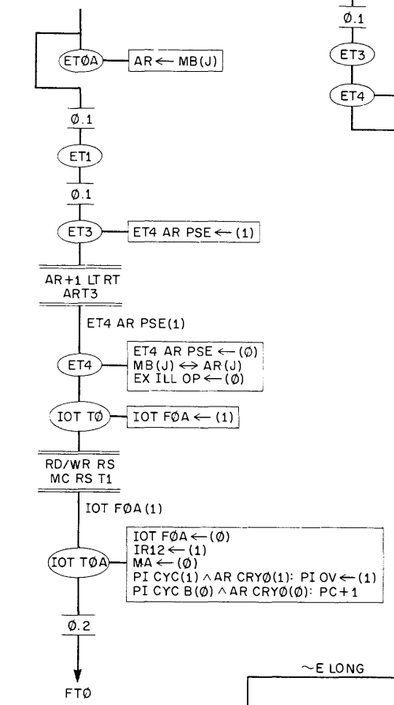
BLK

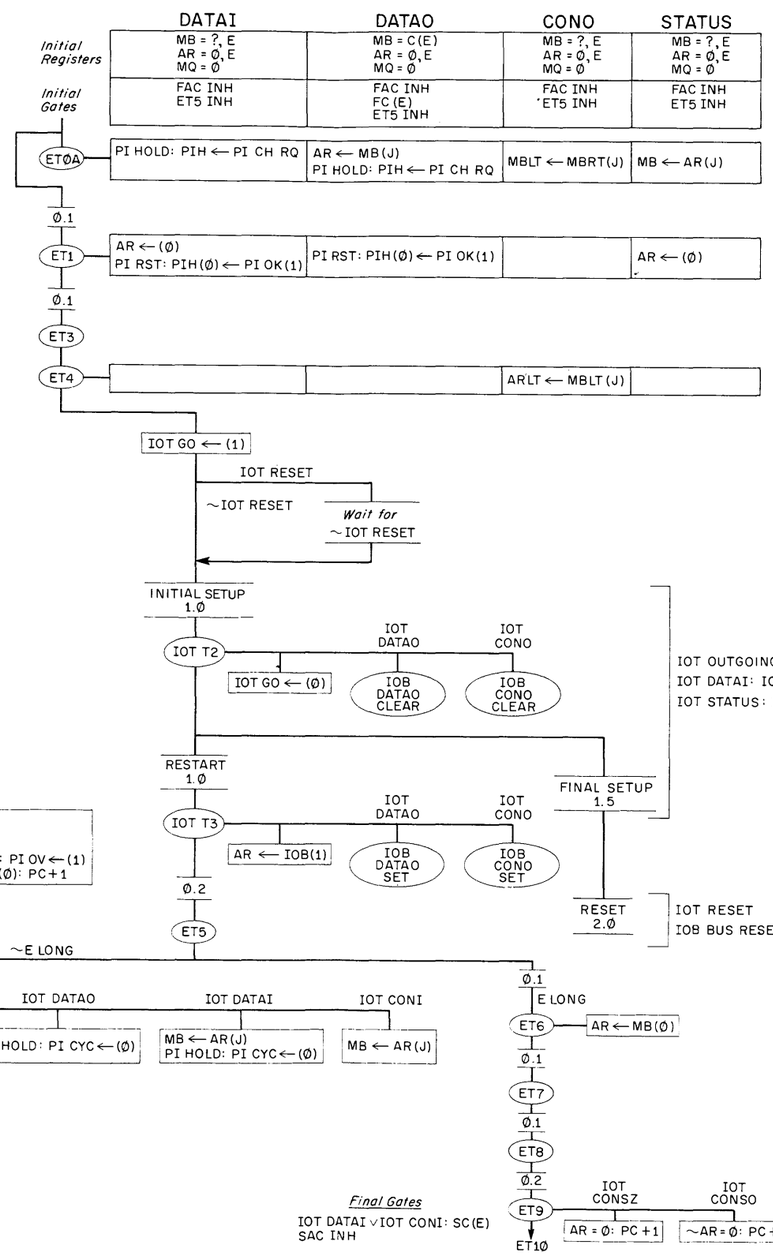
Initial Registers

MB = C(E)
AR = Ø, E
MQ = Ø

Initial Gates

FAC INH
FC(E) PSE
PC + 1 INH
ET4 INH
AR SBR
ET5 INH

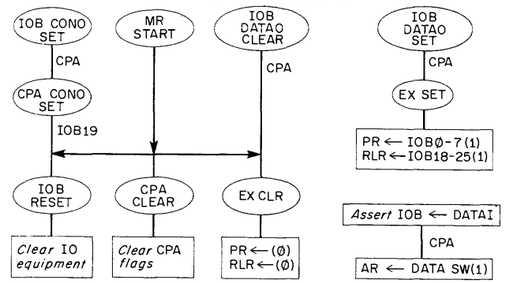




CPA

ØØØ

CPA = IOS3-9 (ØØØ ØØØ Ø)



IOB	CPA CONO SET	CPA STATUS
18	CPA PDL OV ← (Ø)	
19	IOB RESET	CPA PDL OV(1)
2Ø	CPA IOT USER ← (1)	CPA IOT USER(1)
21	CPA IOT USER ← (Ø)	EX USER B(1)
22	CPA ILLEG OP ← (Ø)	CPA ILLEG OP(1)
23	CPA NON EXIST MEM ← (Ø)	CPA NON EXIST MEM(1)
24	CPA CLOCK ENABLE ← (Ø)	
25	CPA CLOCK ENABLE ← (1)	CPA CLOCK ENABLE(1)
26	CPA CLOCK FLAG ← (Ø)	CPA CLOCK FLAG(1)
27	CPA PC CHG ENABLE ← (Ø)	
28	CPA PC CHG ENABLE ← (1)	CPA PC CHG ENABLE(1)
29	AR PC CHG FLAG ← (Ø)	AR PC CHG FLAG B(1)
3Ø	CPA AROV ENABLE ← (Ø)	
31	CPA AROV ENABLE ← (1)	CPA AROV ENABLE(1)
32	AR OV FLAG ← (Ø)	AR OV FLAG B(1)
33-35	CPA PIA ← IOB33-35(1)	CPA PIA B(1)

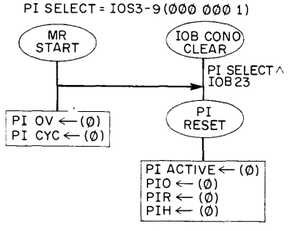
6Ø times per second from power control: CPA CLOCK FLAG ← (1)
MC ILLEG ADDRESS: CPA ILLEG OP ← (1)
MC NON EXIST MEM: CPA NON EXIST MEM ← (1)
ET1Ø ∧ [MB(J) ↔ AR(J)(ET1Ø)] ∧ AR CRYØ(1): CPA PDL OV ← (1)

IOB PI REQ CPA PIA
- CPA ILLEG OP(1)
- CPA NON EXIST MEM(1)
- CPA PDL OV(1)
- CPA CLOCK ENABLE(1) ∧ CPA CLOCK FLAG(1)
- CPA PC CHG ENABLE(1) ∧ AR PC CHG FLAG B(1)
- CPA AROV ENABLE(1) ∧ AR OV FLAG(1)

PI

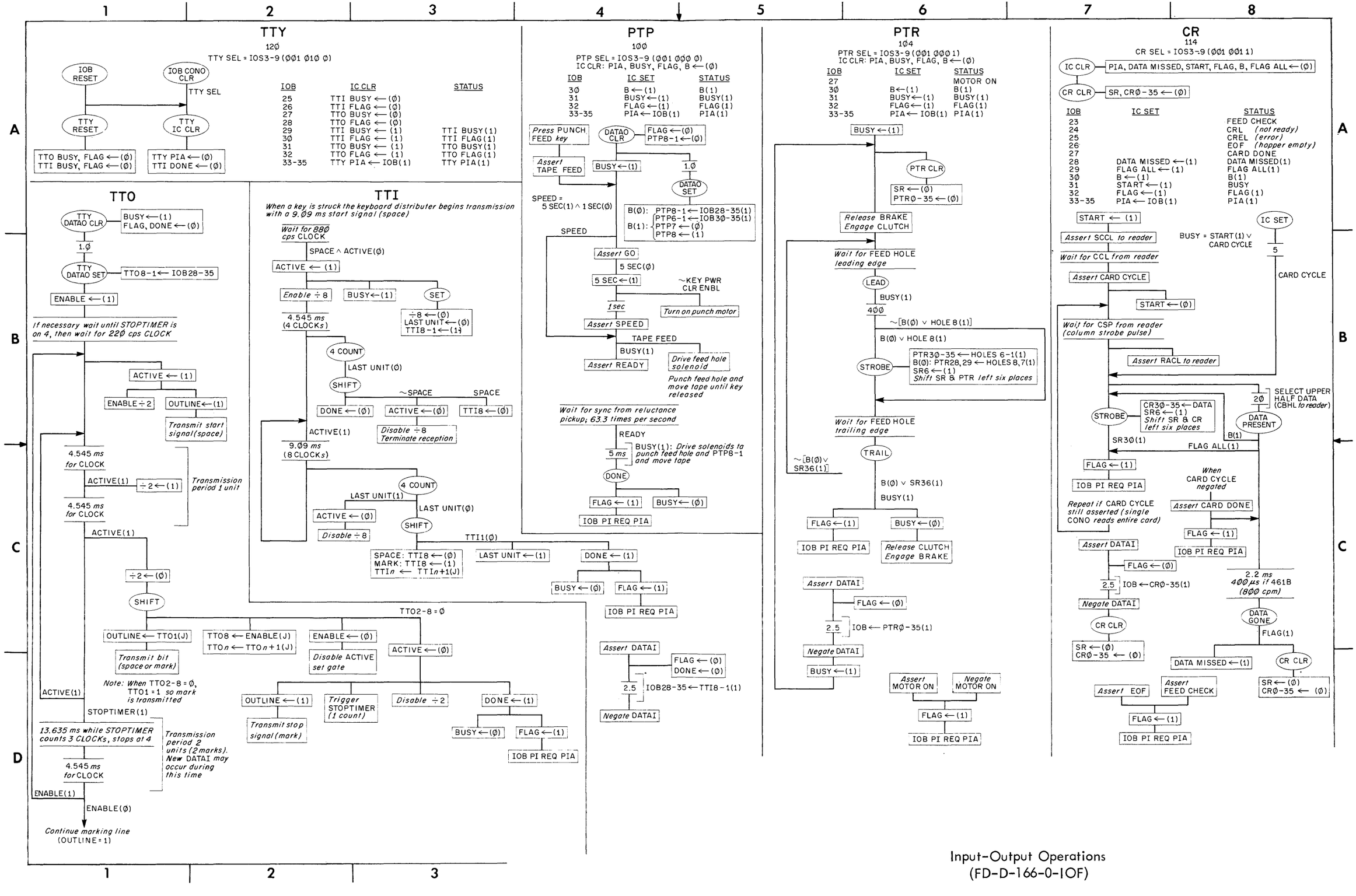
ØØ4

PI SELECT = IOS3-9 (ØØØ ØØØ 1)



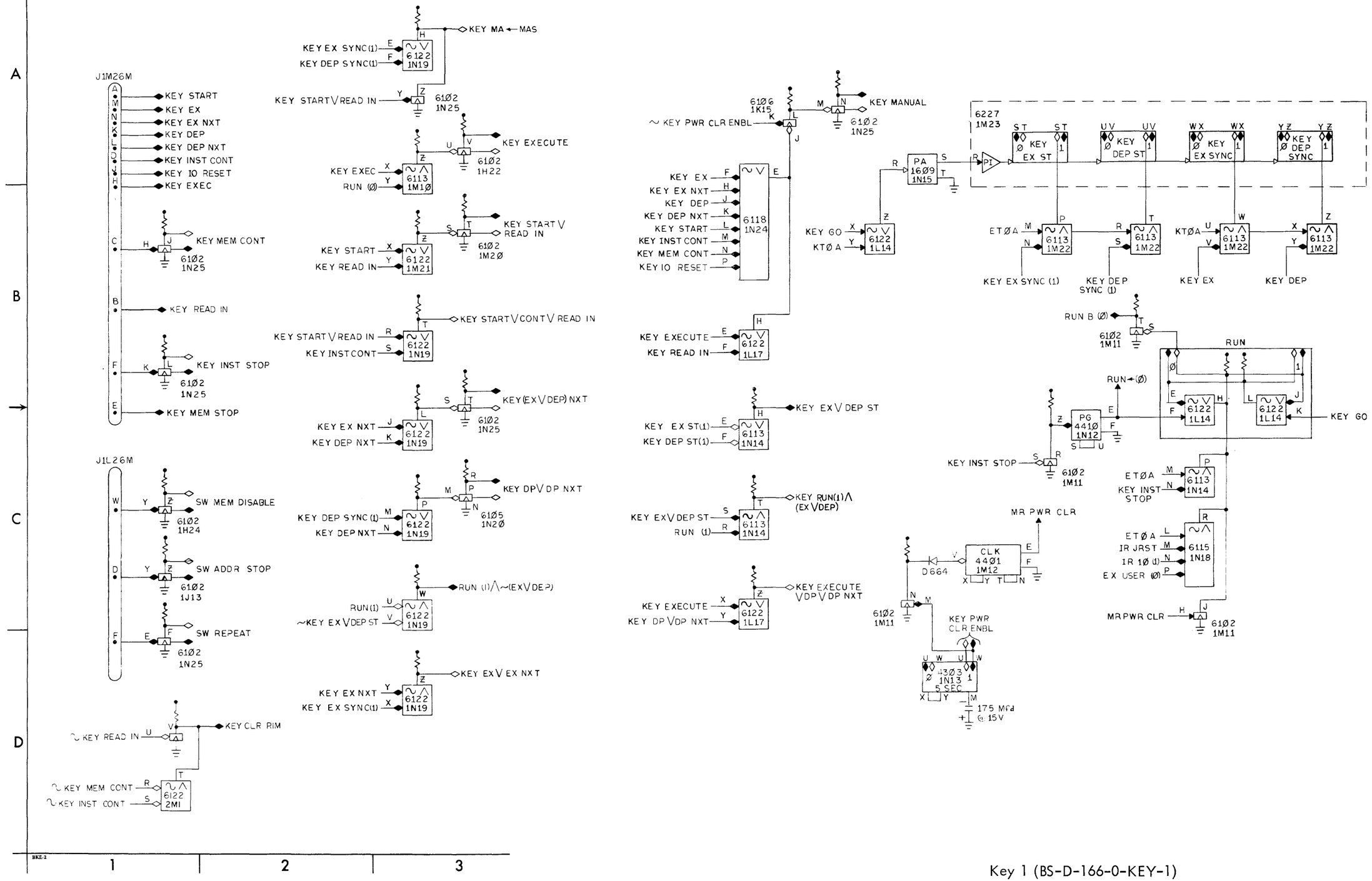
IOB	PI CONO SET	PI STATUS
24	PIR ← IOB 29-35(1)	
25	PIO ← IOB 29-35(1)	
26	PIO(Ø) ← IOB29-35(1)	
27	PI ACTIVE ← (Ø)	
28	PI ACTIVE ← (1)	PI ACTIVE(1)
29-35		PIO1-7(1)

IOT Instructions
(FD-D-166-0-IOTF)



Input-Output Operations
(FD-D-166-0-1OF)

Key 1 (BS-D-166-0-KEY-1)



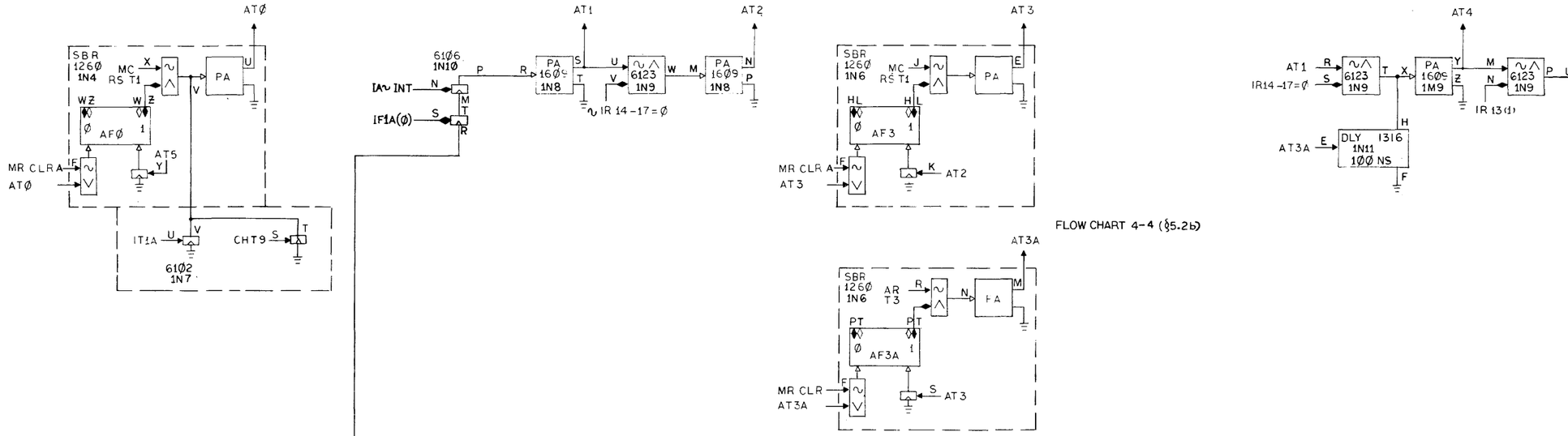
Key 1 (BS-D-166-0-KEY-1)

Key 2 (BS-D-166-0-KEY-2)

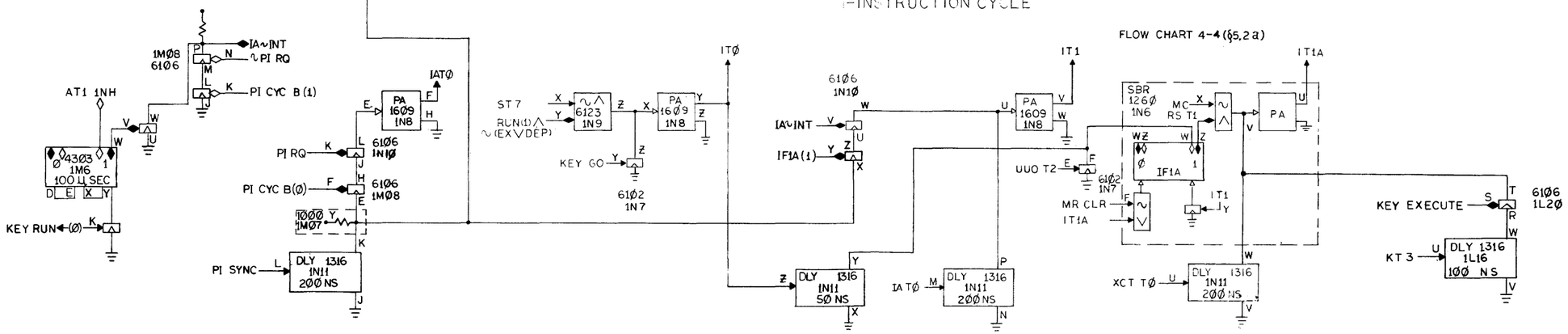
Key 2 (BS-D-166-0-KEY-2)

I, A - Instruction and Address Cycles
(BS-D-166-0-1A)

A-ADDRESS CYCLE



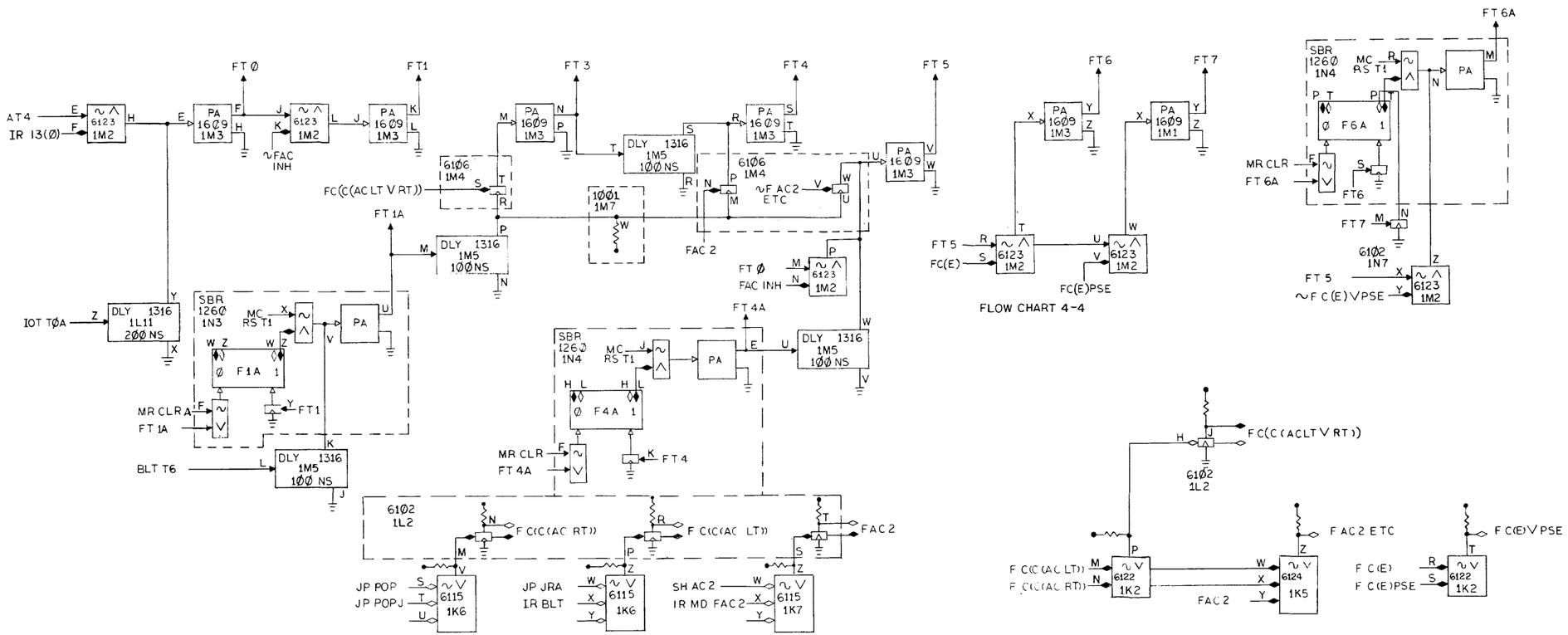
I-INSTRUCTION CYCLE



FLOW CHART 4-4 (§5.2a)

I, A - Instruction and Address Cycles
(BS-D-166-0-1A)

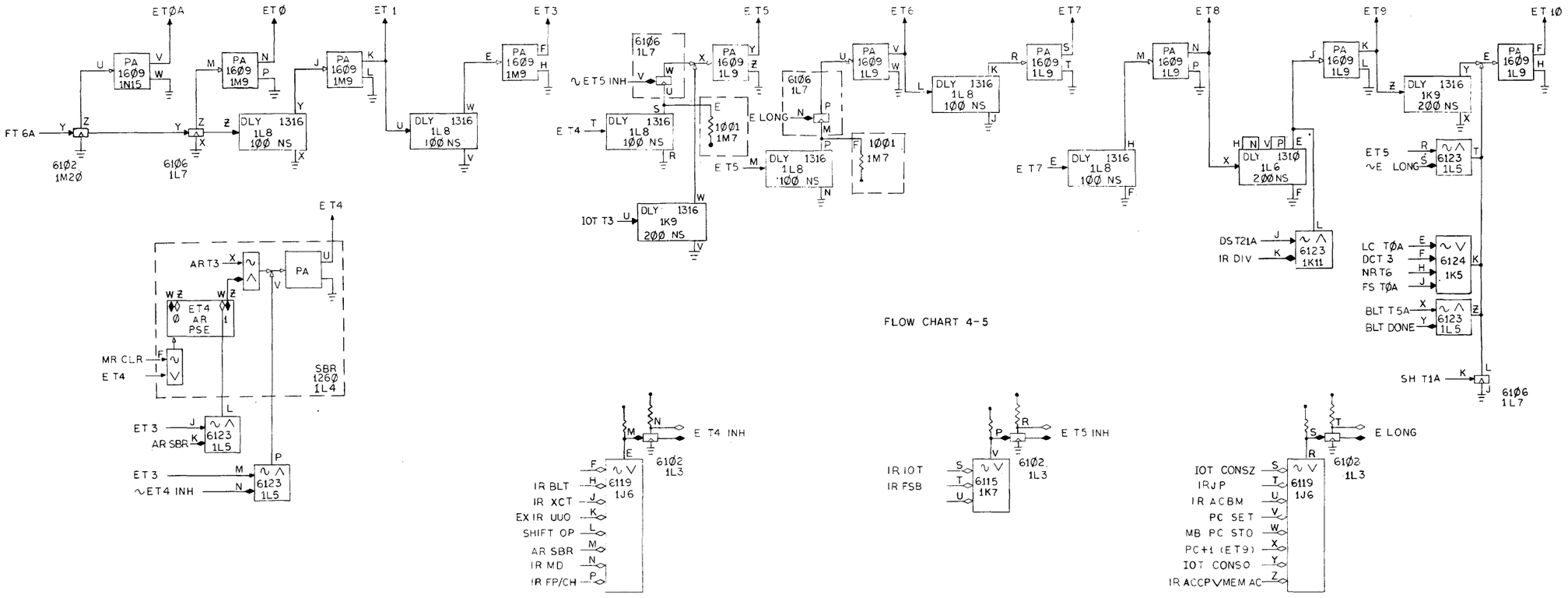
F - Fetch Cycle (BS-D-166-0-F)



BKE-2

F - Fetch Cycle (BS-D-166-0-F)

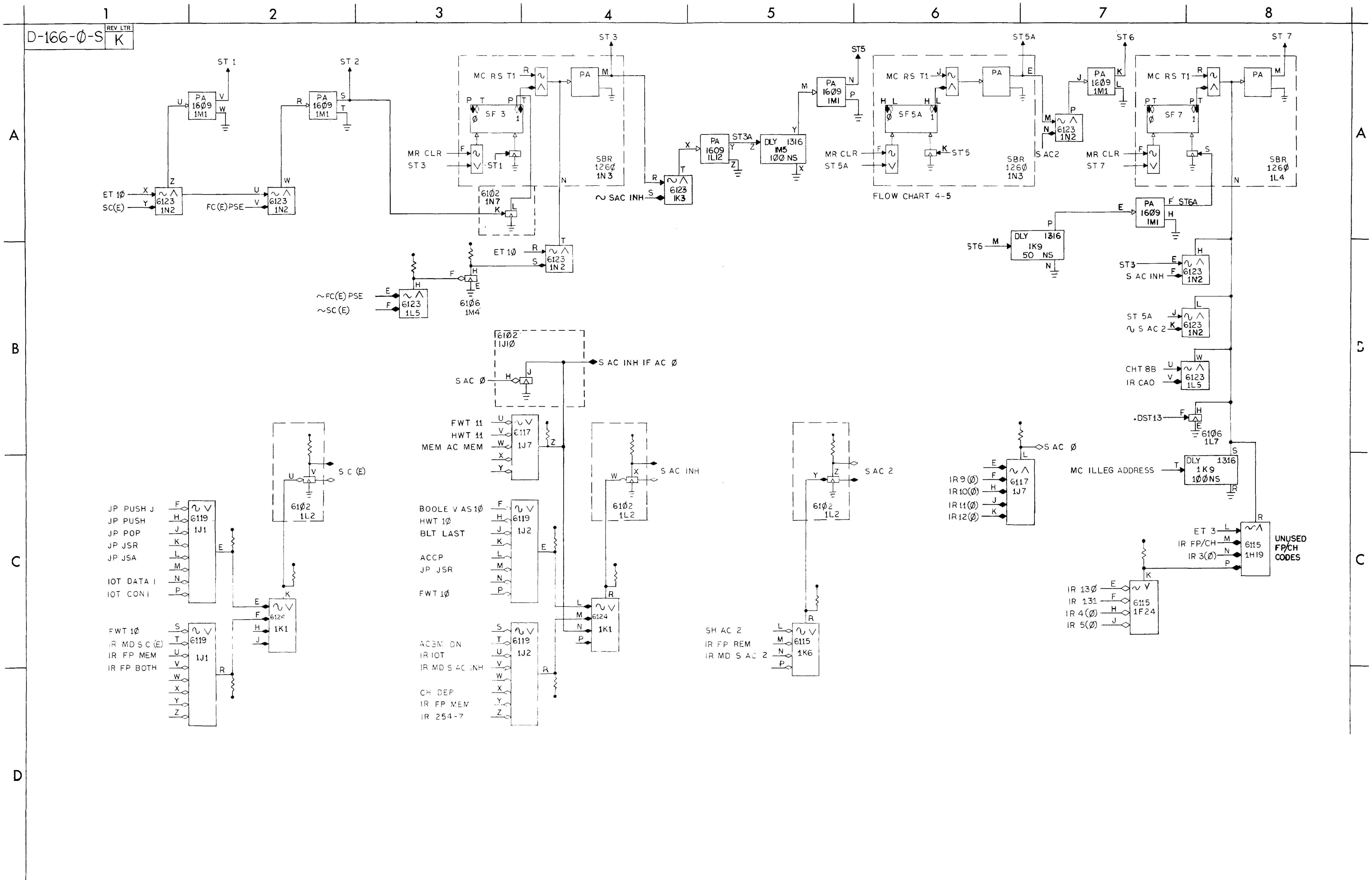
E - Execute Cycle (BS-D-166-0-E)



FLOW CHART 4-5

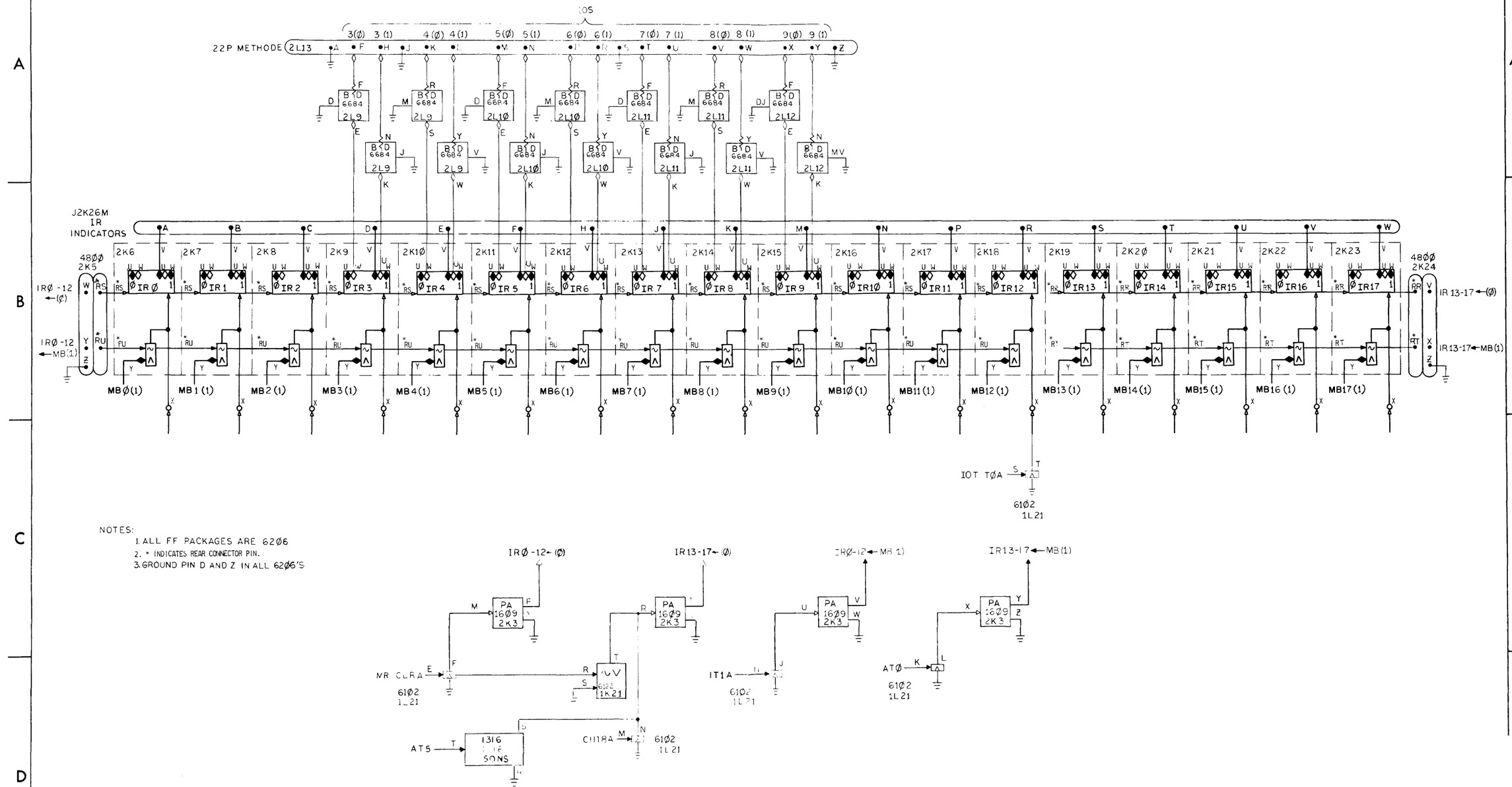
E - Execute Cycle (BS-D-166-0-E)

S - Store Cycle (BS-D-166-0-S)



S - Store Cycle (BS-D-166-0-S)

IR - Instruction Register
(BS-D-166-0-IR-1)

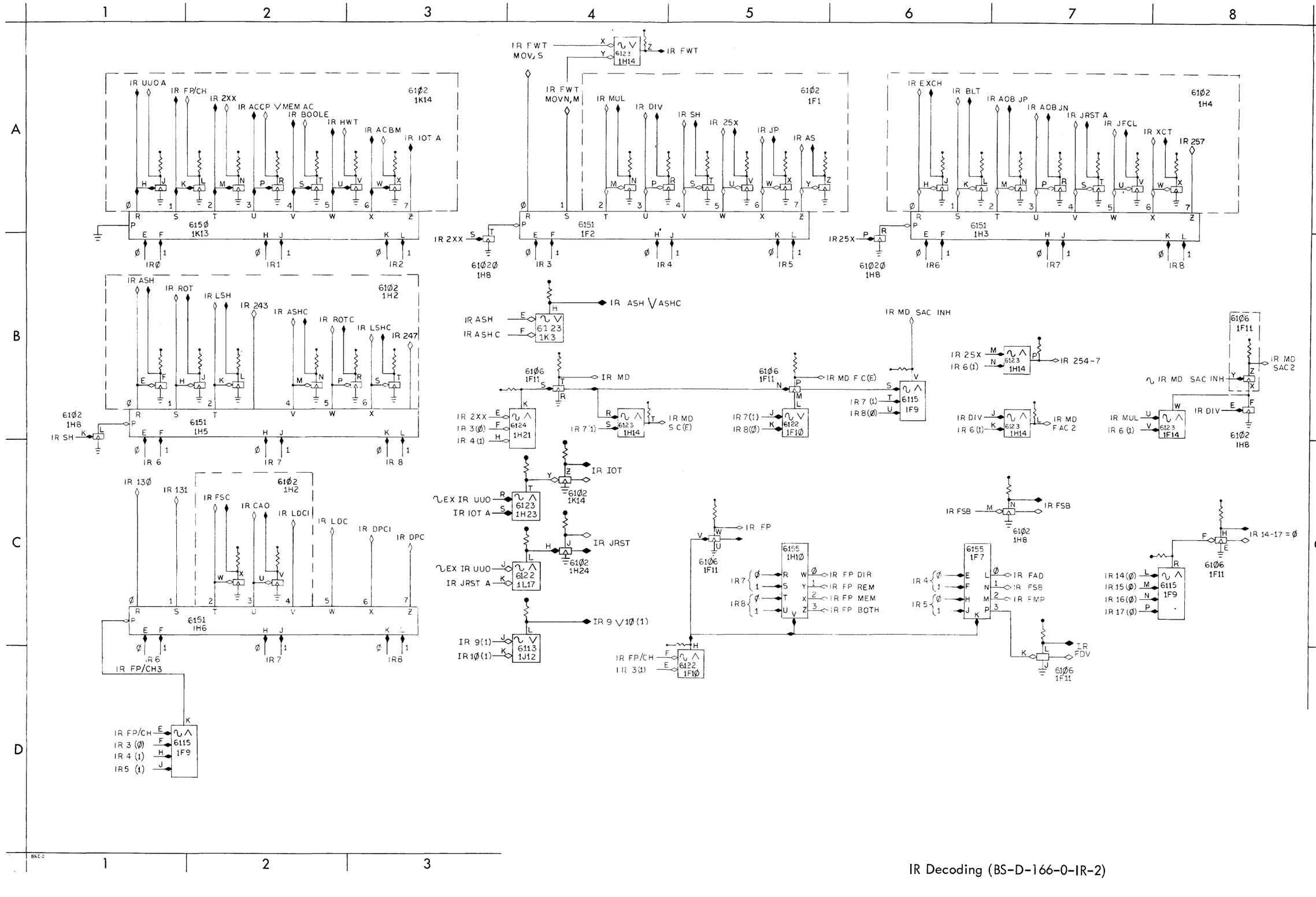


NOTES:

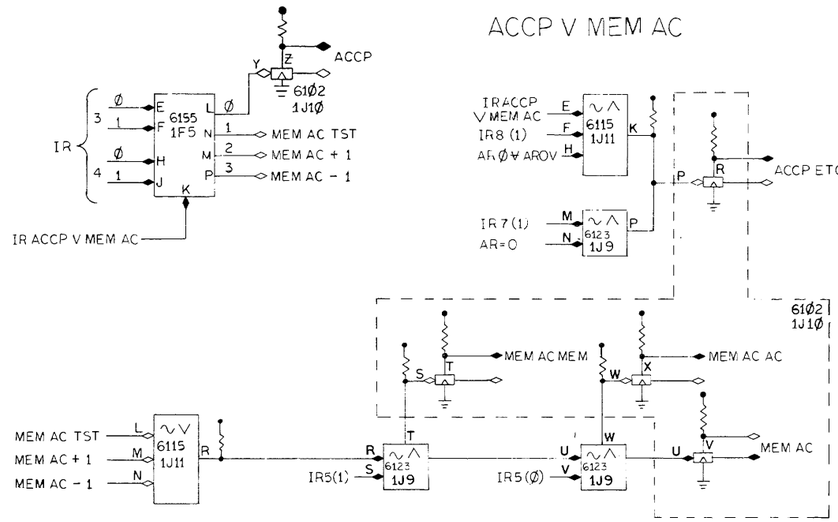
1. ALL FF PACKAGES ARE 62Ø6
2. * INDICATES REAR CONNECTOR PIN.
3. GROUND PIN D AND Z IN ALL 62Ø6'S

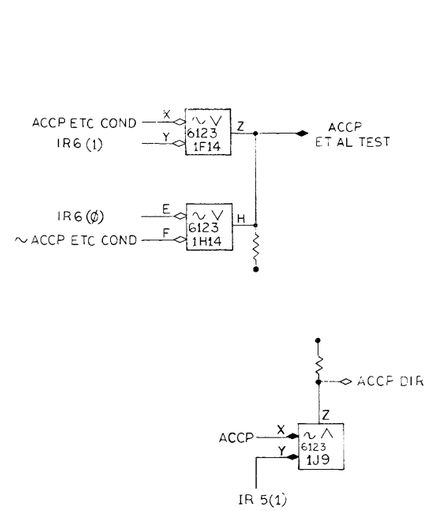
IR - Instruction Register
(BS-D-166-0-IR-1)

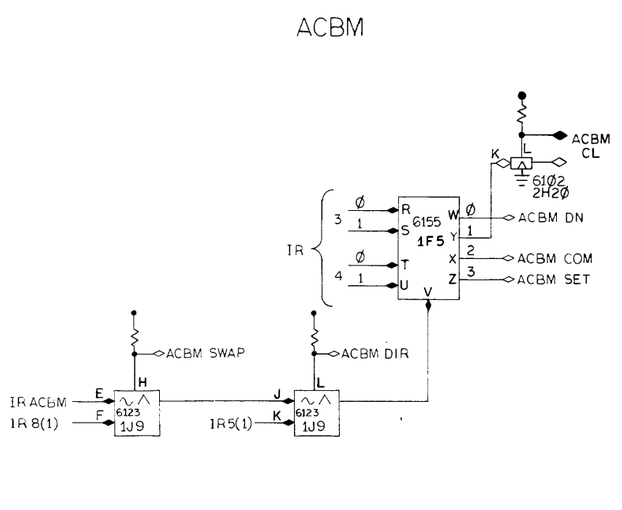
IR Decoding (BS-D-166-0-IR-2)



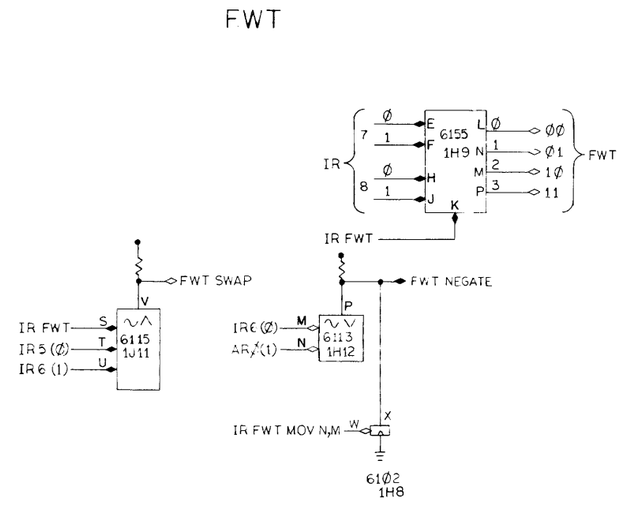
IR Decoding (BS-D-166-0-IR-2)

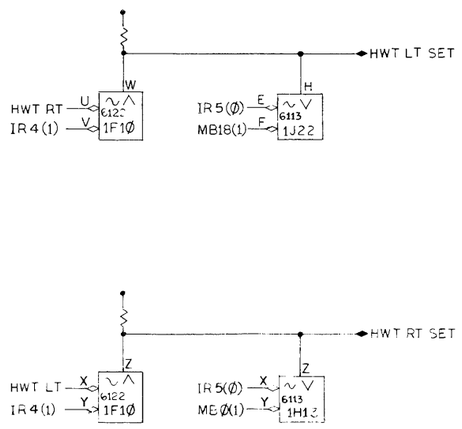


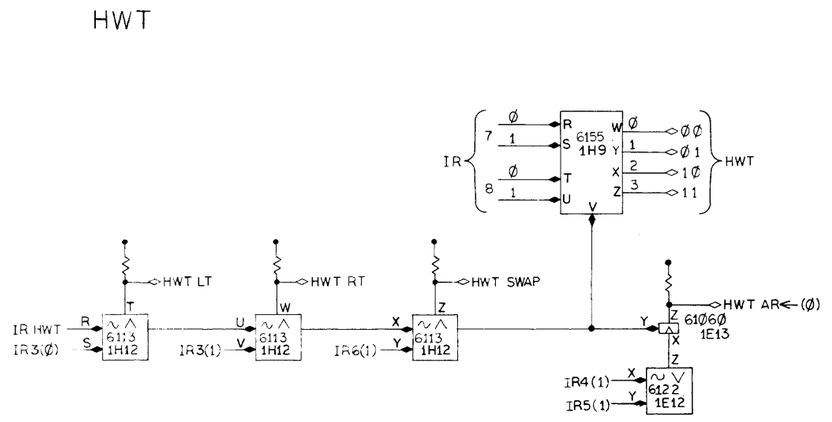




FLOW CHART 4-6



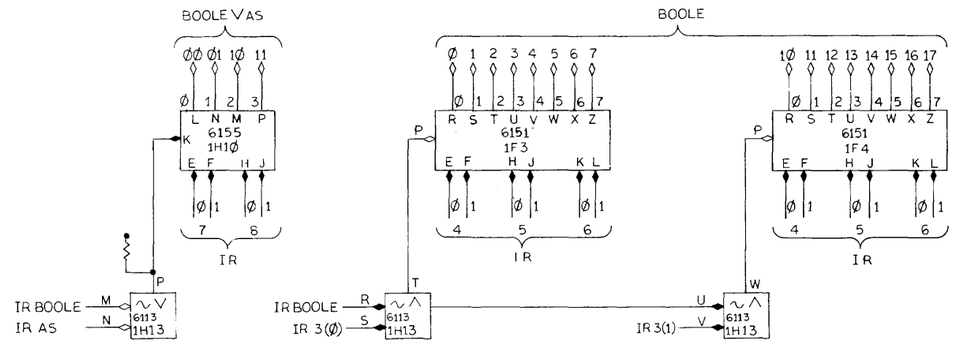




ACCP, MEMAC, ACBM, FWT, HWT
(BS-D-166-0-INS-1)

BOOLE

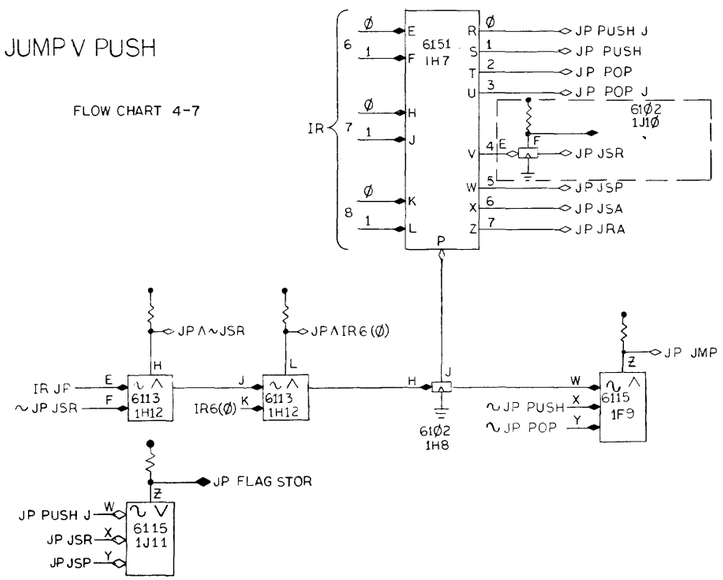
			RESULT BIT FOR OPERAND BIT CONFIGURATION				ETØ T1		ET1 T2			ET4 T3
			MEM AC ØØ	MEM AC Ø1	MEM AC 1Ø	MEM AC 11	AR CLR	AR COM	AR←MB(Ø)	AR←MB(1)	AR←MB ∨	AR COM
SETZ	Ø	ØØ	Ø	Ø	Ø	Ø	✓					
AND	MEM ∧ AC	Ø1	Ø	Ø	Ø	1		✓	✓			
ANDCA	MEM ∧ ∼ AC	Ø2	Ø	Ø	1	Ø		✓	✓			
SETM	MEM	Ø3	Ø	Ø	1	1	✓			✓		
ANDCM	∼ MEM ∧ AC	Ø4	Ø	1	Ø	Ø		✓		✓		✓
SETA	AC	Ø5	Ø	1	Ø	1					✓	
XOR	MEM ∨ AC	Ø6	Ø	1	1	Ø				✓		
IOR	MEM V AC	Ø7	Ø	1	1	1				✓		
ANDCB	∼ MEM ∧ ∼ AC	1Ø	1	Ø	Ø	Ø				✓		✓
EQV	MEM ≡ AC	11	1	Ø	Ø	1				✓	✓	✓
SETCA	∼ AC	12	1	Ø	1	Ø		✓				
ORCA	MEM V ∼ AC	13	1	Ø	1	1		✓		✓		
SETCM	∼ MEM	14	1	1	Ø	Ø	✓				✓	✓
ORCM	∼ MEM V AC	15	1	1	Ø	1		✓	✓			✓
ORCB	∼ MEM V ∼ AC	16	1	1	1	Ø			✓			✓
SETO	1	17	1	1	1	1	✓					✓



FLOW CHART 4-8

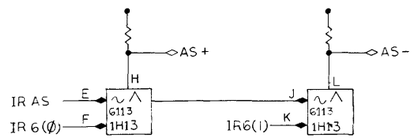
JUMP V PUSH

FLOW CHART 4-7



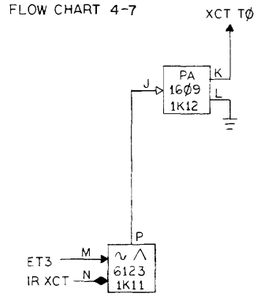
AS

FLOW CHART 4-8

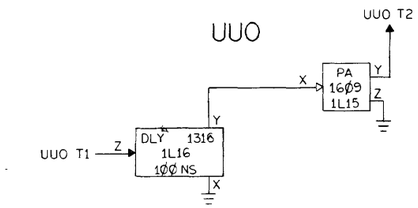


XCT

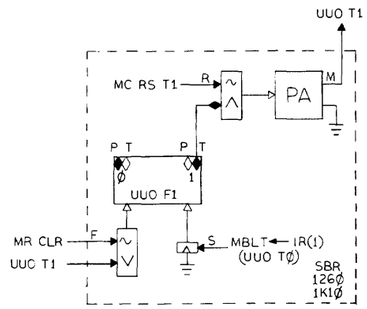
FLOW CHART 4-7



UUO

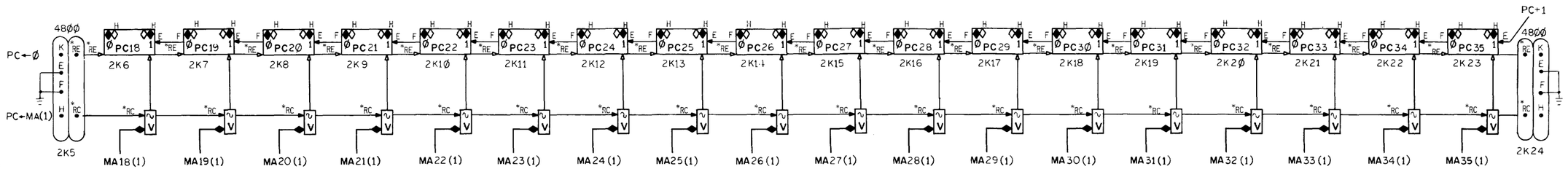


FLOW CHART 4-7



BOOLE, JP, AS, XCT, UUO
(BS-D-166-0-INS-2)

PC - Program Counter (BS-D-166-0-PC-2)

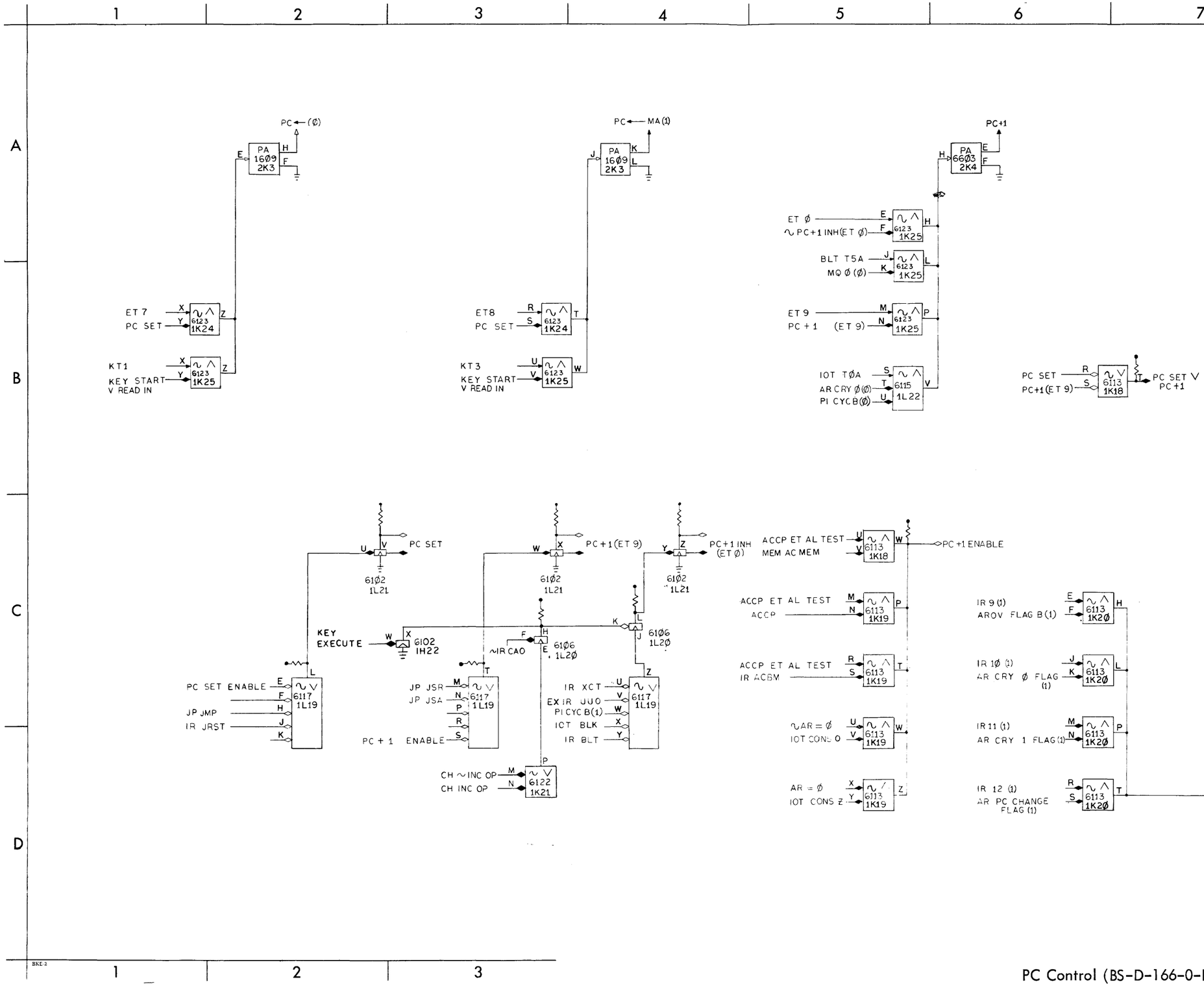


NOTES.

1. ALL FF PACKAGES ARE 62Ø6
2. * INDICATES REAR CONNECTOR PIN
3. GROUND PIN D AND Z IN ALL 62Ø6'S

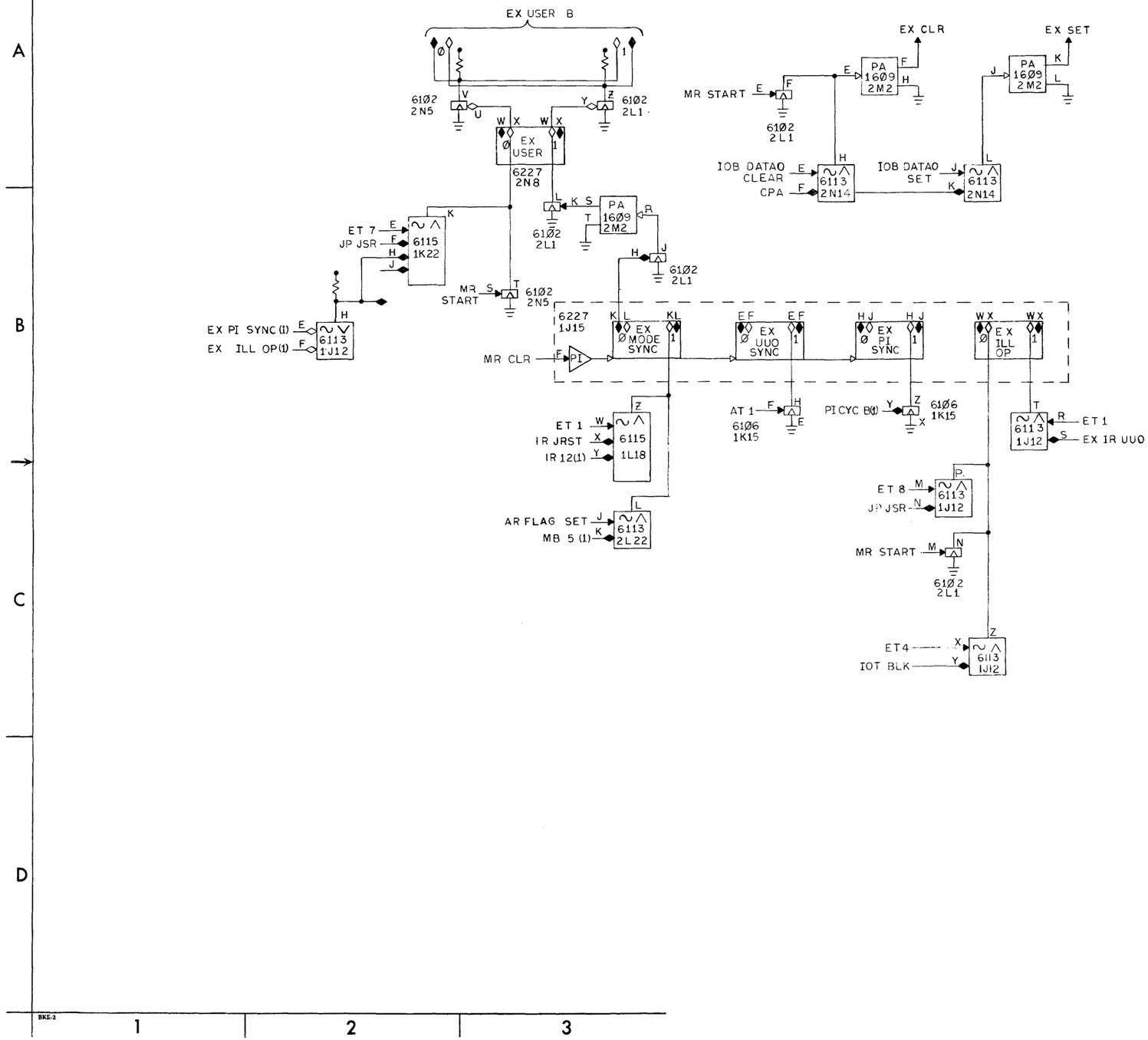
PC - Program Counter (BS-D-166-0-PC-2)

PC Control (BS-D-166-0-PC-1)



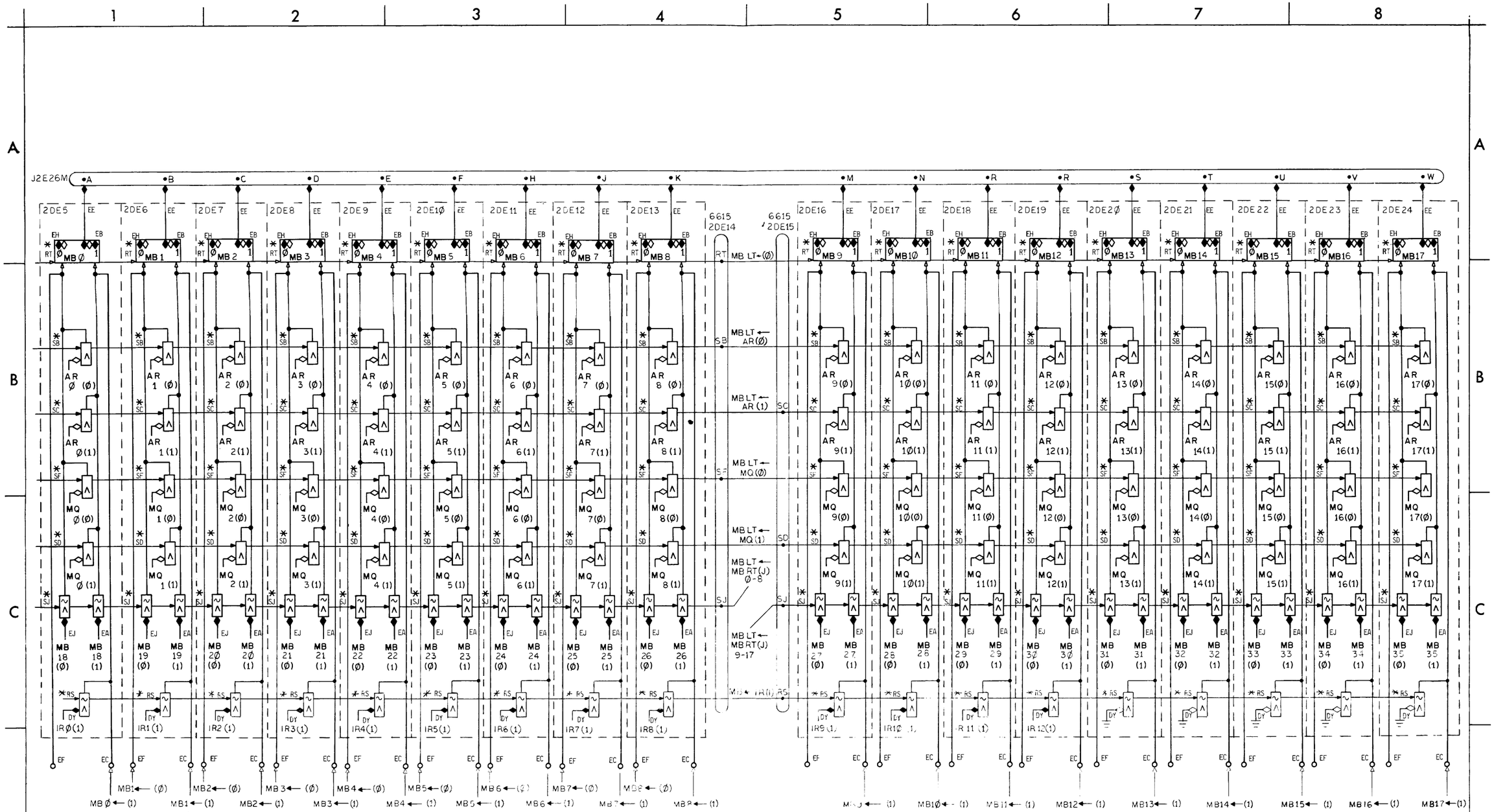
PC Control (BS-D-166-0-PC-1)

EX - Executive Mode (BS-D-166-0-EX)



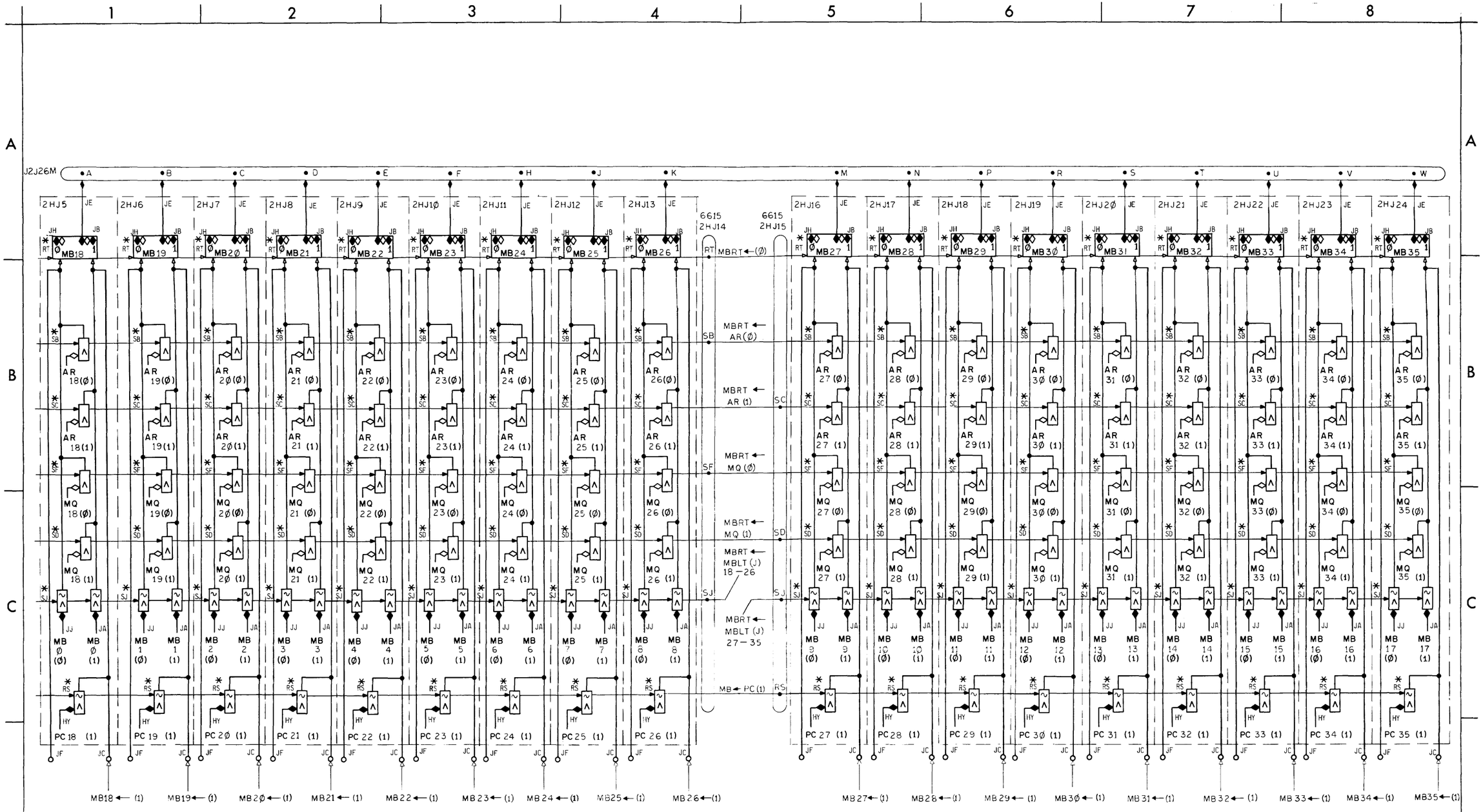
EX - Executive Mode (BS-D-166-0-EX)

MBLT Register 0-17 (BS-D-166-0-MB-2)



NOTES:
1. ALL FF PACKAGES ARE 6205
2. * INDICATES REAR CONNECTOR PIN
3. GROUND PIN D,P AND Z IN ALL 6205s AND 6615s.

MBLT Register 0-17 (BS-D-166-0-MB-2)



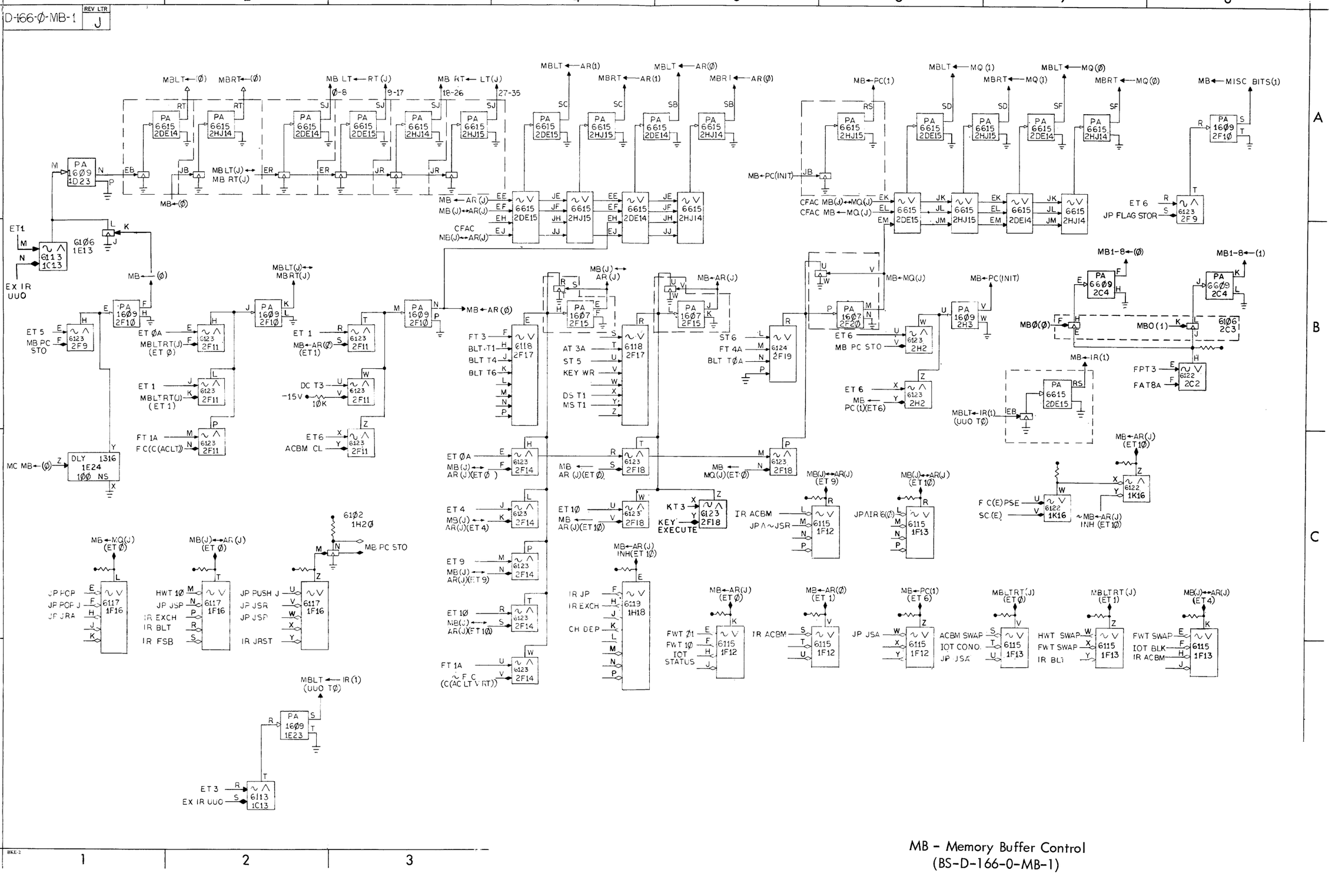
NOTES

1. ALL FF PACKAGES ARE 6205
2. * INDICATES REAR CONNECTOR PIN
3. GROUND PIN D,P AND Z IN ALL 6205's AND 6615's.

MBRT Register 18-35 (BS-D-166-0-MB-3)

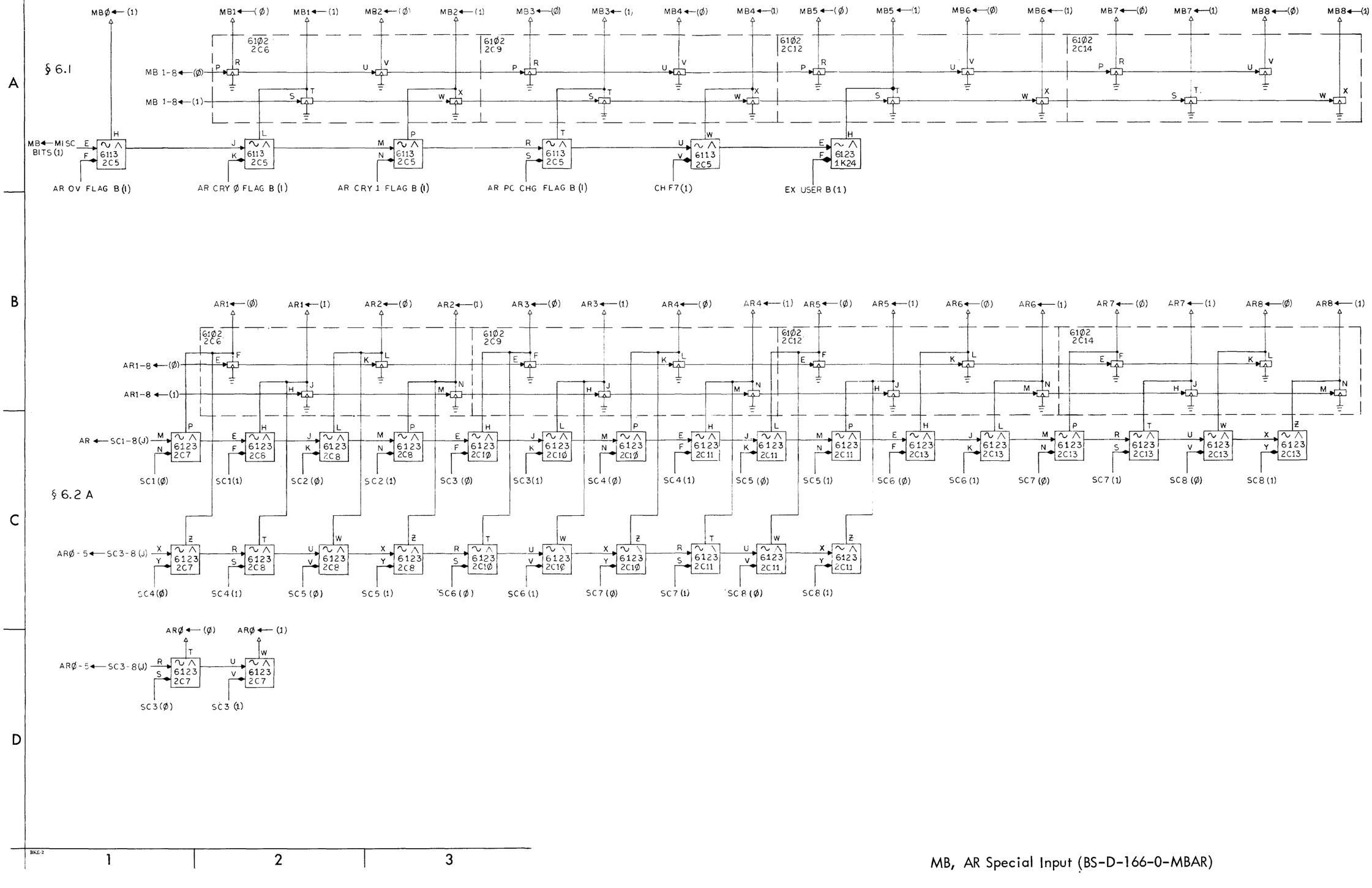
MB - Memory Buffer Control
(BS-D-166-0-MB-1)

D-166-Ø-MB-1 | REV LTR J



MB - Memory Buffer Control
(BS-D-166-0-MB-1)

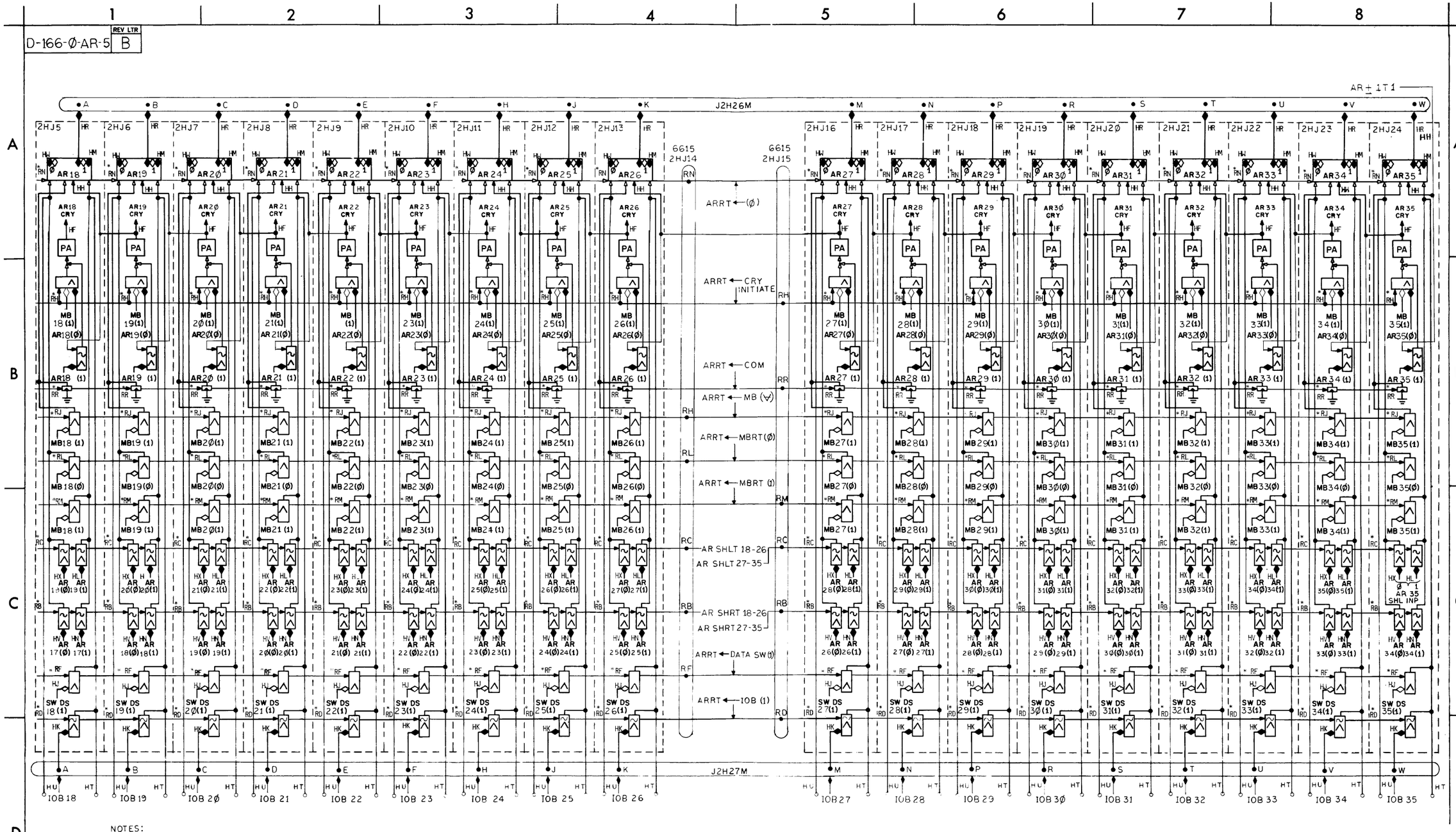
MB, AR Special Input (BS-D-166-0-MBAR)



MB, AR Special Input (BS-D-166-0-MBAR)

ARLT Register 0-17 (BS-D-166-0-AR-4)

D-166-Ø-AR-5 | REV LTR B



NOTES:

1. ALL FF PACKAGES ARE 62Ø5
2. * INDICATES REAR CONNECTOR PIN
3. GROUND PIN D AND Z IN ALL 62Ø5's AND 6615's.

ARRT Register 18-35 (BS-D-166-0-AR-5)

AR, MQ Shift Connections (BS-D-166-0-ARMQ)

D-166-Ø-ARMQ REV LTR D

LSH (SCT1): Ø → ARØ → AR1-35 → Ø

LSHC (SCT1): Ø → ARØ → AR1-35 → MQØ → MQ1-35 → Ø

ASH (SCT1): ARØ → AR1-35 → Ø

ASHC (SCT1) (NRTØ) (NRT2) (FDT1): ARØ → MQØ; ARØ → AR1-35 → MQ1-35 → Ø

ROT (SCT1): ARØ → AR1-35

ROTC (SCT1): ARØ → AR1-35 → MQØ → MQ1-35

MULT (MST2): ARØ → AR1-35 → MQØ → MQ1-35 → MQ 36

MUL LAST (MST5): ARØ → MQØ; AR1-35 → MQ1-35

DIV LOOP (DST14A): ARØ ← AR1-35 ← MQØ ← MQ1-35 ← ∼ARØ

FDV LAST (DST16): ARØ → AR1-35

DIV LAST (DST16): ∼MQ35 → ARØ → AR1-35

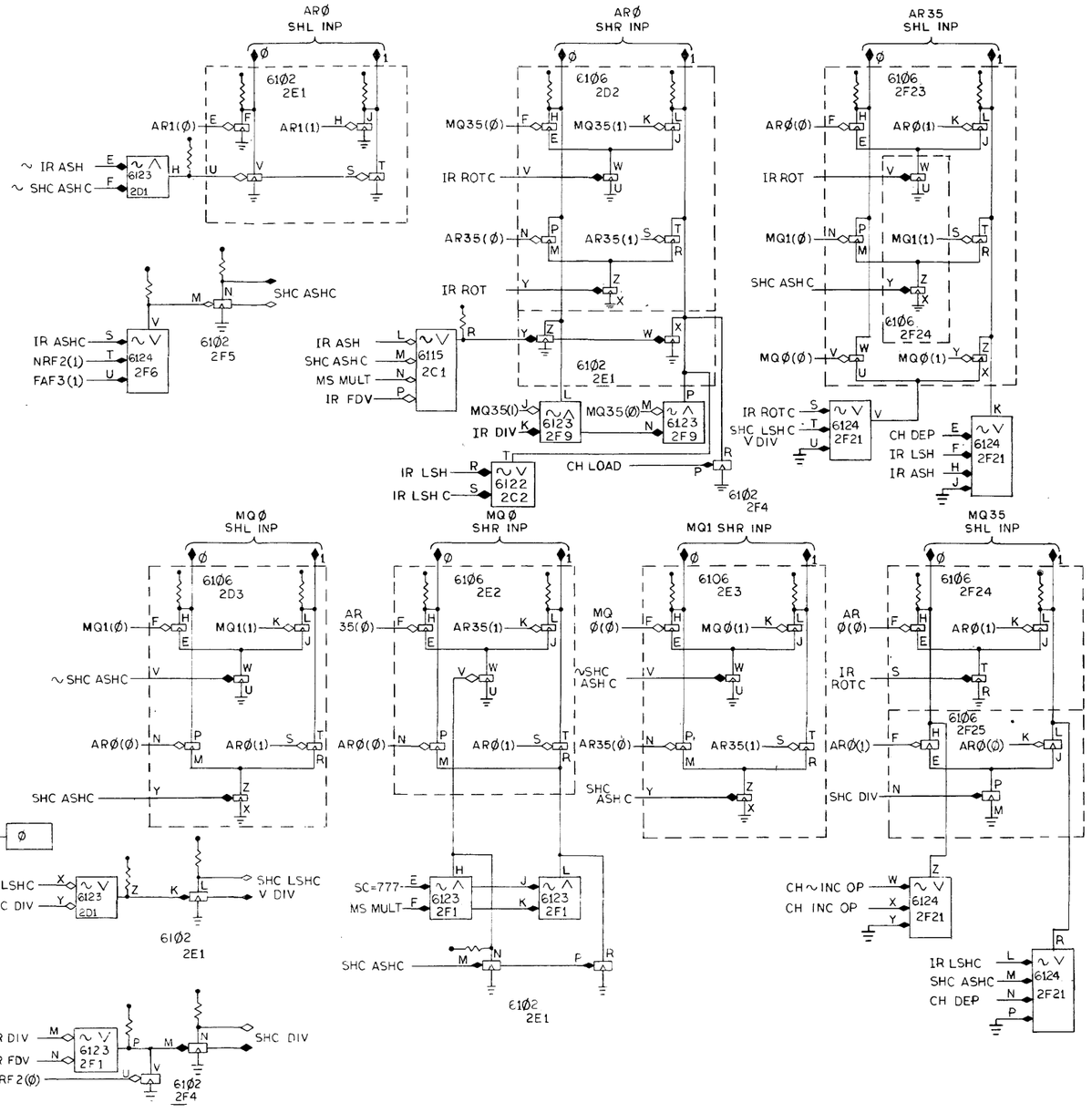
CH DEP (SCT1): ARØ ← AR1-35 ← Ø; MQØ ← MQ1-35 ← Ø

CH LD (SCT1): Ø → ARØ → AR1-35

CHAR FIRST (SCT1): MQØ ← MQ1-35 ← 1

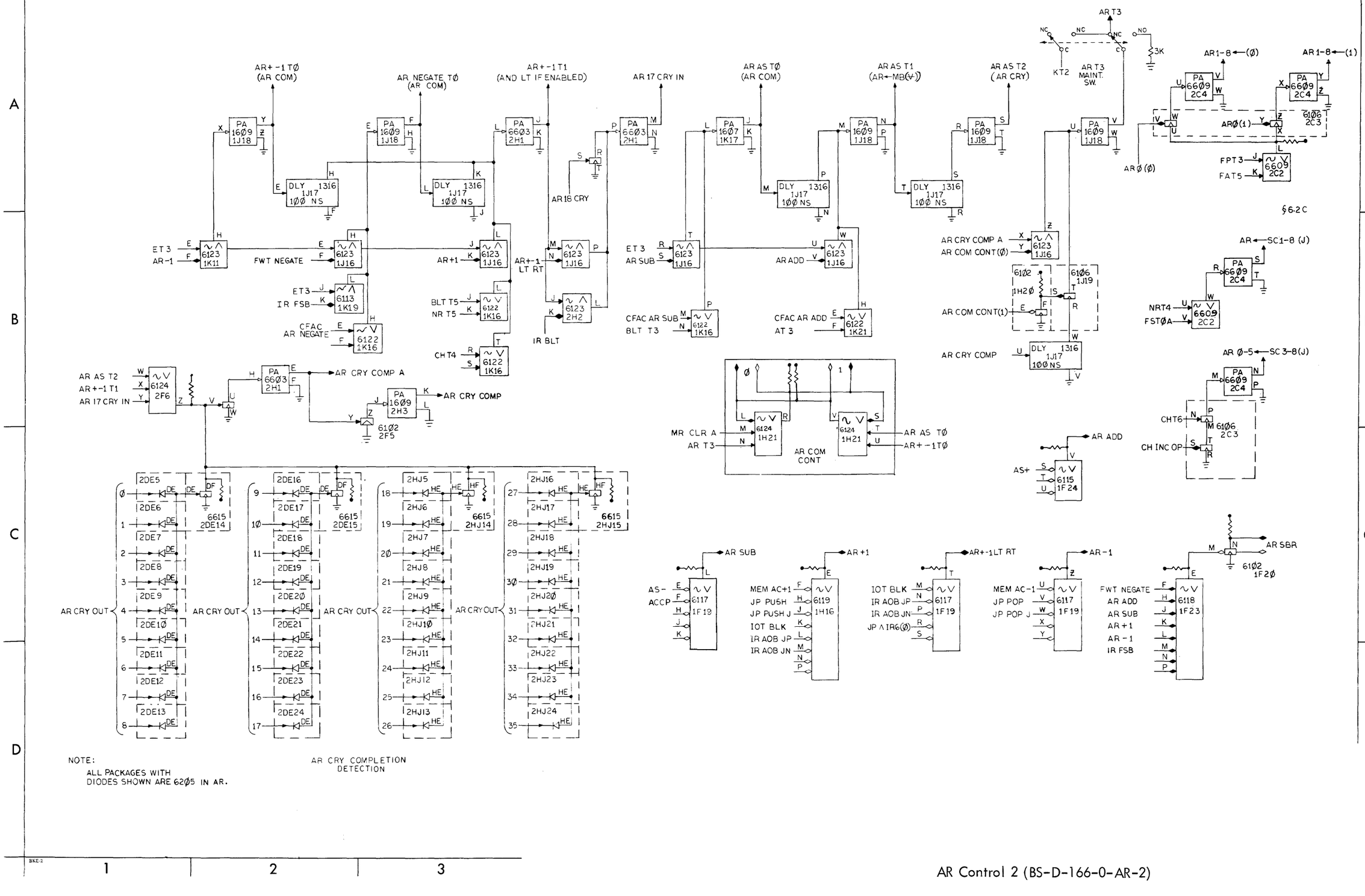
DIV AR FIRST (DST1ØA): ARØ → AR1-35 → MQØ

DIV MQ FIRST (DST1ØB): MQØ ← MQ1-35 ← ∼ARØ

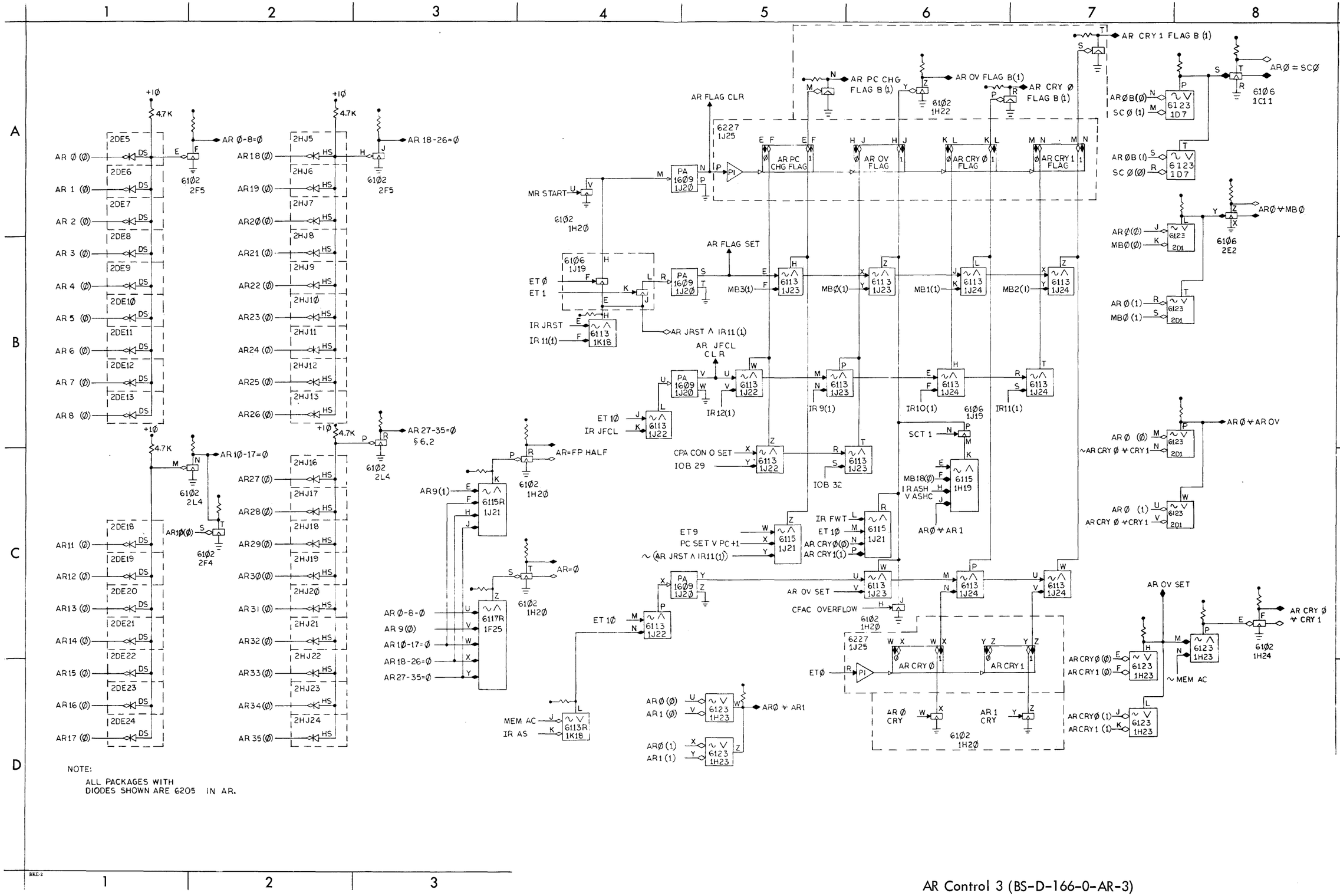


AR, MQ Shift Connections (BS-D-166-0-ARMQ)

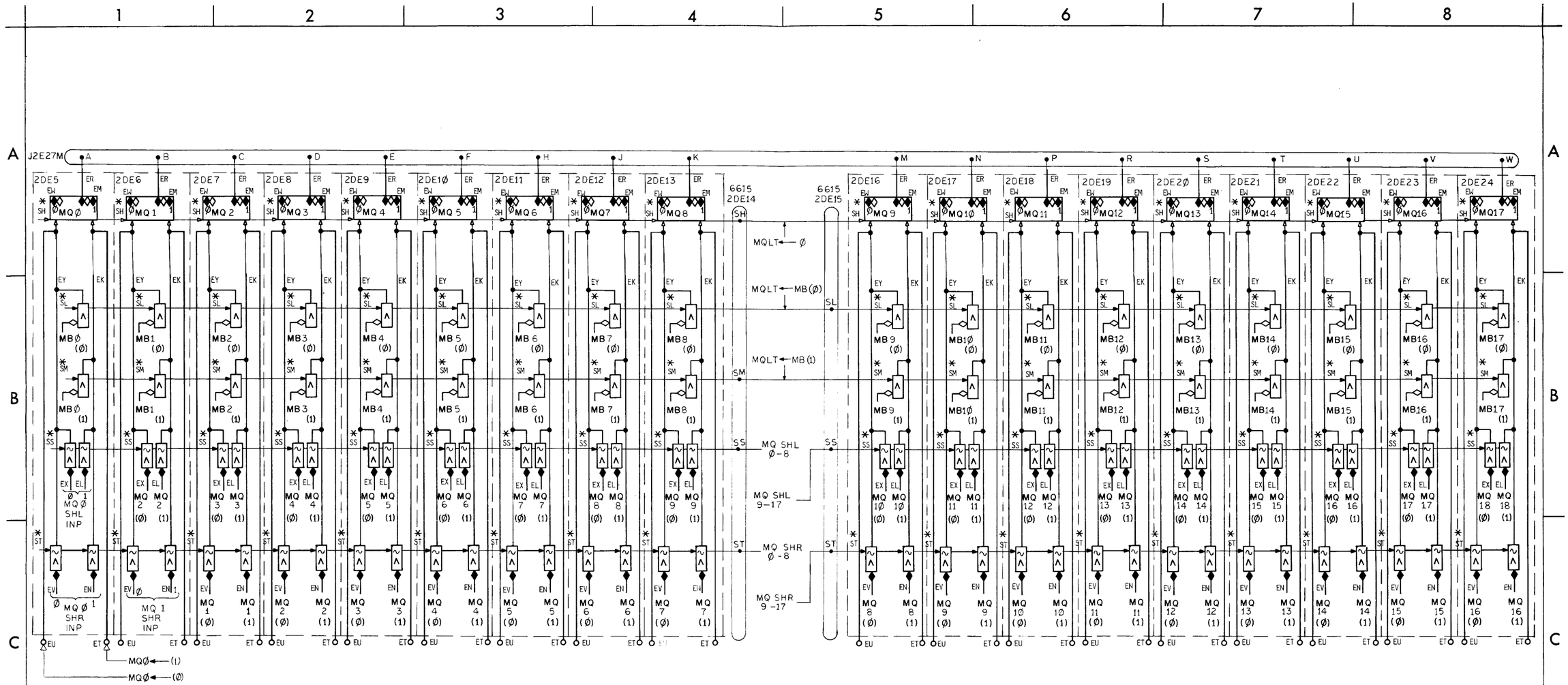
AR Control 1 (BS-D-166-0-AR-1)



AR Control 2 (BS-D-166-0-AR-2)



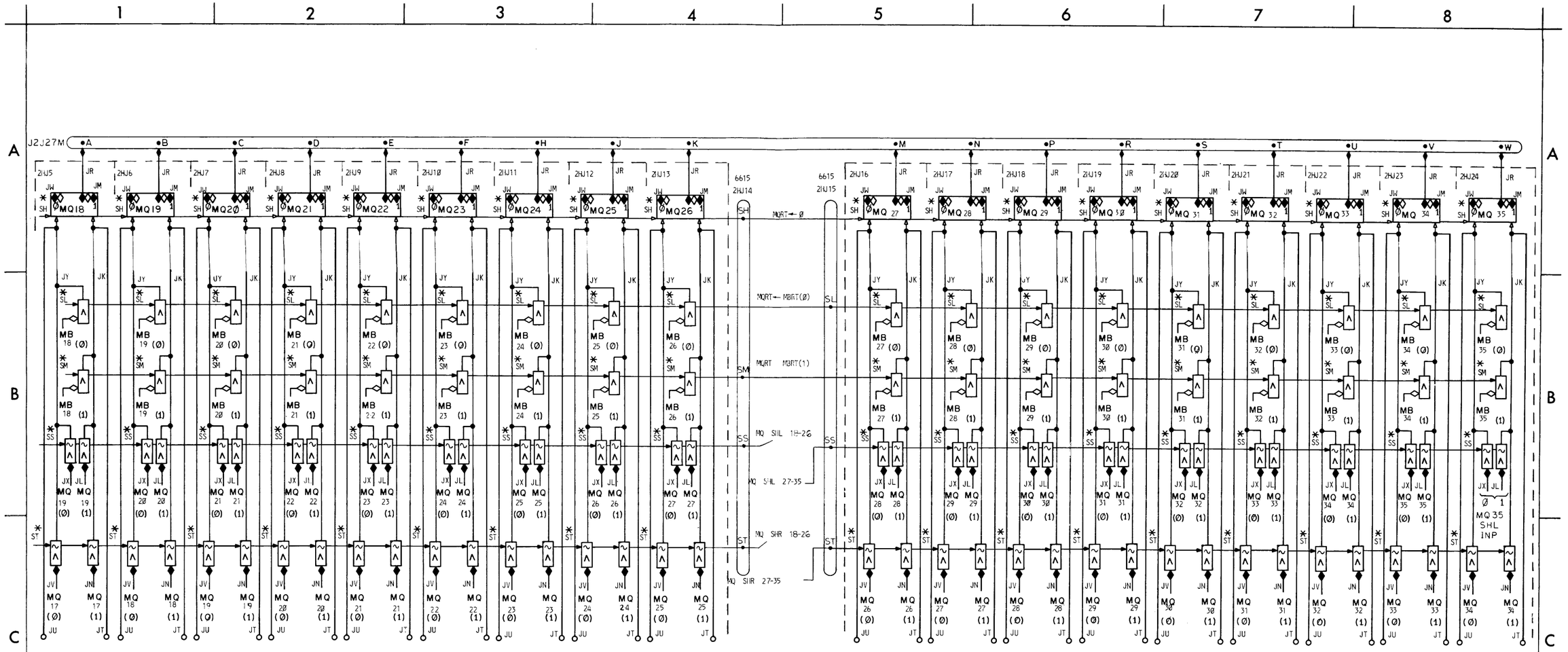
AR Control 3 (BS-D-166-0-AR-3)



NOTES:

1. ALL FF PACKAGES ARE 62Ø5
2. * INDICATES REAR CONNECTOR PIN
3. GROUND PIN D,P AND Z IN ALL 62Ø5'S AND 6615'S.

MQLT Register 0-17 (BS-D-166-0-MQ-2)

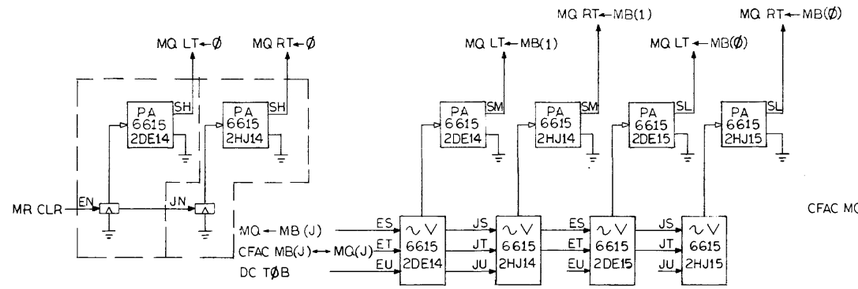


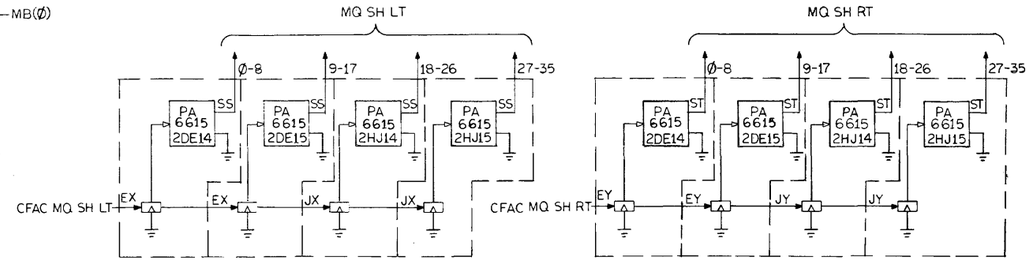
NOTE:

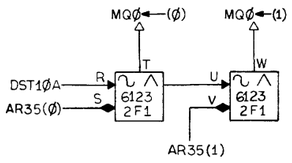
1. ALL FF PACKAGES ARE 6205.
2. * INDICATES REAR CONNECTOR PIN.
3. GROUND PIN D,P AND Z IN ALL 6205's AND 6615s.

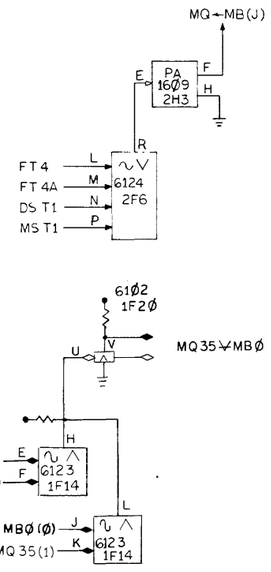
MQRT Register 18-35 (BS-D-166-0-MQ-3)

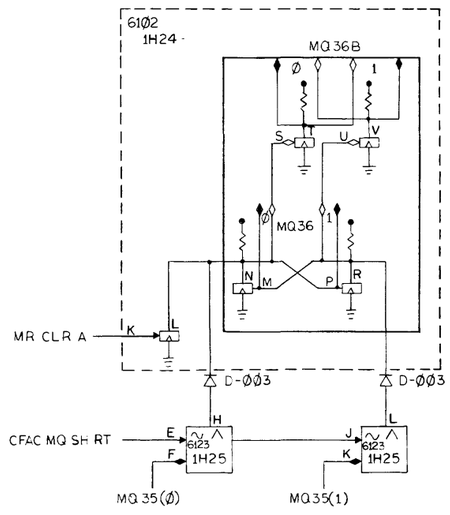
MQ - Multiplier Quotient Control
(BS-D-166-0-MQ-1)

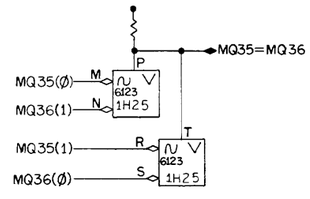






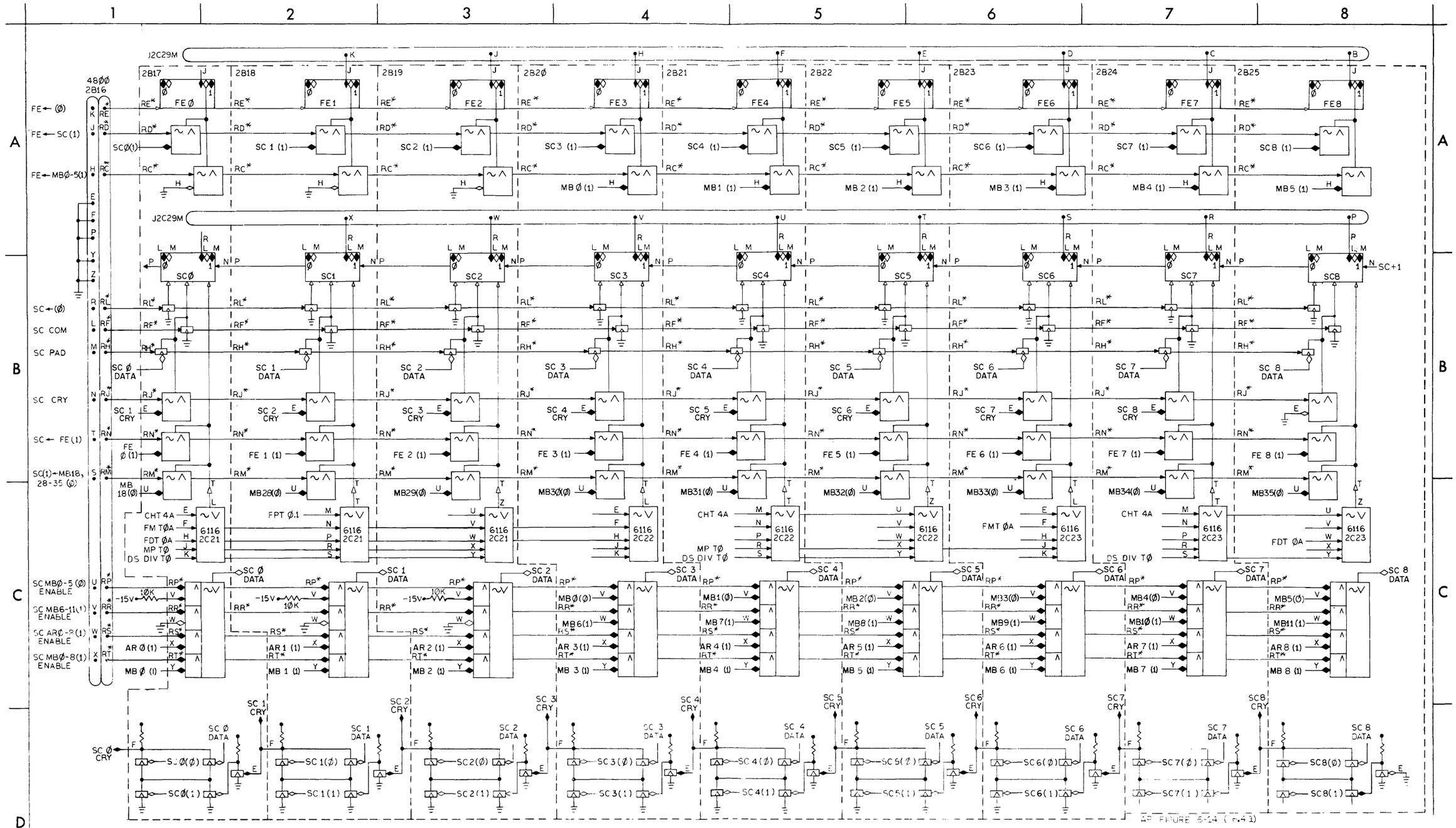






MQ - Multiplier Quotient Control
(BS-D-166-0-MQ-1)

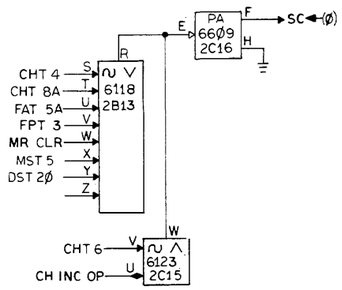
SC, FE - Shift Counter, Floating Exponent
(BS-D-166-0-SCFE)

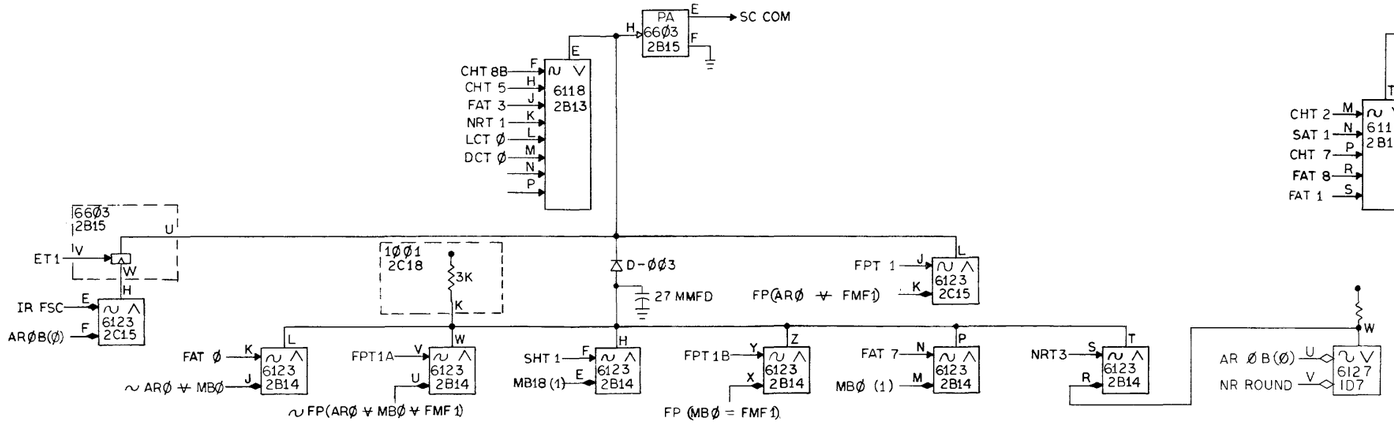


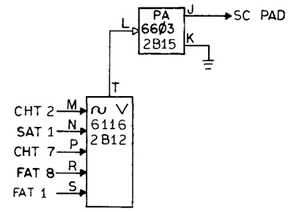
NOTE:

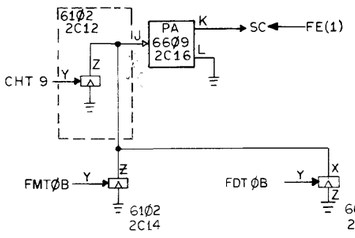
1. UNLESS OTHERWISE INDICATED ALL PACKAGES ARE 6203s.
2. * INDICATES REAR CONNECTOR PIN.

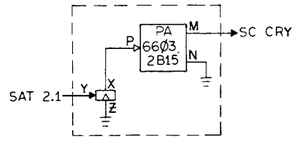
SC, FE - Shift Counter, Floating Exponent
(BS-D-166-0-SCFE)

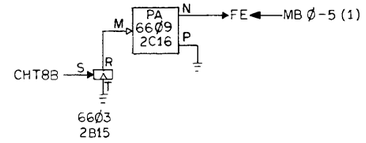


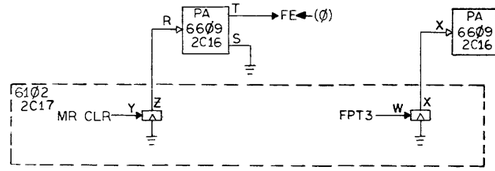


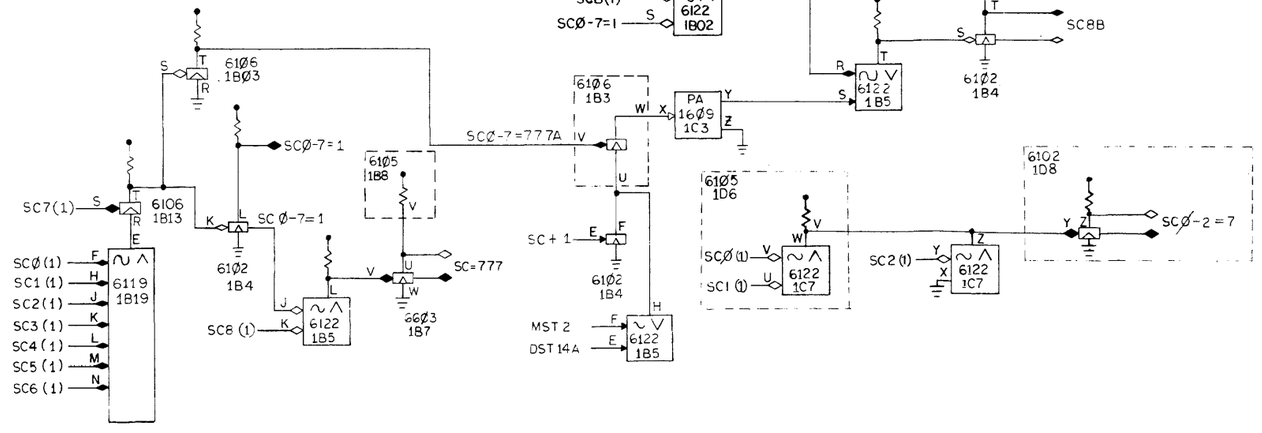


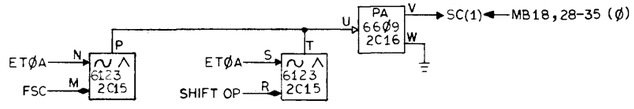






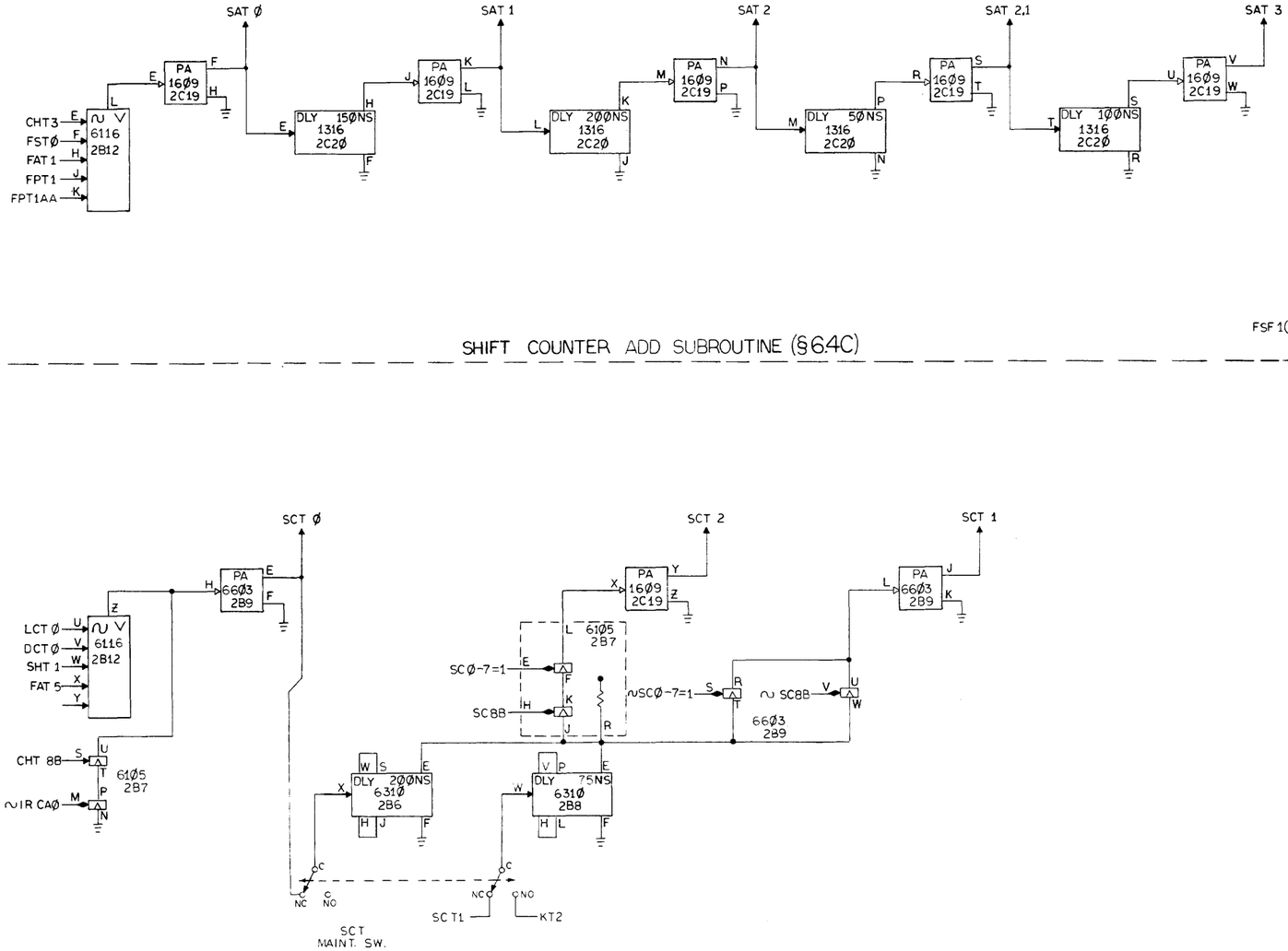


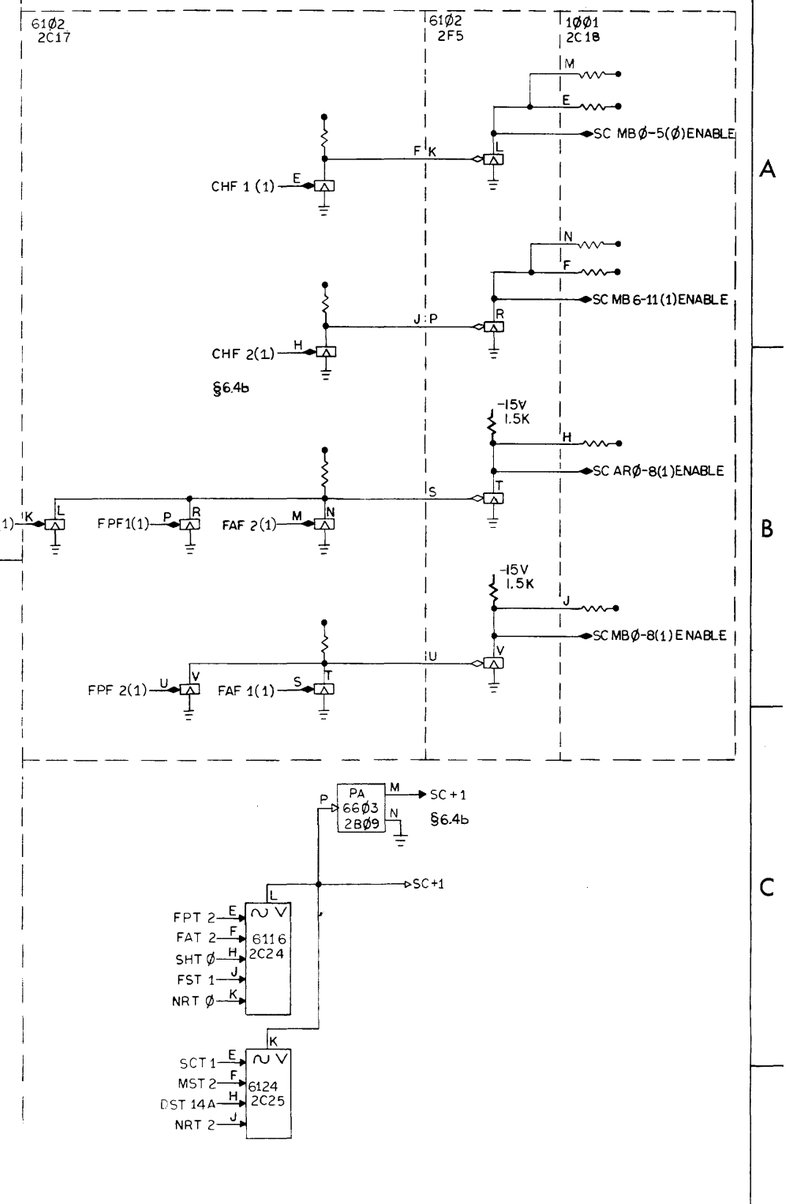




BKE-2

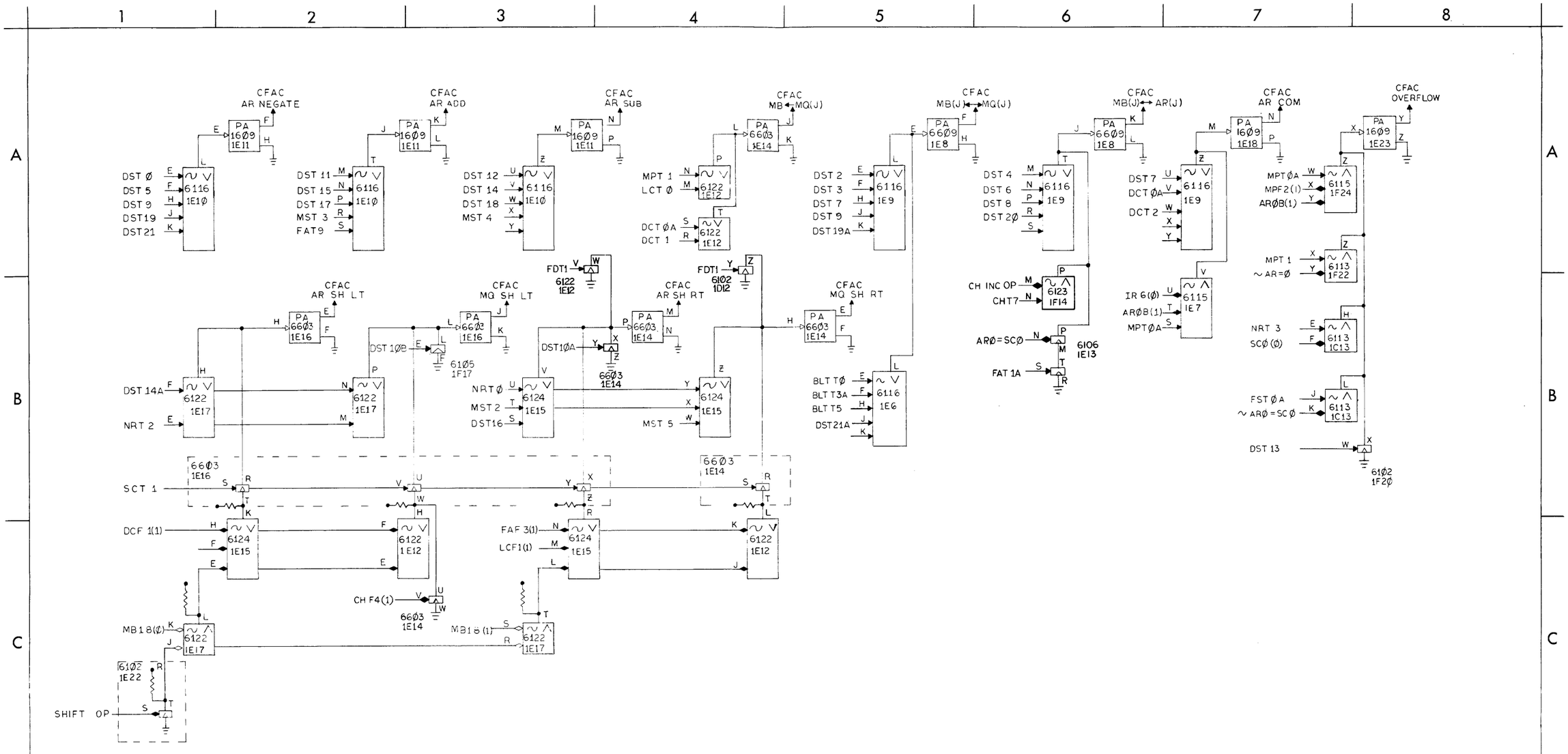
SC Control (BS-D-166-0-SC-1)





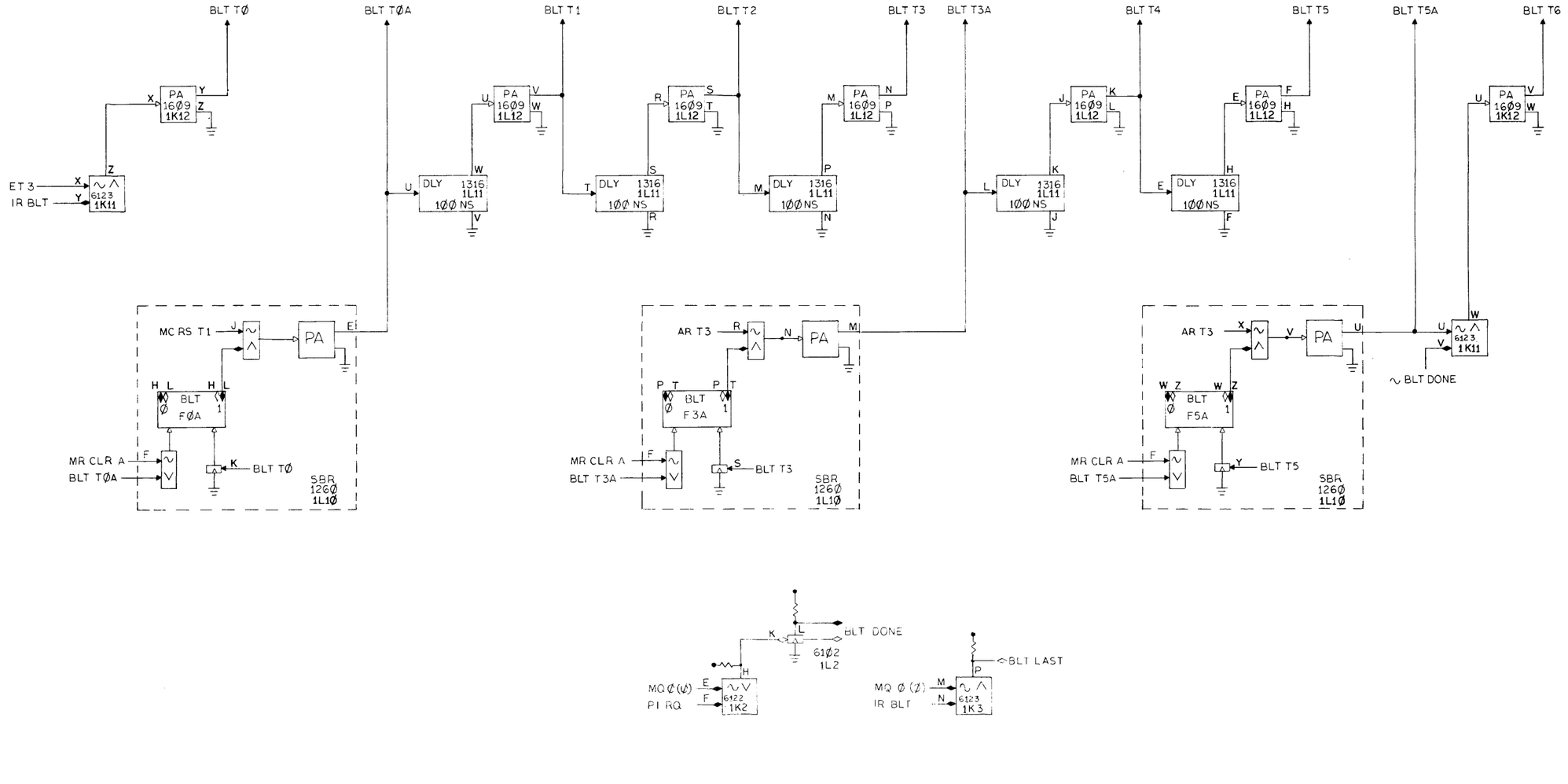
SC Control (BS-D-166-0-SC-2)

CFAC - Floating Arithmetic Connection
(BS-D-166-0-CFAC)



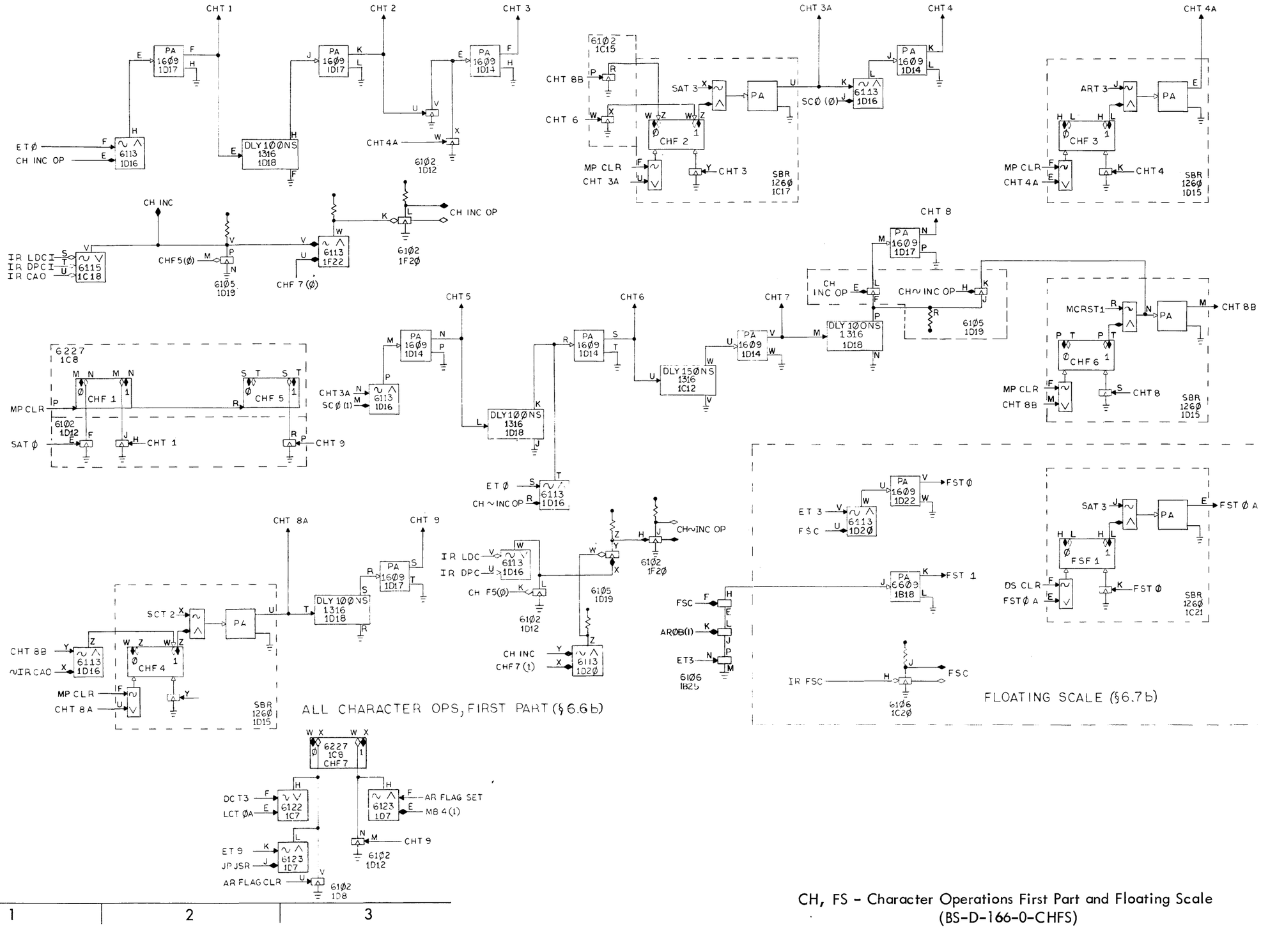
CFAC - Floating Arithmetic Connection
(BS-D-166-0-CFAC)

BLT - Block Transfer
(BS-D-166-0-BLT)



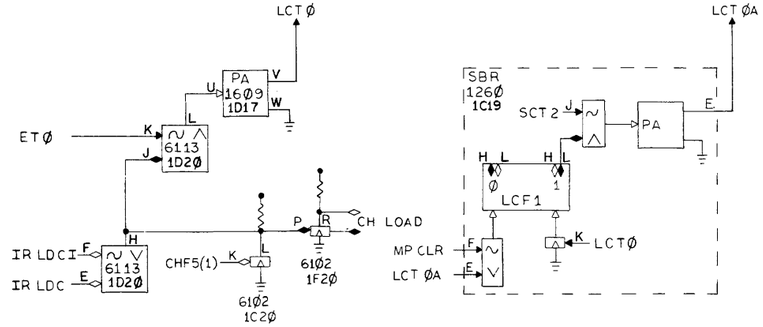
BLT - Block Transfer
(BS-D-166-0-BLT)

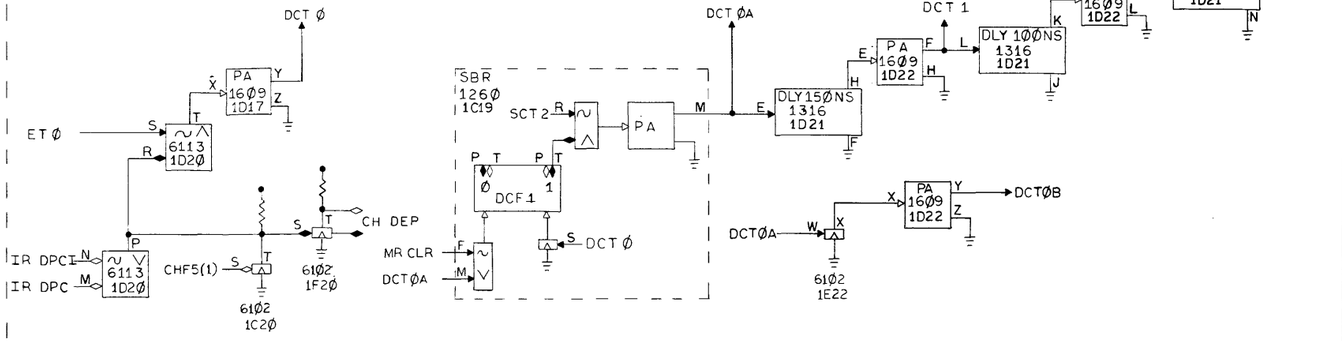
CH, FS - Character Operations First Part and Floating Scale
(BS-D-166-0-CHFS)



CH, FS - Character Operations First Part and Floating Scale
(BS-D-166-0-CHFS)

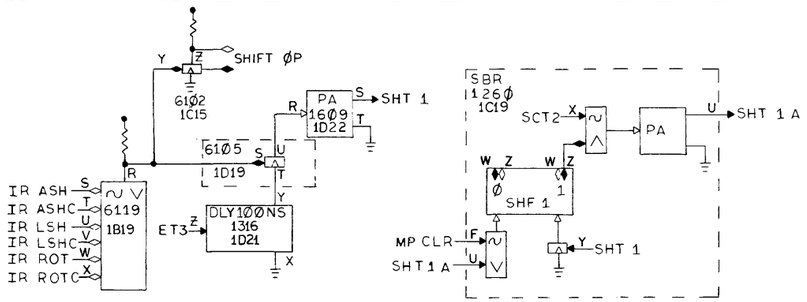
LC, DC, SH - Character Operations Second Part
and Shift Operations
(BS-D-166-0-LDCS)

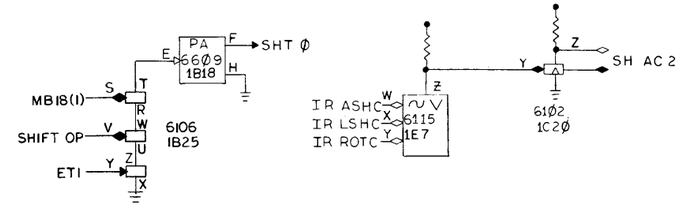




LOAD CHARACTER SECOND PART

DEPOSIT CHARACTER SECOND PART

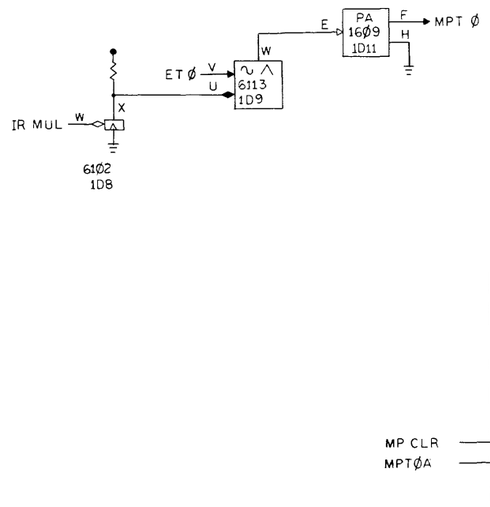


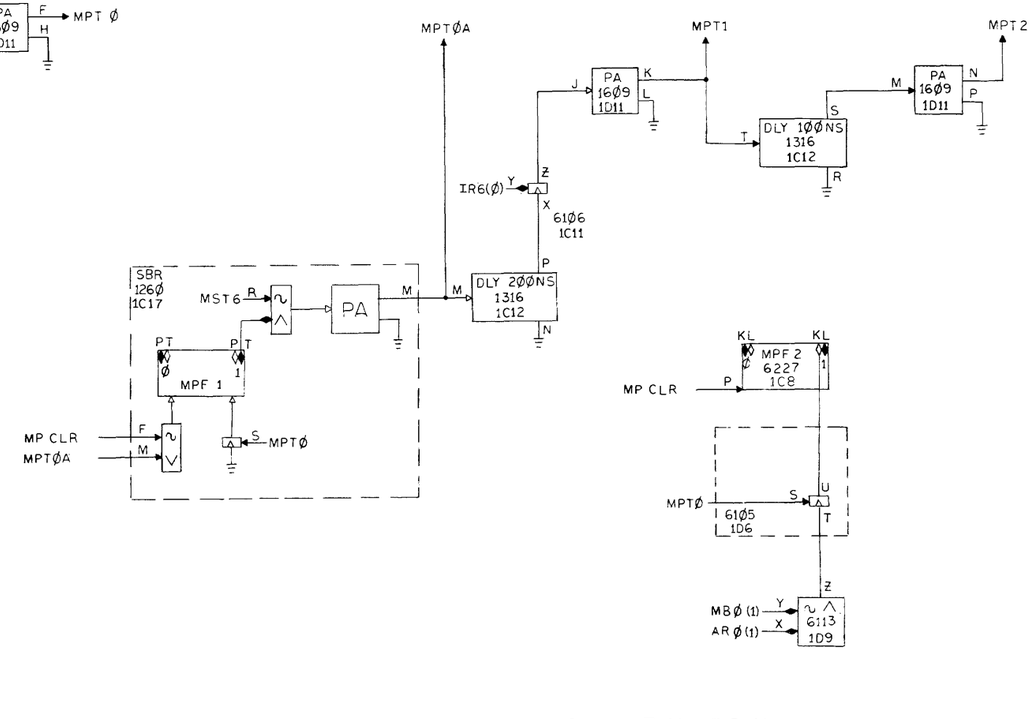


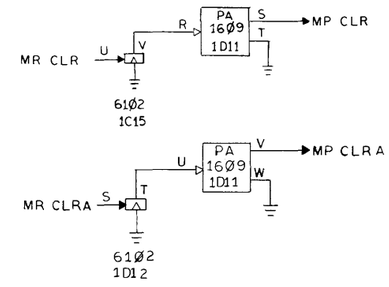
SHIFT OPERATIONS

LC, DC, SH - Character Operations Second Part
and Shift Operations
(BS-D-166-0-LDCS)

MP - Fixed Point Multiply
(BS-D-166-0-MP)

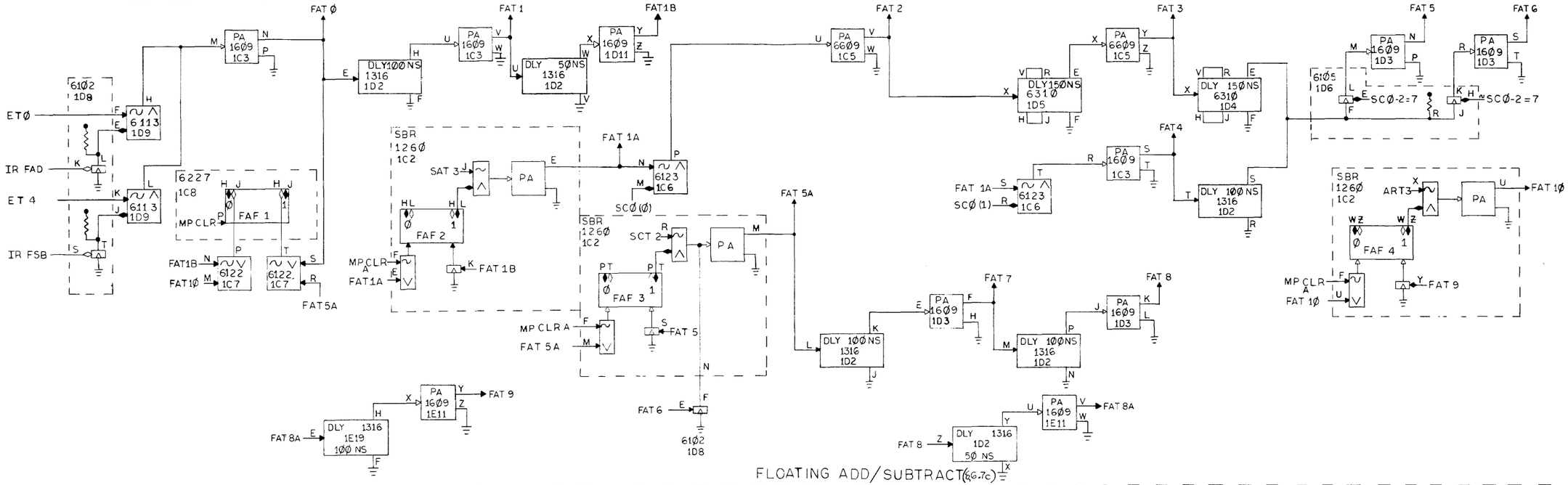




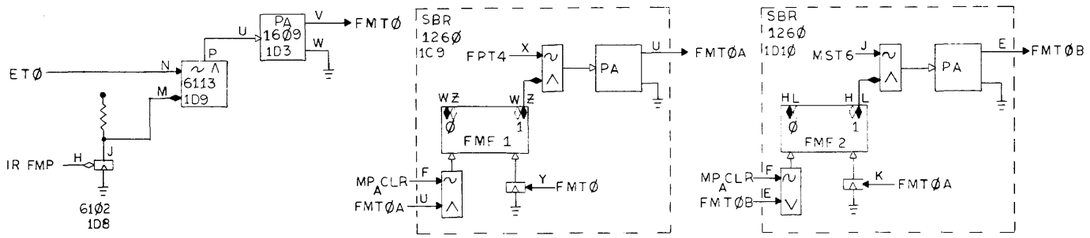


FIXED POINT MULTIPLY

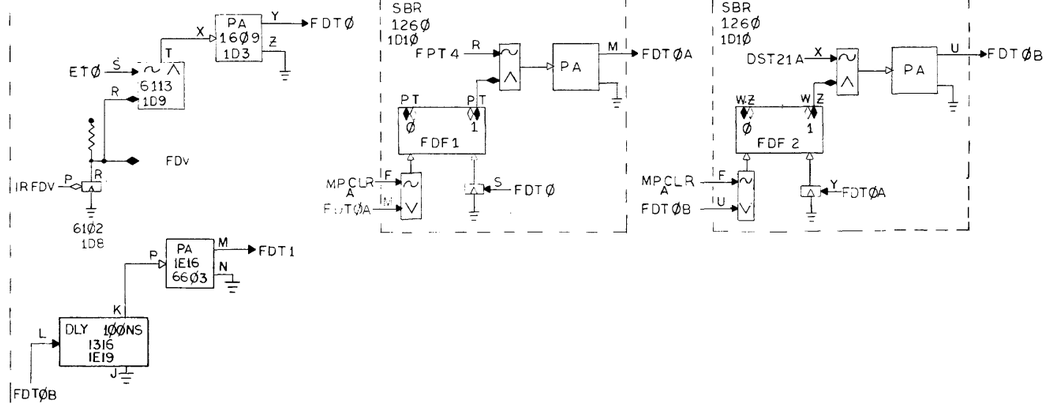
MP - Fixed Point Multiply
(BS-D-166-0-MP)



FLOATING ADD/SUBTRACT(§6.7c)



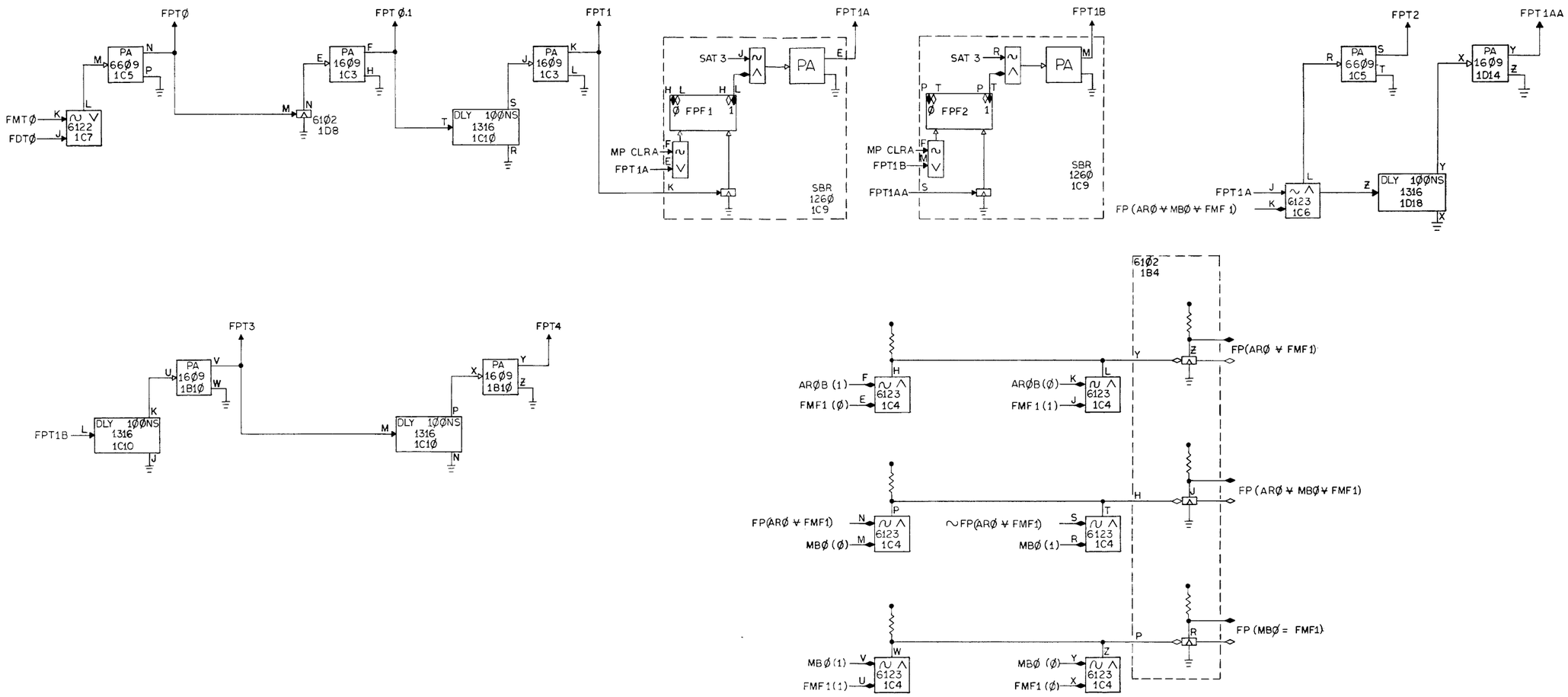
FLOATING MULTIPLY(§6.7d)



FLOATING DIVIDE(§6.7d)

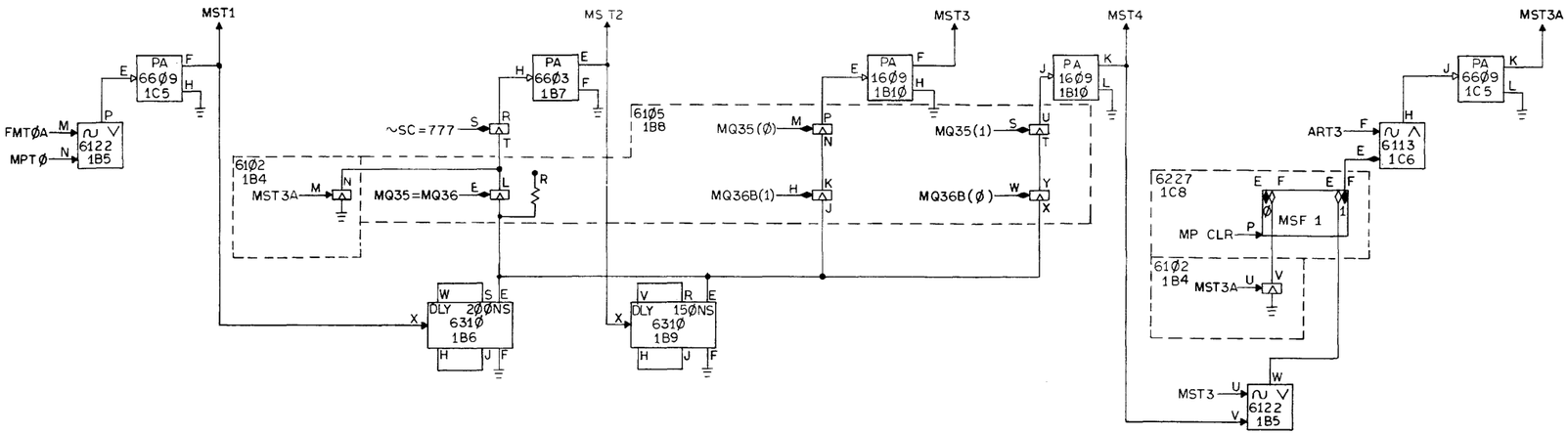
FA, FD, FM - Floating Point Instructions
(BS-D-166-0-FADM)

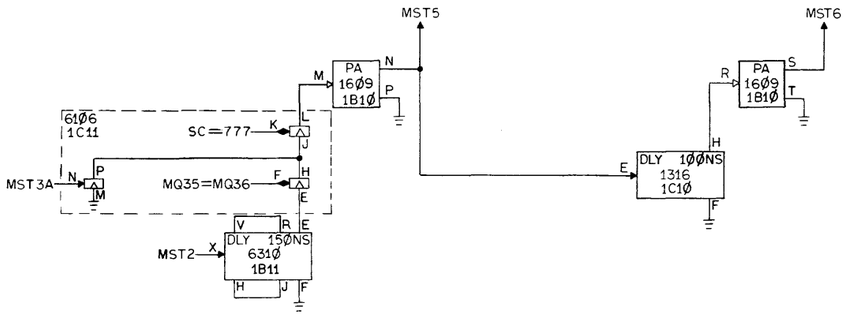
FP - Exponent Calculate Subroutine
(BS-D-166-0-FP)

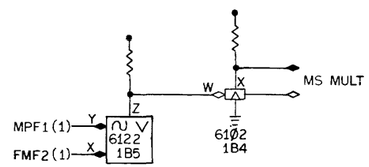


FP - Exponent Calculate Subroutine
(BS-D-166-0-FP)

MS - Multiply Subroutine (BS-D-166-0-MS)

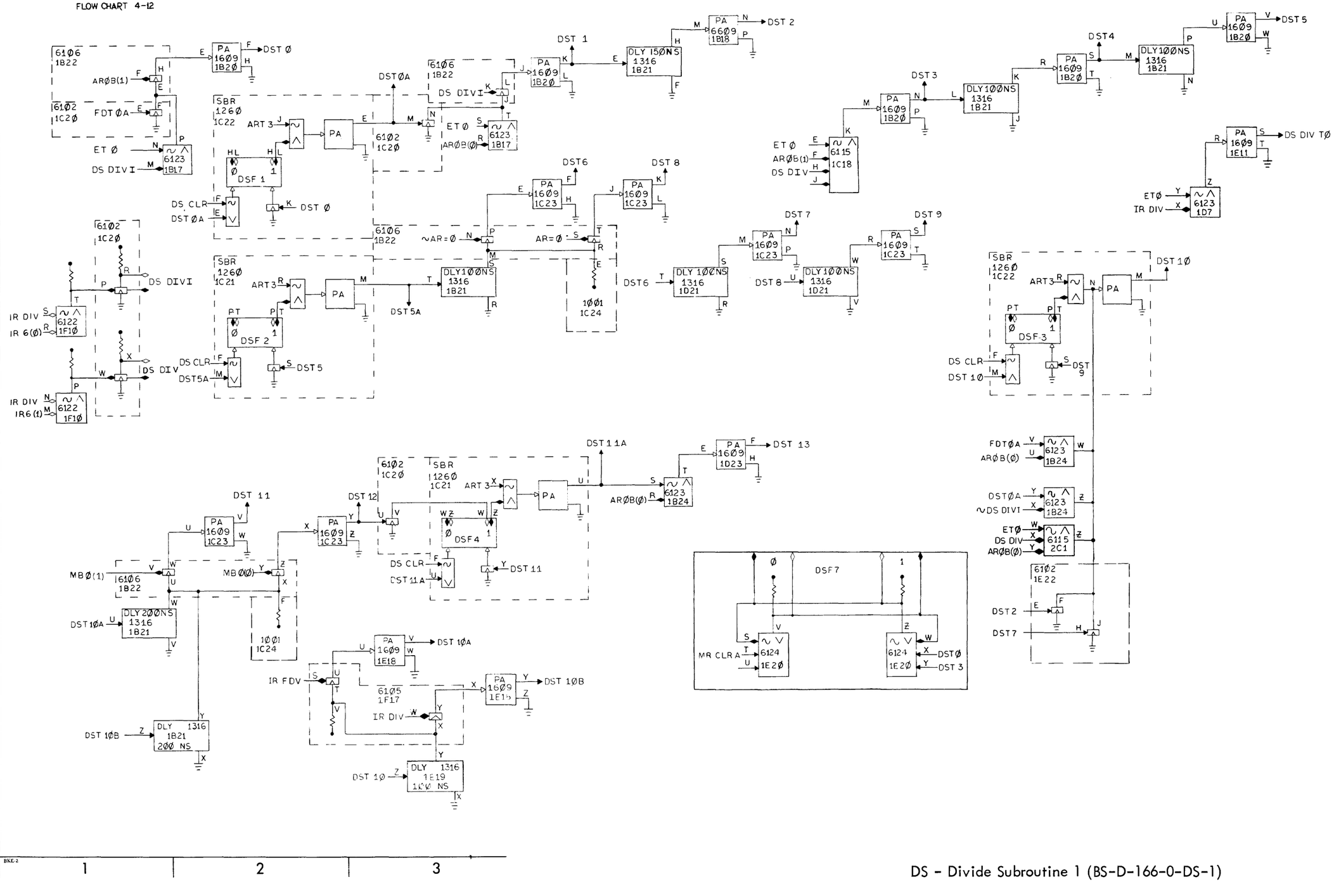






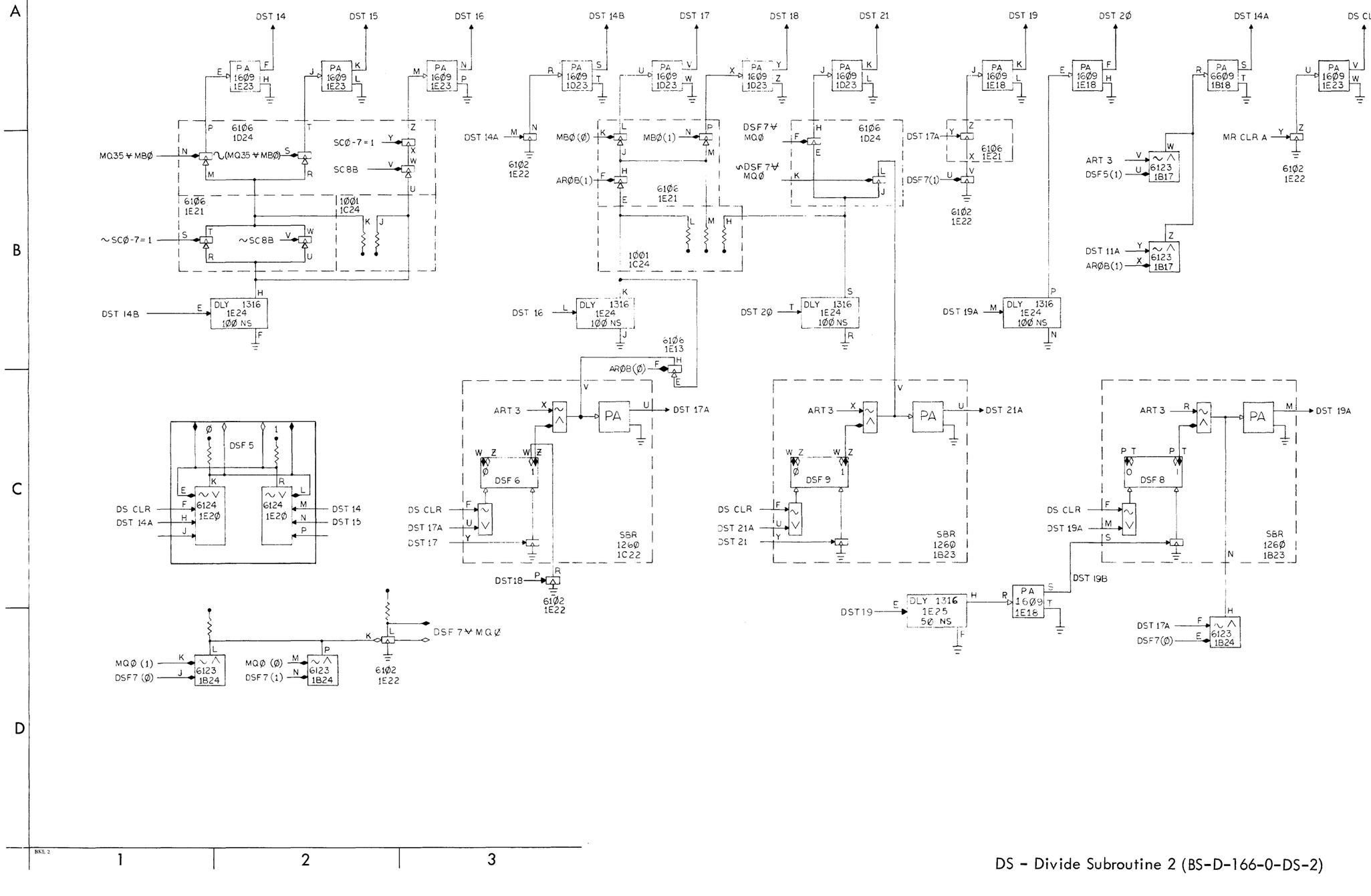
MS - Multiply Subroutine (BS-D-166-0-MS)

FLOW CHART 4-12



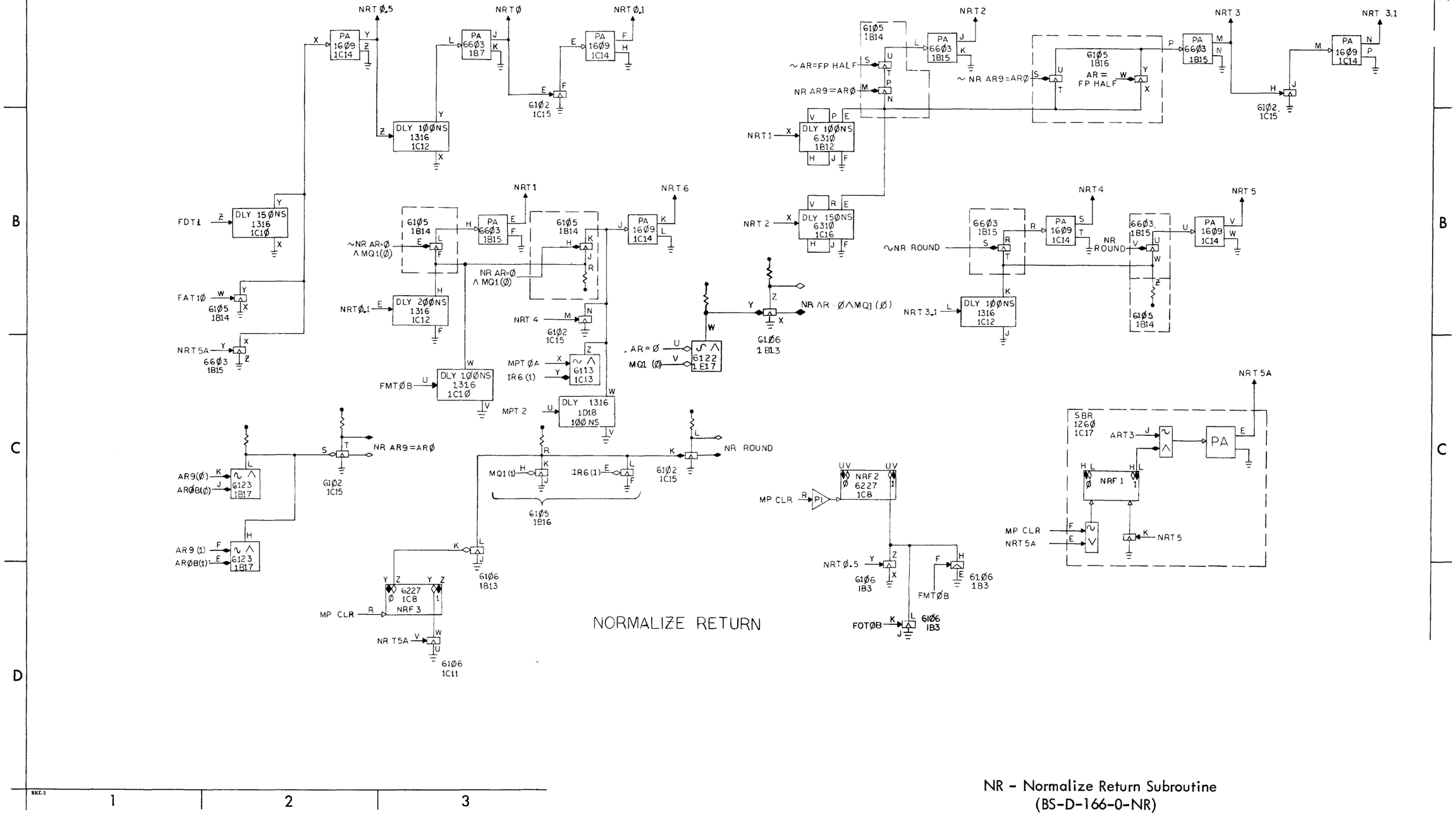
DS - Divide Subroutine 1 (BS-D-166-0-DS-1)

DS - Divide Subroutine 2 (BS-D-166-0-DS-2)



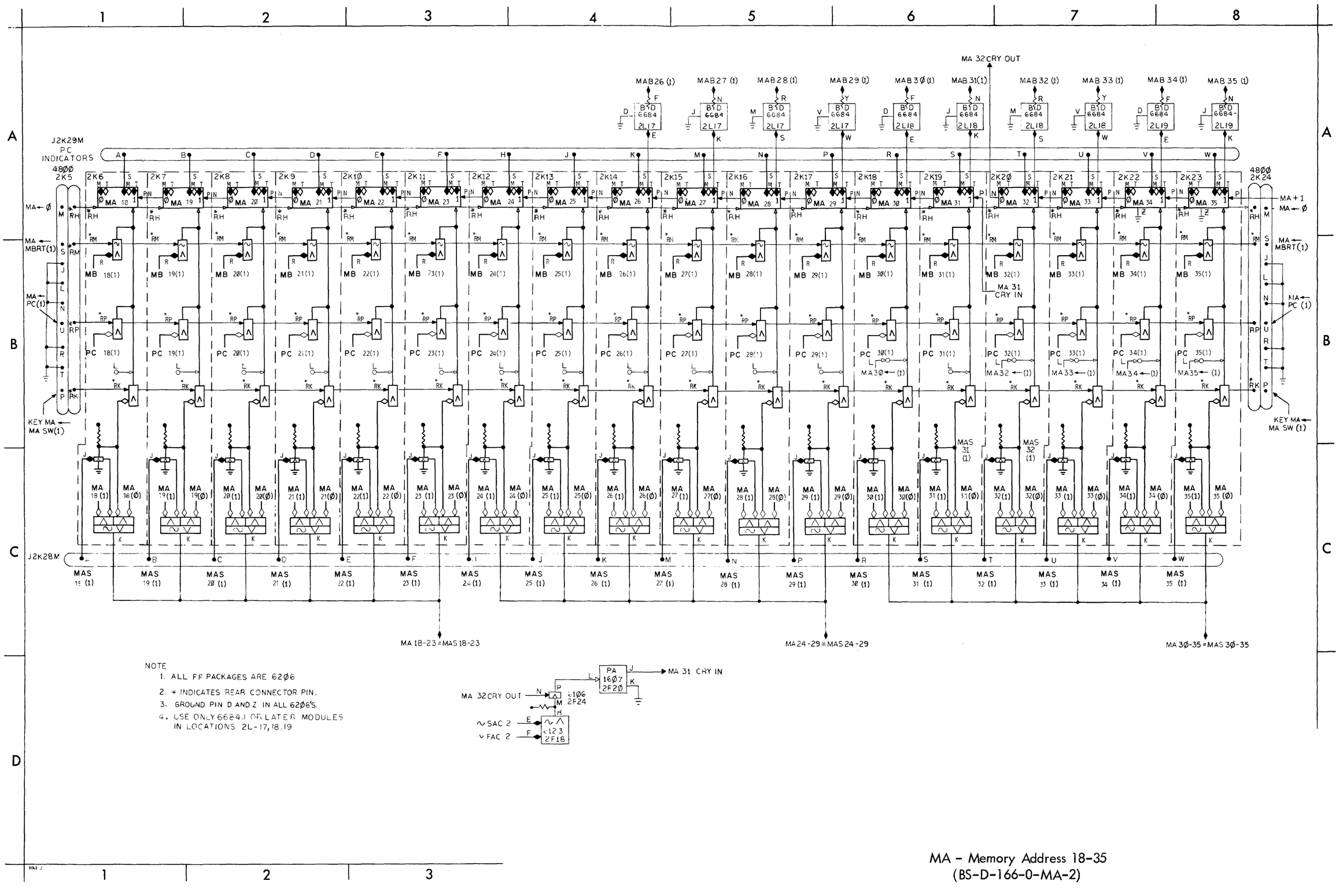
DS - Divide Subroutine 2 (BS-D-166-0-DS-2)

NR - Normalize Return Subroutine
(BS-D-166-0-NR)



NR - Normalize Return Subroutine
(BS-D-166-0-NR)

MA - Memory Address 18-35
(BS-D-166-0-MA-2)

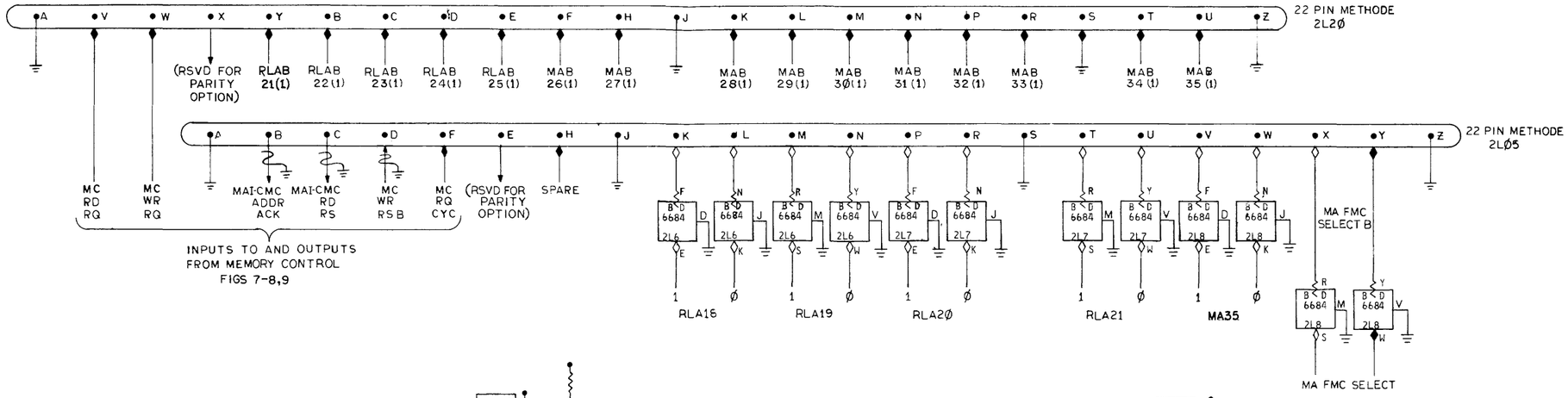


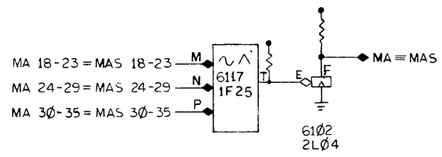
NOTE

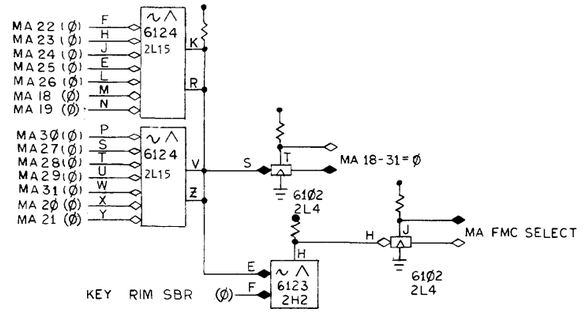
1. ALL FF PACKAGES ARE 62Ø6
2. * INDICATES REAR CONNECTOR PIN.
3. GROUND PIN D AND Z IN ALL 62Ø6'S.
4. USE ONLY 6684J OR LATER MODULES IN LOCATIONS 2L-17, 18, 19

MA – Memory Address 18-35
(BS-D-166-0-MA-2)

MA, MAI - Memory Address Interface
(BS-D-166-0-MA-3)

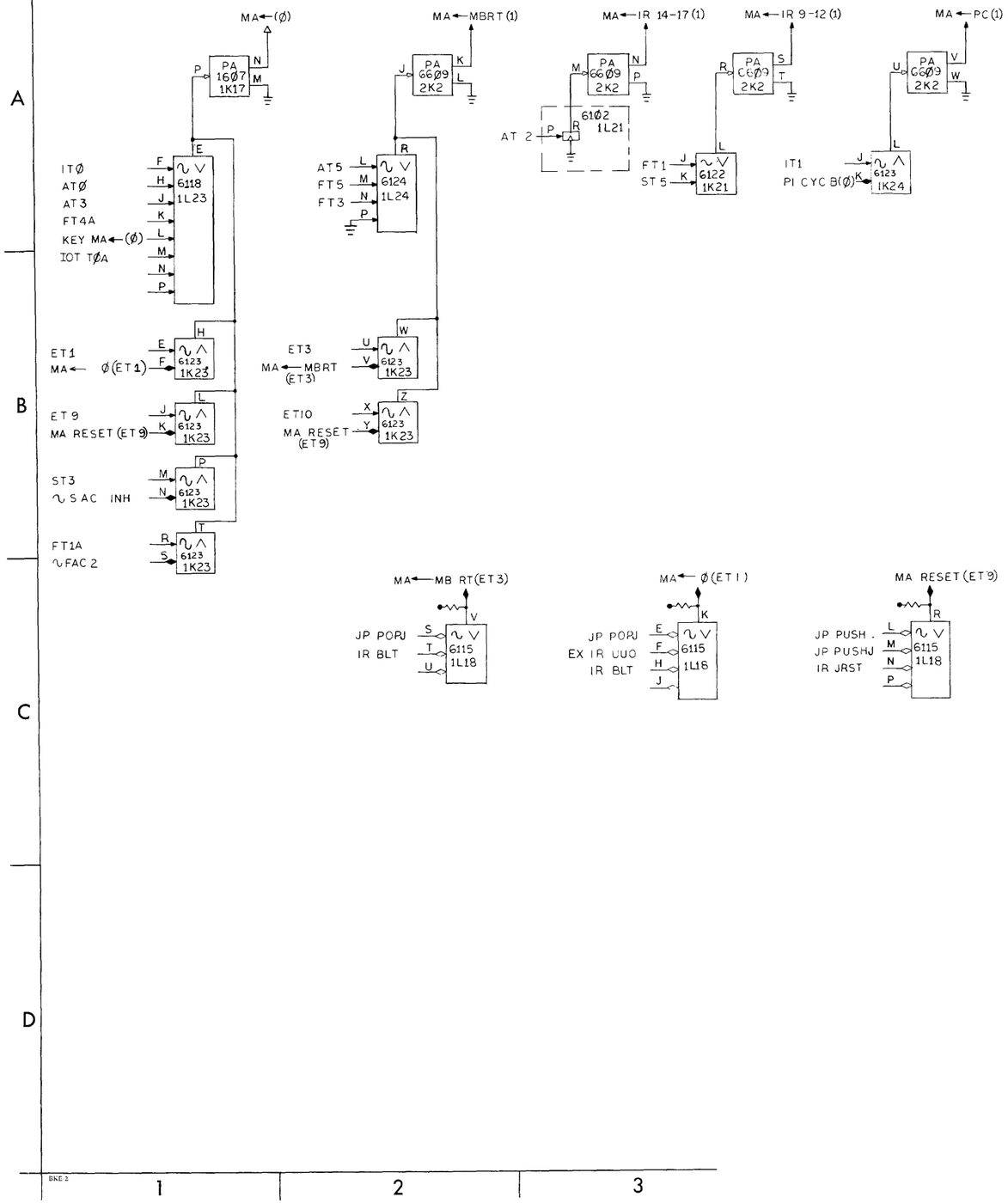




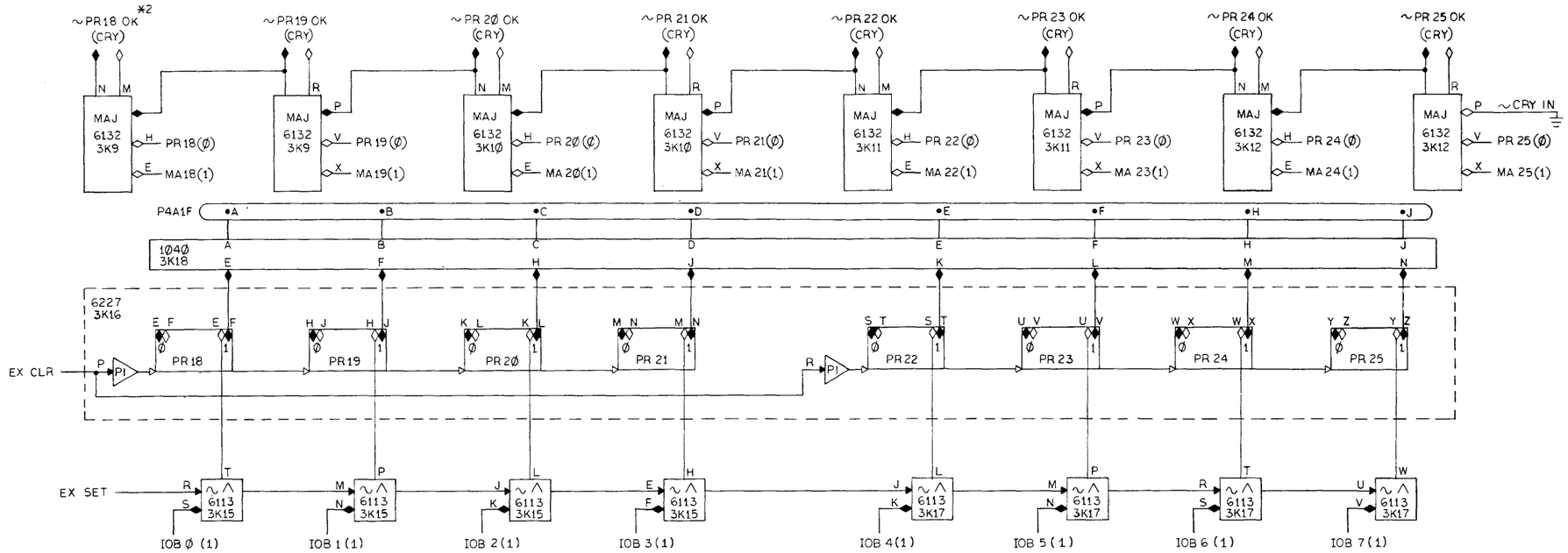


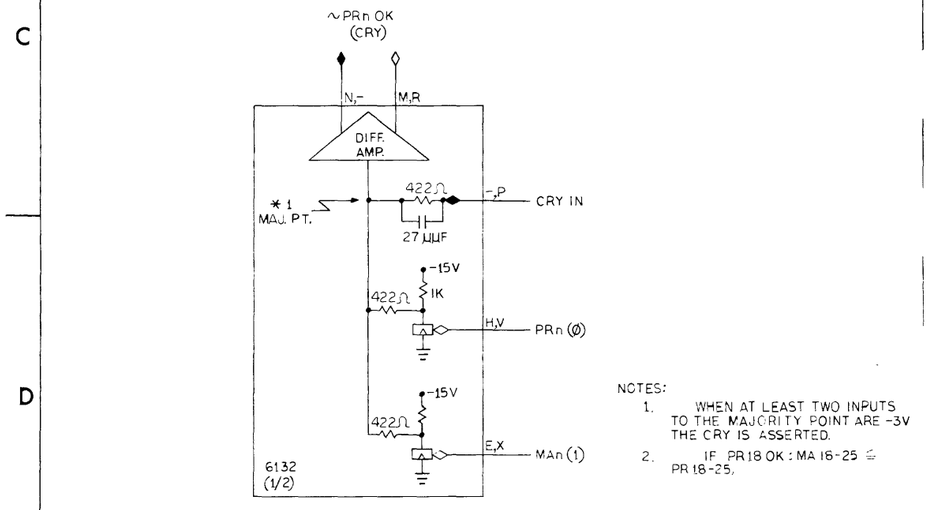
NOTE: 1 USE ONLY 6684J OR LATER MODULES IN LOCATIONS 2L-6,7,8

MA, MAI - Memory Address Interface
(BS-D-166-0-MA-3)

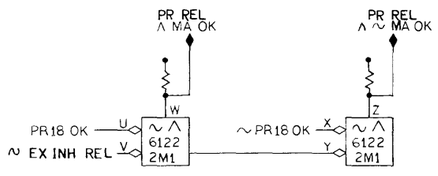


MA Control (BS-D-166-0-MA-1)



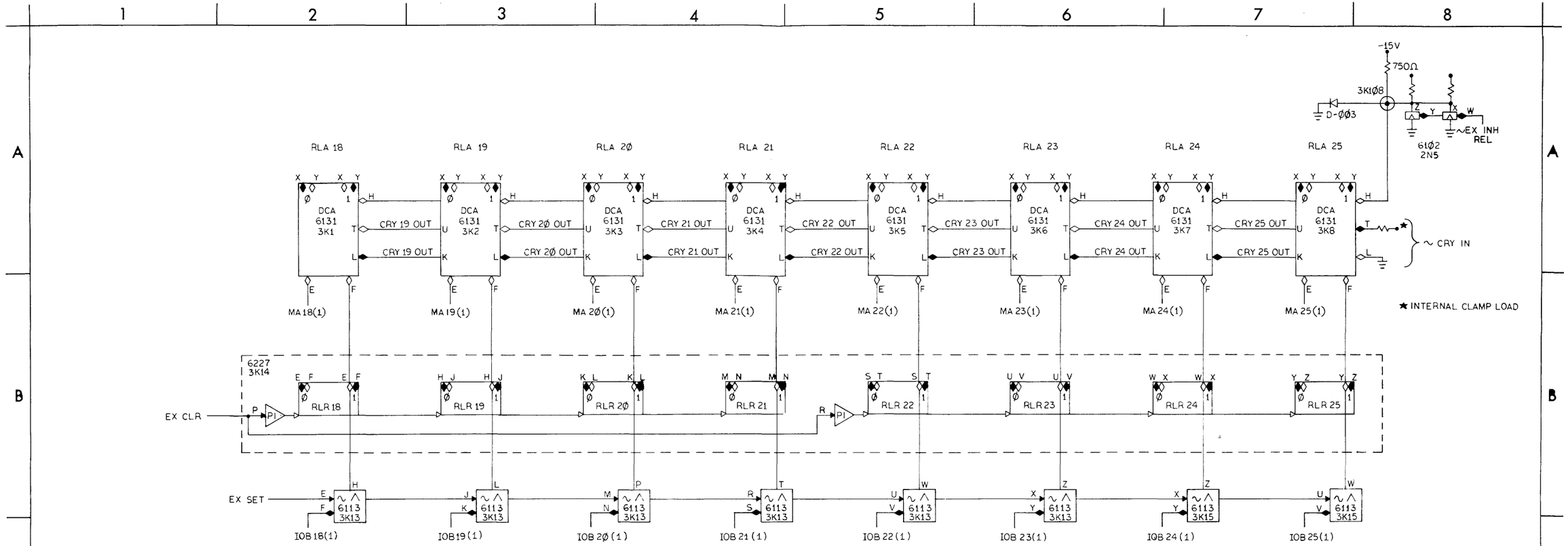


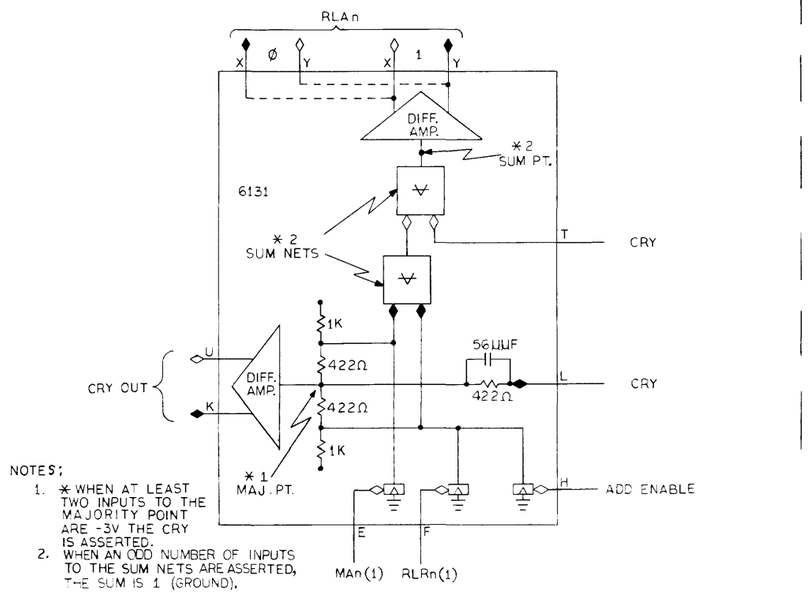
NOTES:
1. WHEN AT LEAST TWO INPUTS TO THE MAJORITY POINT ARE -3V THE CRY IS ASSERTED.
2. IF PR18 OK : MA 18-25 ≙ PR 18-25.

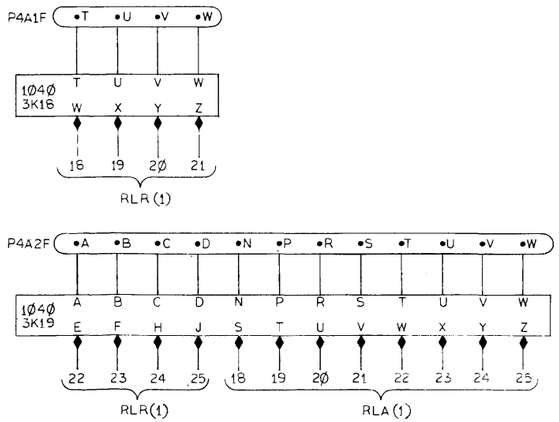


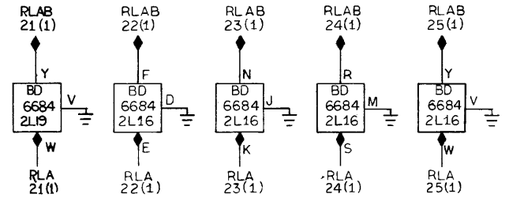
PR - Protection (BS-D-166-0-PR)

RLA, RLR - Relocation Adder and Register
(BS-D-166-0-RLAR)



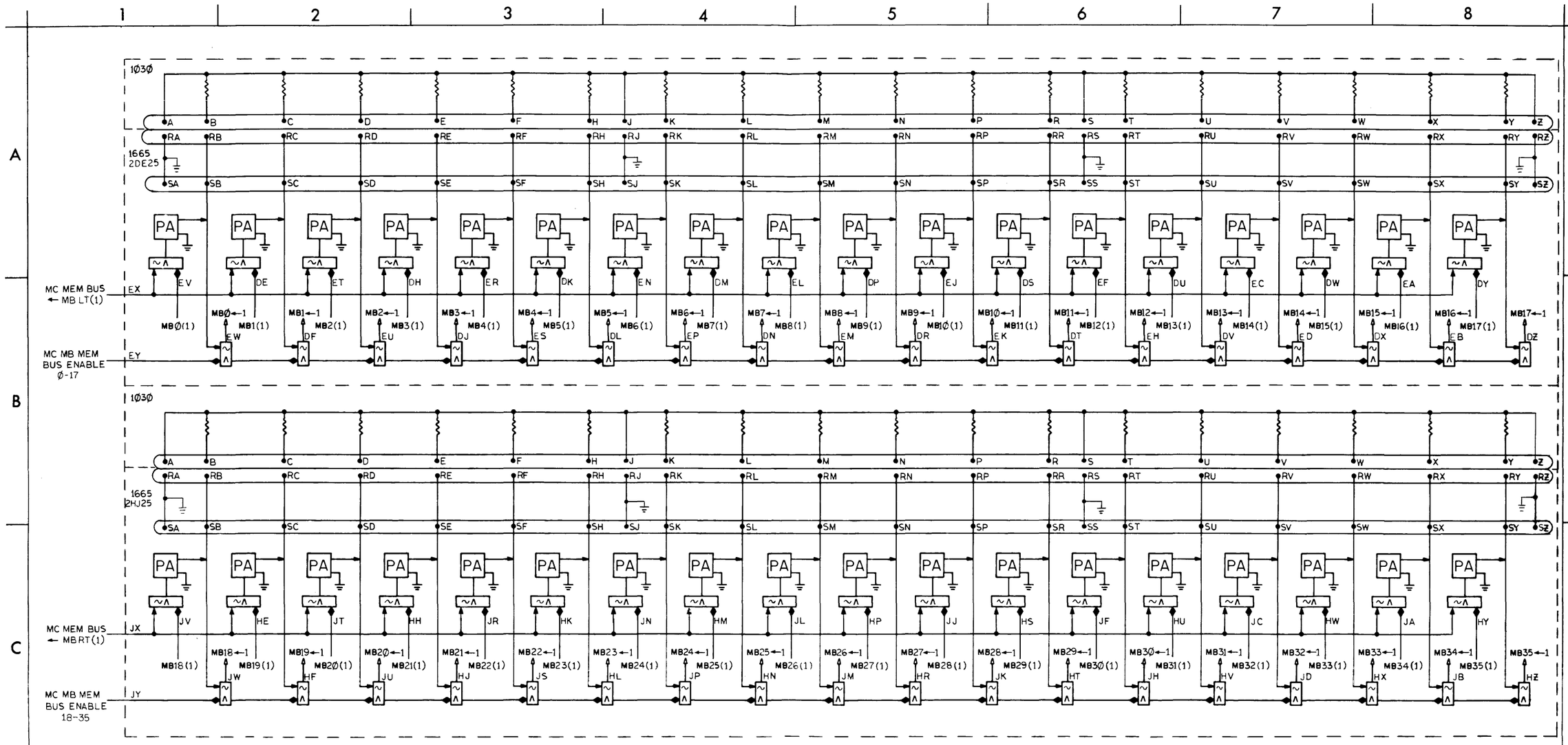






RLA, RLR - Relocation Adder and Register
(BS-D-166-0-RLAR)

MB - Memory Buffer Interface
(BS-D-166-0-MB-4)

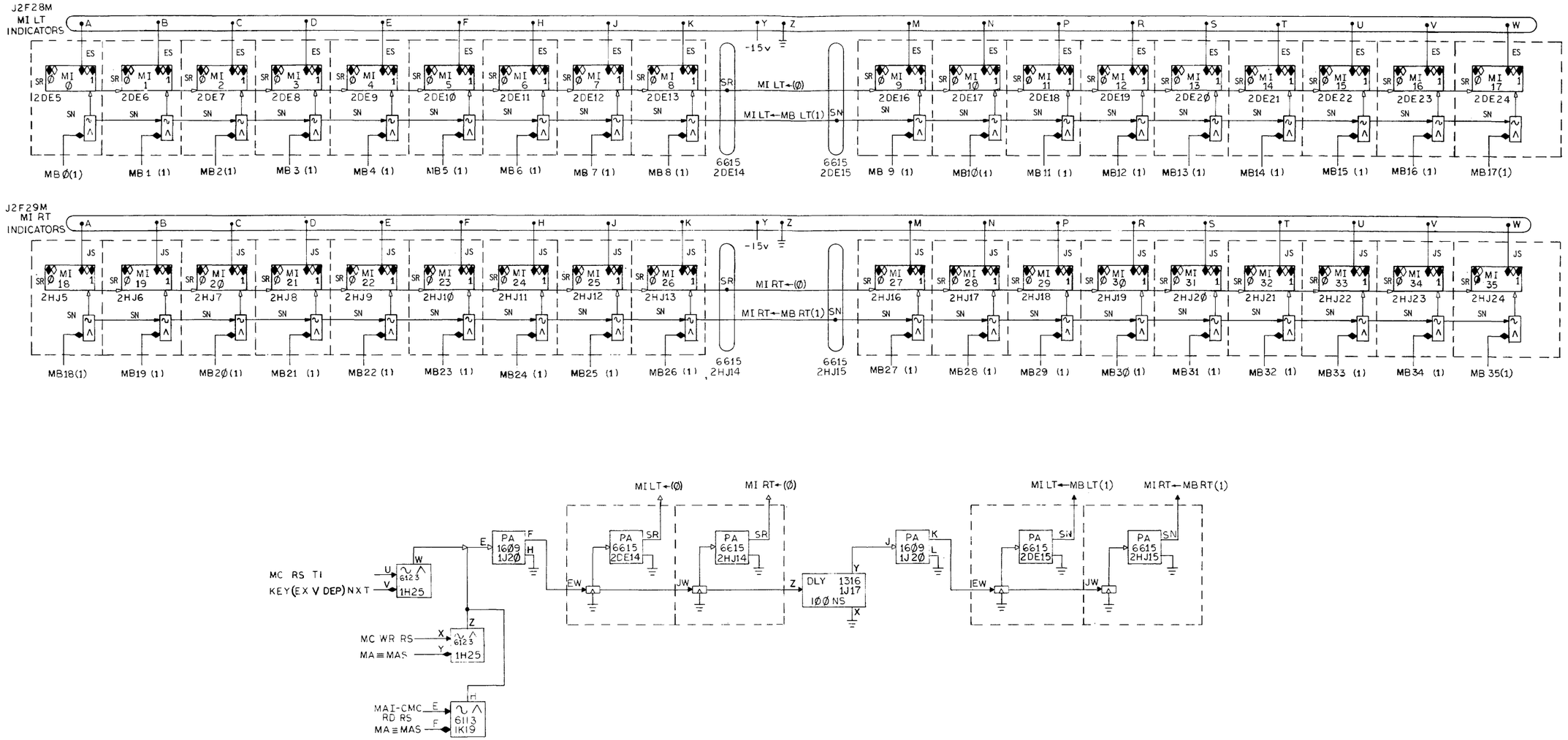


NOTE:

ALL TERMINATOR RES. IN THE
1Ø3Ø MODULES ARE 1ØØ OHMS.

MB - Memory Buffer Interface
(BS-D-166-0-MB-4)

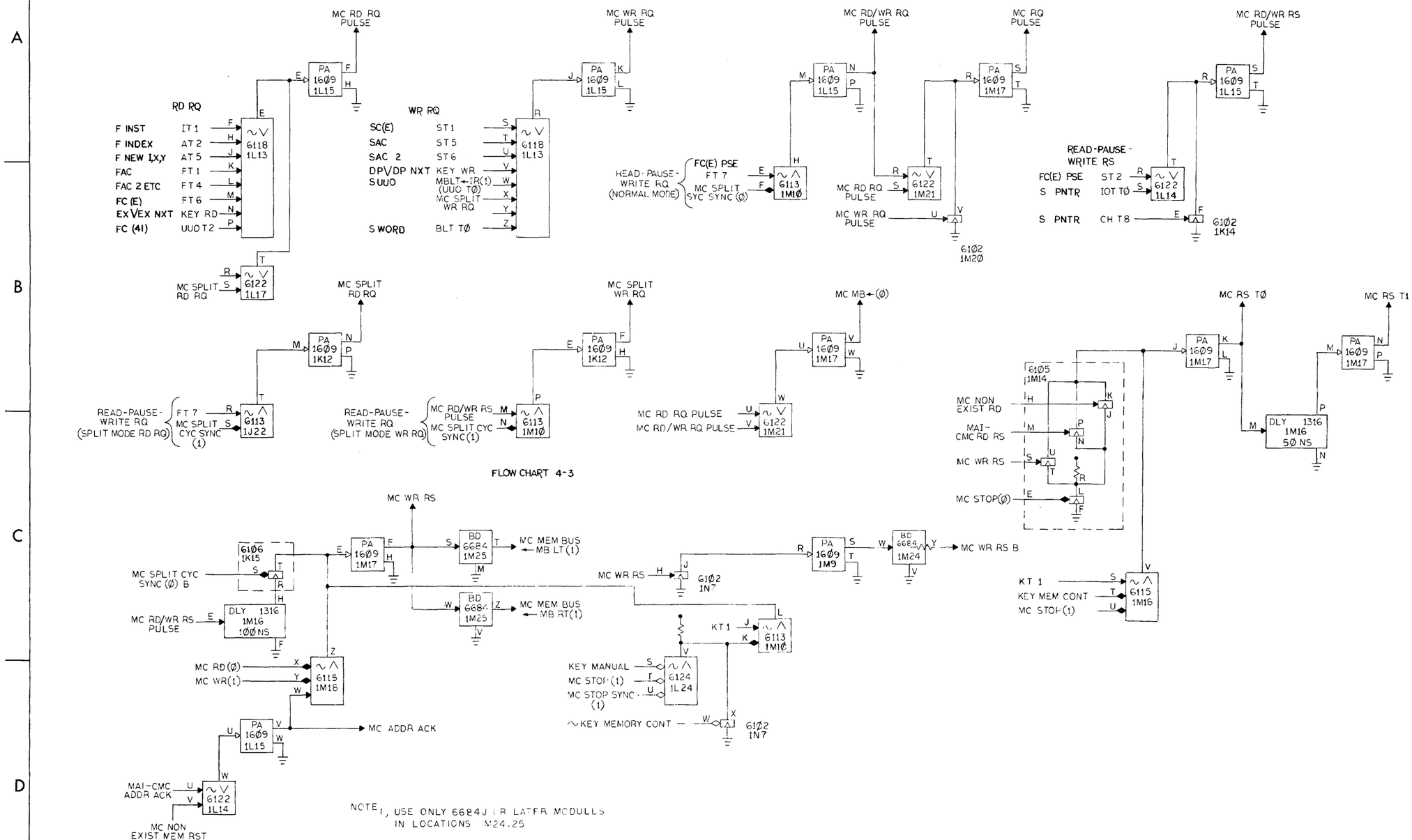
MI - Memory Indicators (BS-D-166-0-MI)



NOTE:

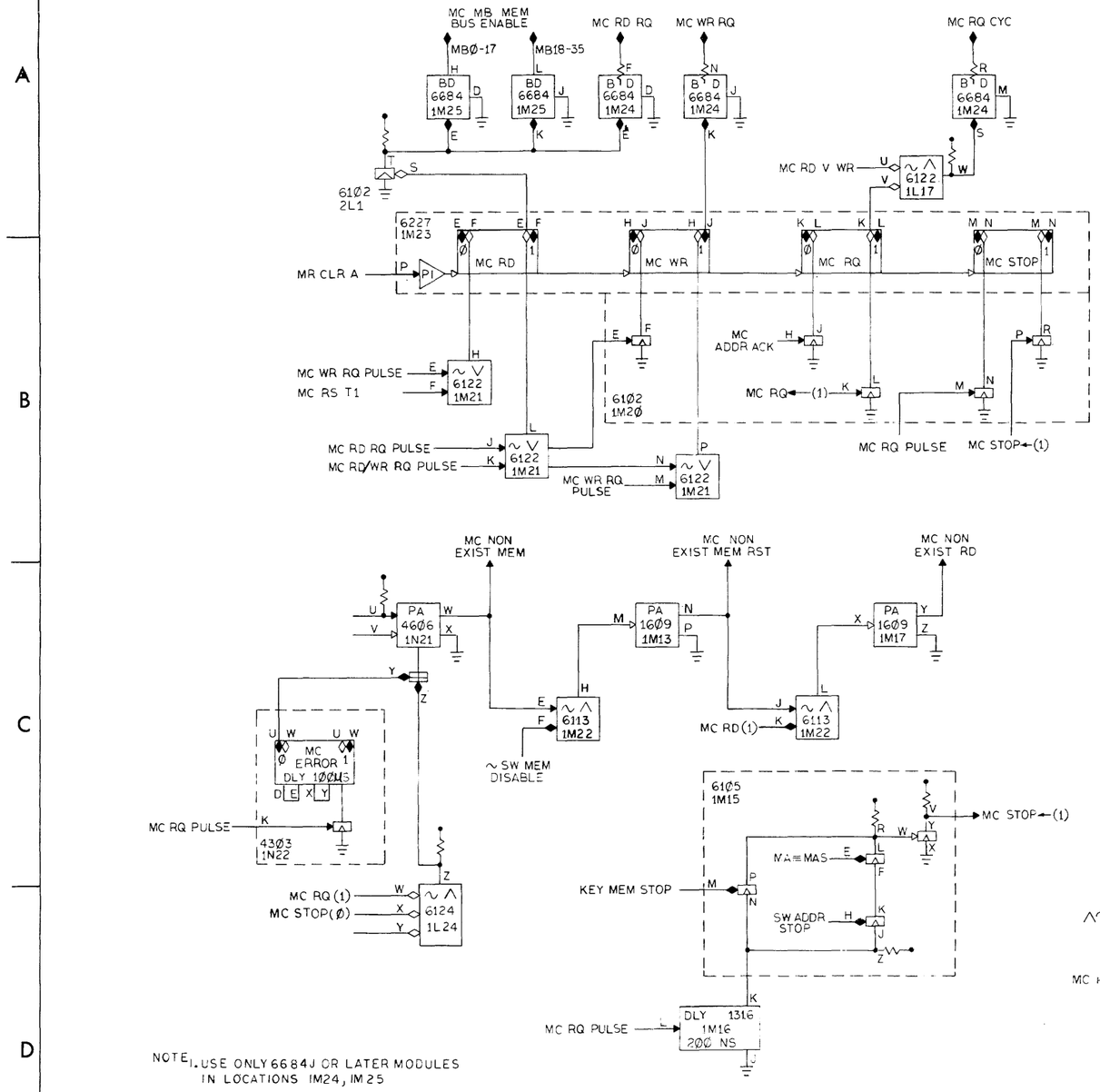
1. ALL FF PACKAGES ARE 62Ø5
2. SN AND SR ARE PINS IN LOWER REAR CONNECTOR

MC - Memory Control (BS-D-166-0-MC-1)



MC - Memory Control (BS-D-166-0-MC-1)

MC - Memory Control (BS-D-166-0-MC-2)

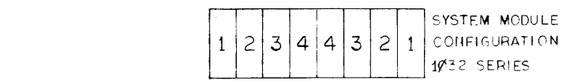


MC - Memory Control (BS-D-166-0-MC-2)

In-out and Memory Busses (CD-D-166-0-IOMB)

IO BUS(§8.4)

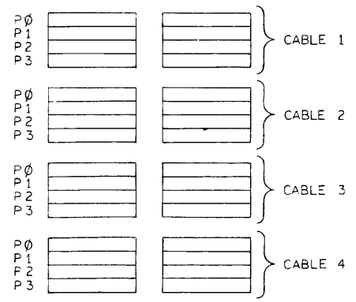
FLIP CHIP PIN		SYSTEM MODULE PIN	IO CABLE 1	IO CABLE 2	IO CABLE 3	IO CABLE 4
A	C	A	GND	GND	GND	GND
	D	B	IOB Ø (1) —◇	IOB 18 (1) —◇	IOB RESET →	DATAO CLEAR →
	E	C	IOB 1 (1) —◇	IOB 19 (1) —◇	POWER ON -15V	DATAO SET →
	H	D	IOB 2 (1) —◇	IOB 2Ø (1) —◇		CONO CLEAR →
	K	E	IOB 3 (1) —◇	IOB 21 (1) —◇	MC DR SPLIT —◇	CONO SET →
	M	F	IOB 4 (1) —◇	IOB 22 (1) —◇	IOS 3 (Ø) —◇	IOB ← DATAI —◆
	P	H	IOB 5 (1) —◇	IOB 23 (1) —◇	IOS 3 (1) —◇	IOB ← STATUS —◆
B	N	J	GND	GND	GND	GND
	S	K	IOB 6 (1) —◇	IOB 24 (1) —◇	IOS 4 (Ø) —◇	
	T	L	IOB 7 (1) —◇	IOB 25 (1) —◇	IOS 4 (1) —◇	
	V	M	IOB 8 (1) —◇	IOB 26 (1) —◇	IOS 5 (Ø) —◇	
	D	N	IOB 9 (1) —◇	IOB 27 (1) —◇	IOS 5 (1) —◇	
	E	P	IOB 1Ø (1) —◇	IOB 28 (1) —◇	IOS 6 (Ø) —◇	
	H	R	IOB 11 (1) —◇	IOB 29 (1) —◇	IOS 6 (1) —◇	PI REQ 1 —◇
	J	S	GND	GND	GND	GND
B	K	T	IOB 12 (1) —◇	IOB 3Ø (1) —◇	IOS 7 (Ø) —◇	PI REQ 2 —◇
	M	U	IOB 13 (1) —◇	IOB 31 (1) —◇	IOS 7 (1) —◇	PI REQ 3 —◇
	P	V	IOB 14 (1) —◇	IOB 32 (1) —◇	IOS 8 (Ø) —◇	PI REQ 4 —◇
	S	W	IOB 15 (1) —◇	IOB 33 (1) —◇	IOS 8 (1) —◇	PI REQ 5 —◇
	T	X	IOB 16 (1) —◇	IOB 34 (1) —◇	IOS 9 (Ø) —◇	PI REQ 6 —◇
	V	Y	IOB 17 (1) —◇	IOB 35 (1) —◇	IOS 9 (1) —◇	PI REQ 7 —◇
C	U	Z	GND	GND	GND	GND
			(SOURCE 3L6)	(SOURCE 3L7)	(SOURCE 2L13)	(SOURCE 2N25)



NOTE: 1 SYSTEM MODULE CONNECTOR (1Ø32) = 2 FLIP CHIP CONNECTORS (W028)

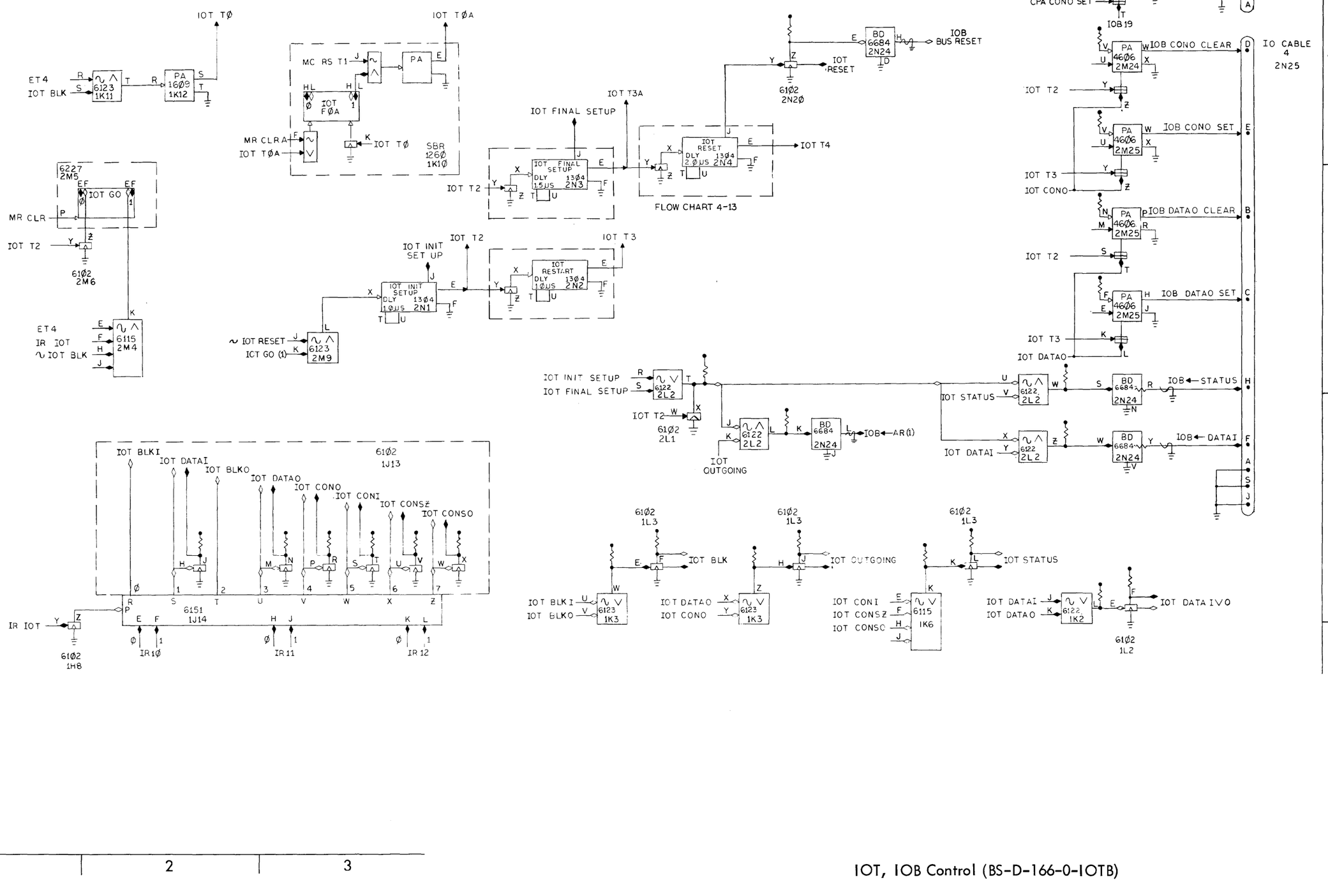
MEM BUS(§§7.1a, 7.2)

FLIP CHIP PIN		SYSTEM MODULE PIN	MEM CABLE 1	MEM CABLE 2	MEM CABLE 3	MEM CABLE 4
A	C	A	GND	GND	GND	GND
	D	B	ADDR ACK →	MA 22 (1) ◆→	MB Ø (1) →	MB 18 (1) →
	E	C	RD RS →	MA 23 (1) ◆→	MB 1 (1) →	MB 19 (1) →
	H	D	WR RS →	MA 24 (1) ◆→	MB 2 (1) →	MB 2Ø (1) →
	K	E	PAR (1) →	MA 25 (1) ◆→	MB 3 (1) →	MB 21 (1) →
	M	F	RQ CYCLE ◆→	MA 26 (1) ◆→	MB 4 (1) →	MB 22 (1) →
	P	H	SPARE	MA 27 (1) ◆→	MB 5 (1) →	MB 23 (1) →
	N	J	GND	GND	GND	GND
	S	K	MA 18 (1) —◇	MA 28 (1) ◆→	MB 6 (1) →	MB 24 (1) →
	T	L	MA 18 (Ø) —◇	MA 29 (1) ◆→	MB 7 (1) →	MB 25 (1) →
	V	M	MA 19 (1) —◇	MA 3Ø (1) ◆→	MB 8 (1) →	MB 26 (1) →
	D	N	MA 19 (Ø) —◇	MA 31 (1) ◆→	MB 9 (1) →	MB 27 (1) →
	E	P	MA 2Ø (1) —◇	MA 32 (1) ◆→	MB 1Ø (1) →	MB 28 (1) →
	H	R	MA 2Ø (Ø) —◇	MA 33 (1) ◆→	MB 11 (1) →	MB 29 (1) →
	J	S	GND	GND	GND	GND
B	K	T	MA 21 (1) —◇	MA 34 (1) ◆→	MB 12 (1) →	MB 30 (1) →
	M	U	MA 21 (Ø) —◇	MA 35 (1) ◆→	MB 13 (1) →	MB 31 (1) →
	P	V	MA 35 (1) —◇	MC RD RQ ◆→	MB 14 (1) →	MB 32 (1) →
	S	W	MA 35 (Ø) —◇	MC WR RQ ◆→	MB 15 (1) →	MB 33 (1) →
	T	X	MAEMC —◇	PAR OPTION →	MB 16 (1) →	MB 34 (1) →
	V	Y	SELECT B ◆→	MA 21 (1) ◆→	MB 17 (1) →	MB 35 (1) →
C	U	Z	GND	GND	GND	GND
			(SOURCE 2L5)	(SOURCE 2L2Ø)	(SOURCE 2E25)	(SOURCE 2J25)



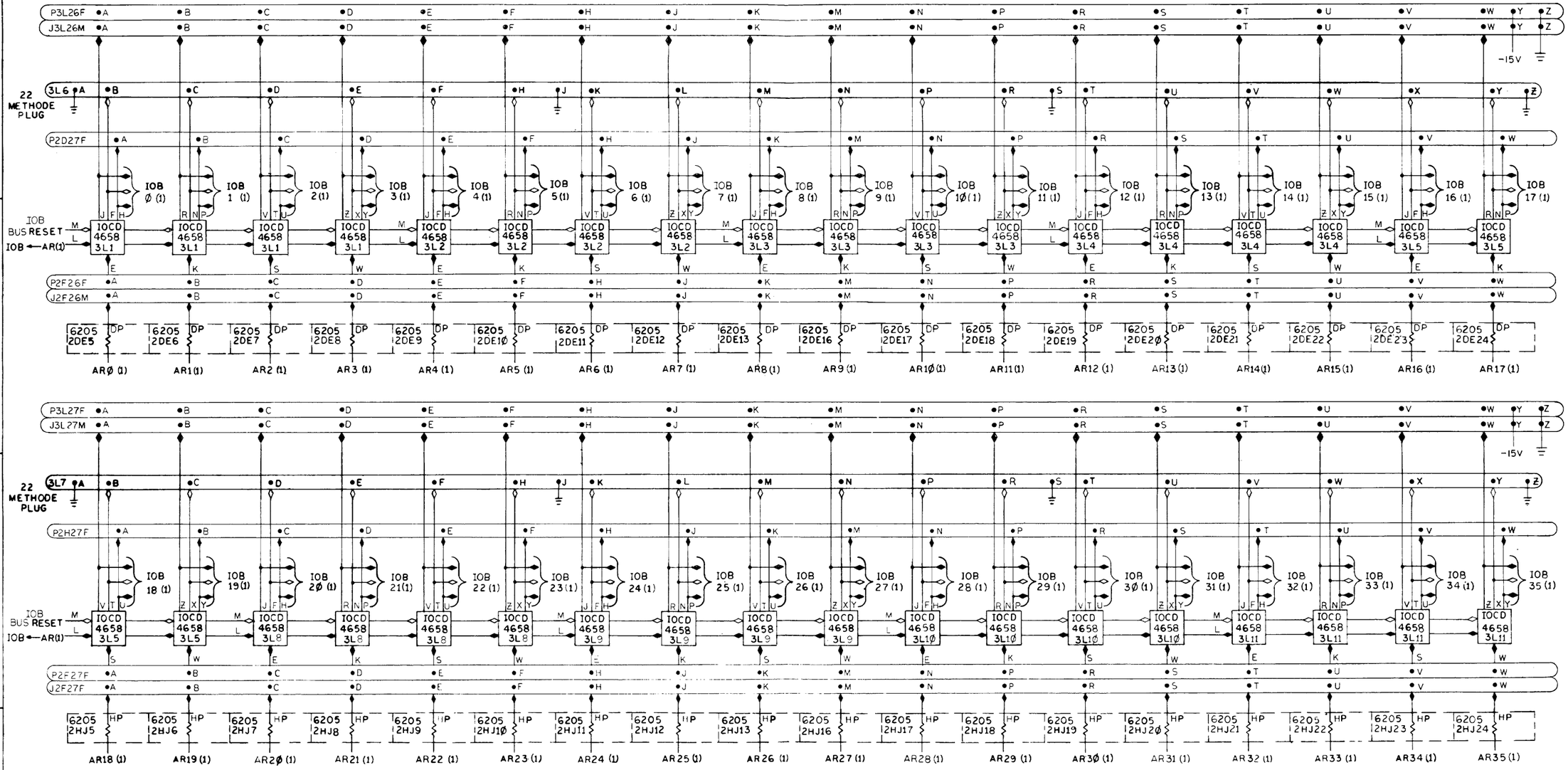
In-out and Memory Busses (CD-D-166-0-10MB)

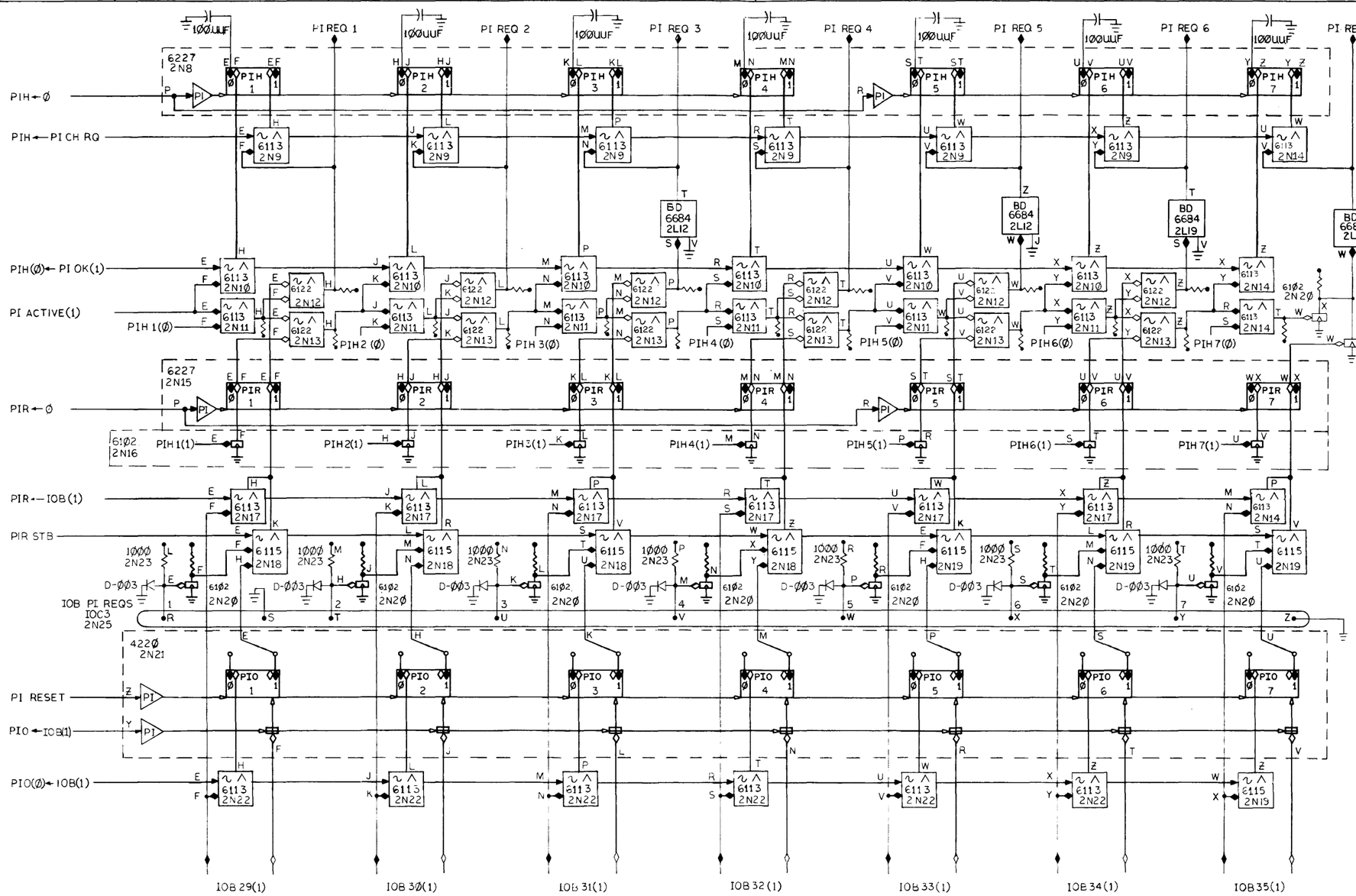
IOT, IOB Control (BS-D-166-0-IOTB)



IOT, IOB Control (BS-D-166-0-IOTB)

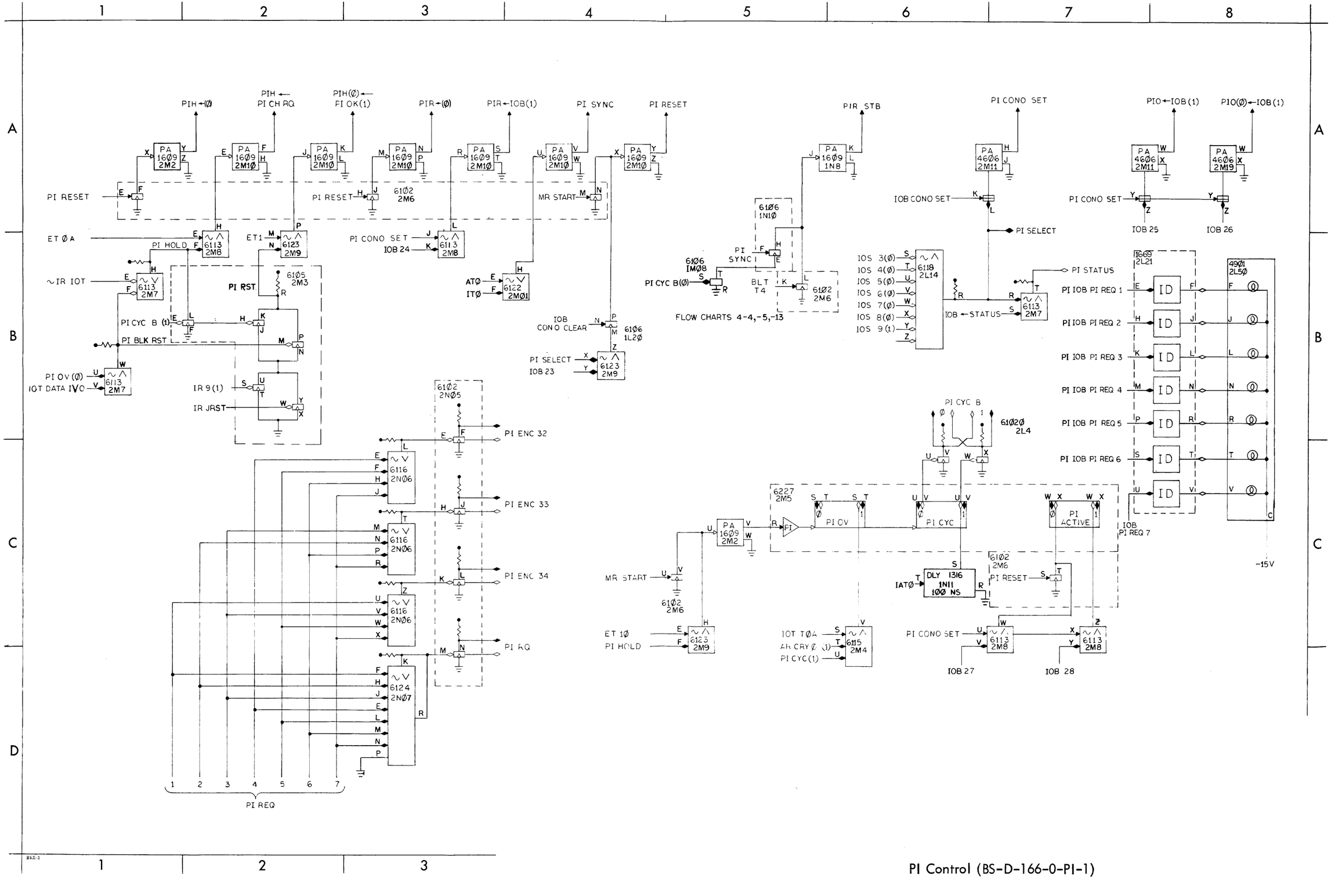
10B - 10 Bus (BS-D-166-0-10B)


P3L26F
J3L26M
3L6
22 METHODE PLUG
P2D27F
P2F26F
J2F26M
IOB BUS RESET
IOB ← AR(1)
IOB Ø (1)
IOB 1 (1)
IOB 2(1)
IOB 3(1)
IOB 4 (1)
IOB 5(1)
IOB 6 (1)
IOB 7 (1)
IOB 8 (1)
IOB 9 (1)
IOB 1Ø (1)
IOB 11 (1)
IOB 12 (1)
IOB 13 (1)
IOB 14 (1)
IOB 15 (1)
IOB 16 (1)
IOB 17 (1)
IOCD 4658 3L1
IOCD 4658 3L2
IOCD 4658 3L3
IOCD 4658 3L4
IOCD 4658 3L5
6205 2DE5 DP
6205 2DE6 DP
6205 2DE7 DP
6205 2DE8 DP
6205 2DE9 DP
6205 2DE1Ø DP
6205 2DE11 DP
6205 2DE12 DP
6205 2DE13 DP
6205 2DE16 DP
6205 2DE17 DP
6205 2DE18 DP
6205 2DE19 DP
6205 2DE2Ø DP
6205 2DE21 DP
6205 2DE22 DP
6205 2DE23 DP
6205 2DE24 DP
ARØ (1)
AR1(1)
AR2 (1)
AR3 (1)
AR4 (1)
AR5 (1)
AR6 (1)
AR7 (1)
AR8(1)
AR9 (1)
AR1Ø(1)
AR11(1)
AR12 (1)
AR13 (1)
AR14(1)
AR15(1)
AR16 (1)
AR17 (1)
P3L27F
J3L27M
3L7
22 METHODE PLUG
P2H27F
P2F27F
J2F27F
IOB BUS RESET
IOB ← AR(1)
IOB 18 (1)
IOB 19(1)
IOB 2Ø (1)
IOB 21(1)
IOB 22 (1)
IOB 23(1)
IOB 24 (1)
IOB 25 (1)
IOB 26 (1)
IOB 27 (1)
IOB 28 (1)
IOB 29 (1)
IOB 3Ø (1)
IOB 31 (1)
IOB 32 (1)
IOB 33 (1)
IOB 34 (1)
IOB 35 (1)
IOCD 4658 3L5
IOCD 4658 3L8
IOCD 4658 3L9
IOCD 4658 3L1Ø
IOCD 4658 3L11
6205 2HJ5 HP
6205 2HJ6 HP
6205 2HJ7 HP
6205 2HJ8 HP
6205 2HJ9 HP
6205 2HJ1Ø HP
6205 2HJ11 HP
6205 2HJ12 HP
6205 2HJ13 HP
6205 2HJ16 HP
6205 2HJ17 HP
6205 2HJ18 HP
6205 2HJ19 HP
6205 2HJ2Ø HP
6205 2HJ21 HP
6205 2HJ22 HP
6205 2HJ23 HP
6205 2HJ24 HP
AR18 (1)
AR19 (1)
AR2Ø (1)
AR21 (1)
AR22 (1)
AR23 (1)
AR24 (1)
AR25 (1)
AR26 (1)
AR27 (1)
AR28 (1)
AR29 (1)
AR3Ø (1)
AR31 (1)
AR32 (1)
AR33 (1)
AR34 (1)
AR35 (1)
-15V



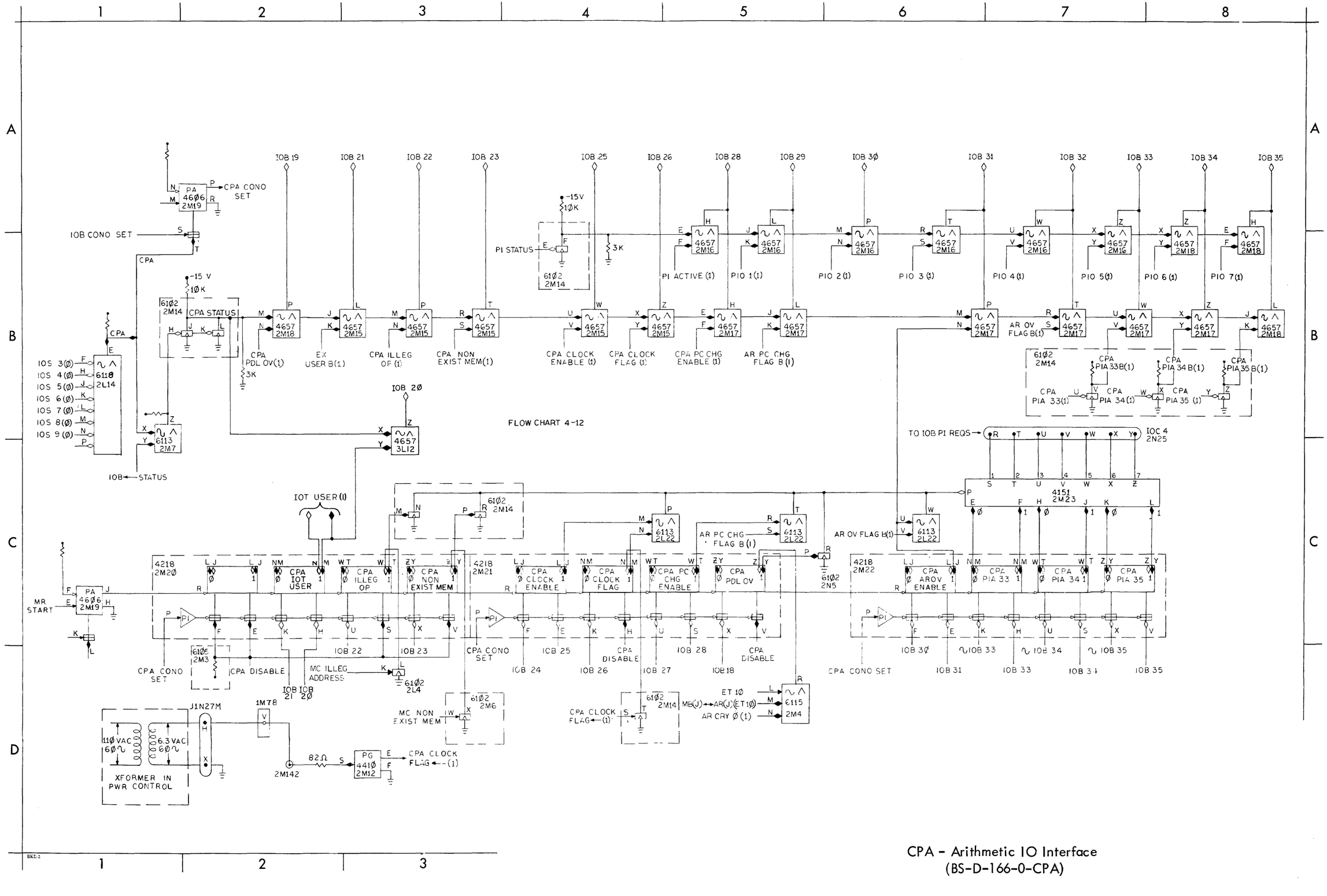
PI - PIH, R, O, 1-7 (BS-D-166-0-PI-2)

PI Control (BS-D-166-0-PI-1)



PI Control (BS-D-166-0-PI-1)

CPA - Arithmetic IO Interface
(BS-D-166-0-CPA)

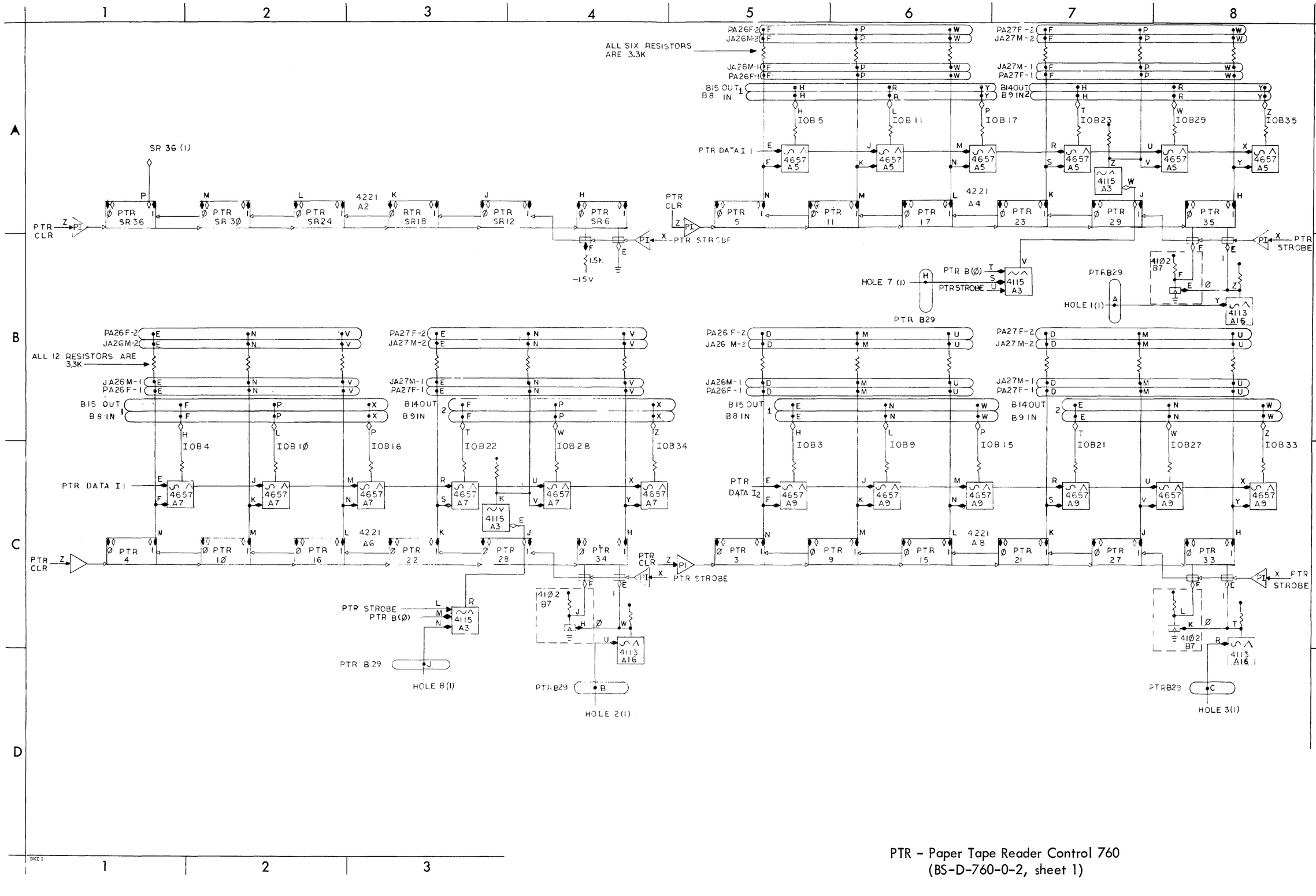


CPA - Arithmetic IO Interface
(BS-D-166-0-CPA)

PTR - Paper Tape Reader Control 760
(BS-D-760-0-2, sheet 3)

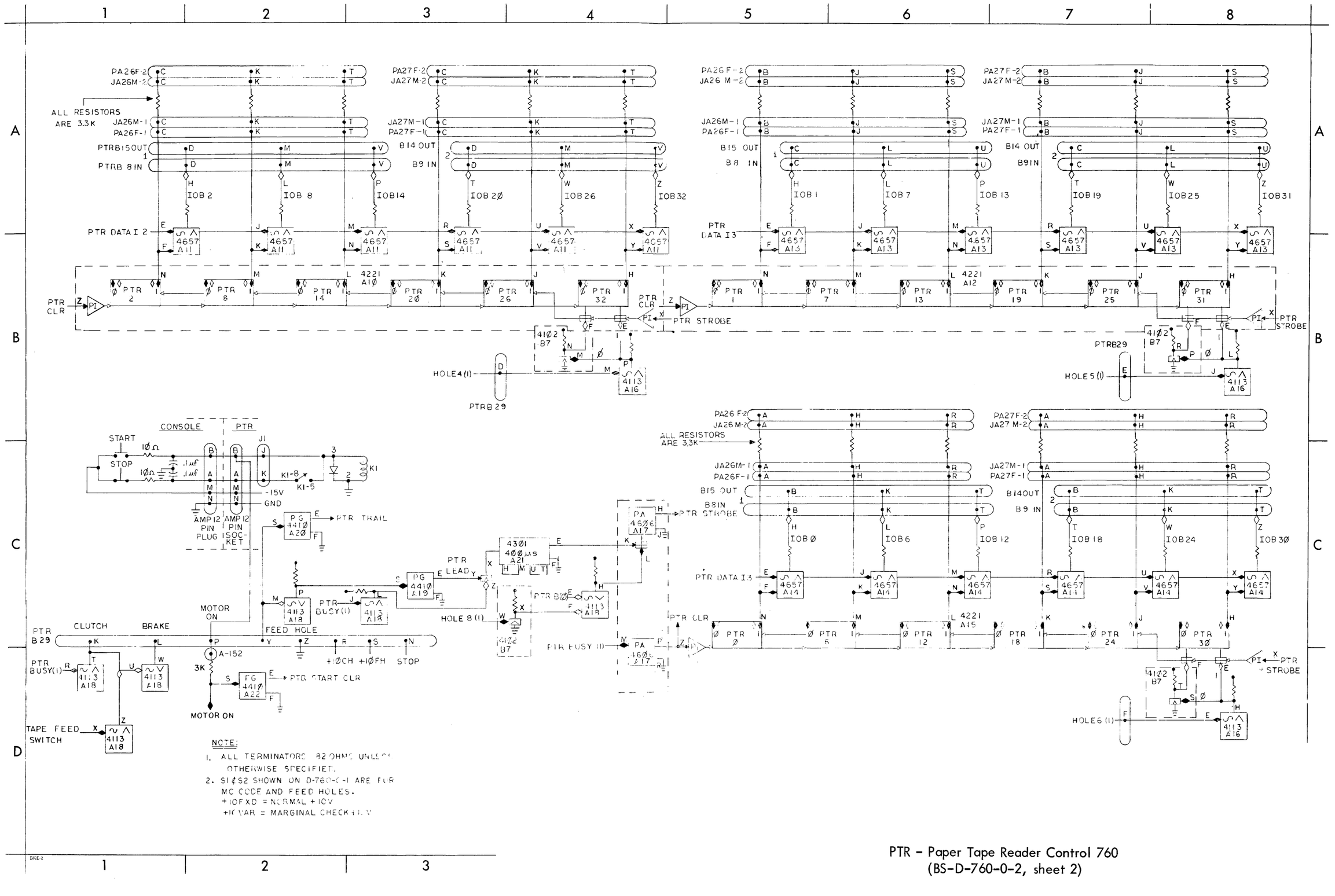
PTR – Paper Tape Reader Control 760
(BS-D-760-0-2, sheet 3)

PTR – Paper Tape Reader Control 760
(BS-D-760-0-2, sheet 1)



PTR - Paper Tape Reader Control 760
(BS-D-760-0-2, sheet 1)

PTR - Paper Tape Reader Control 760
(BS-D-760-0-2, sheet 2)

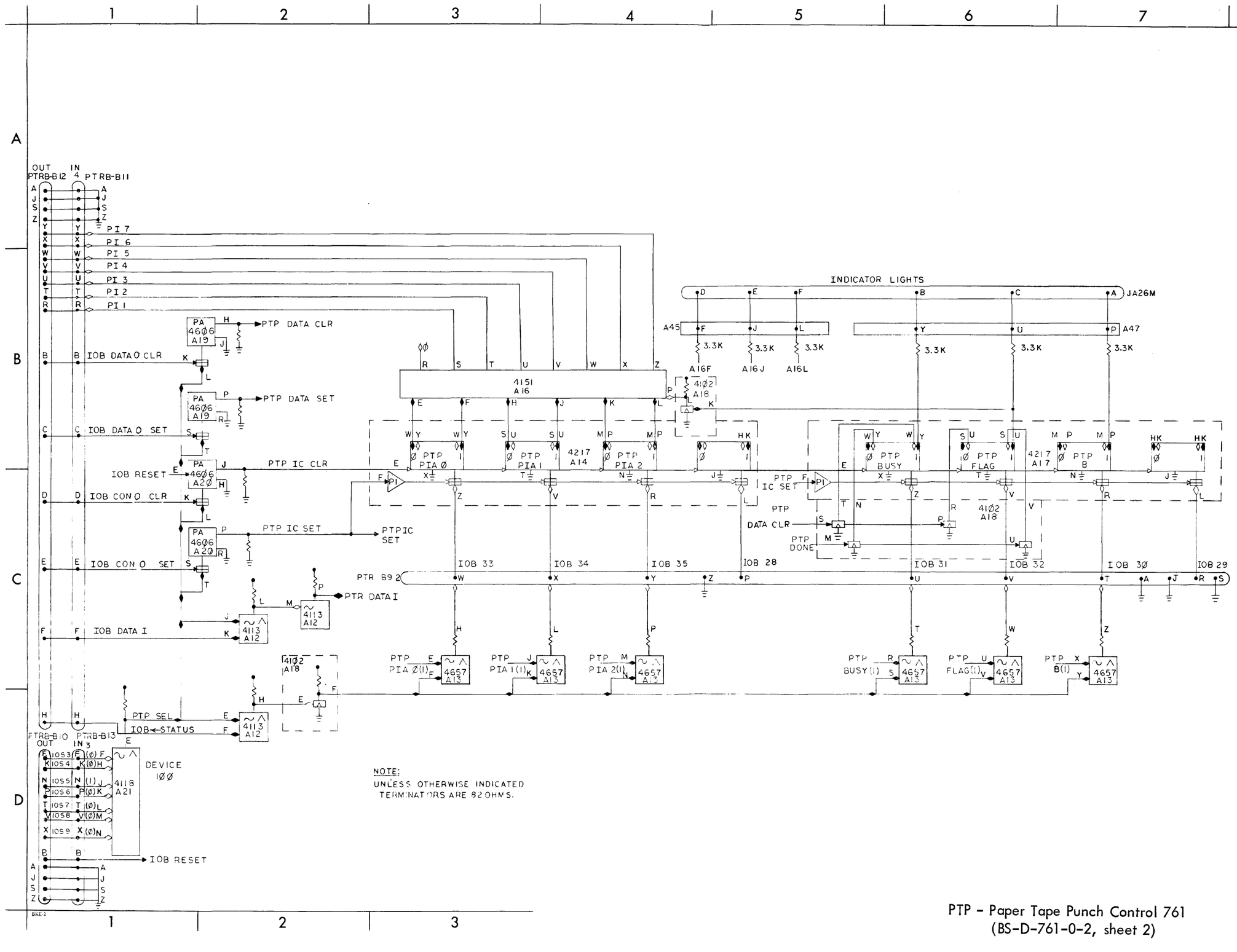


NOTE:

1. ALL TERMINATORS 82 OHMS UNLESS OTHERWISE SPECIFIED.
2. SIGS2 SHOWN ON D-760-0-1 ARE FOR MC CODE AND FEED HOLES.
   +IOFXD = NORMAL +10V
   +IOFXAR = MARGINAL CHECK +1.5V

PTR - Paper Tape Reader Control 760
(BS-D-760-0-2, sheet 2)

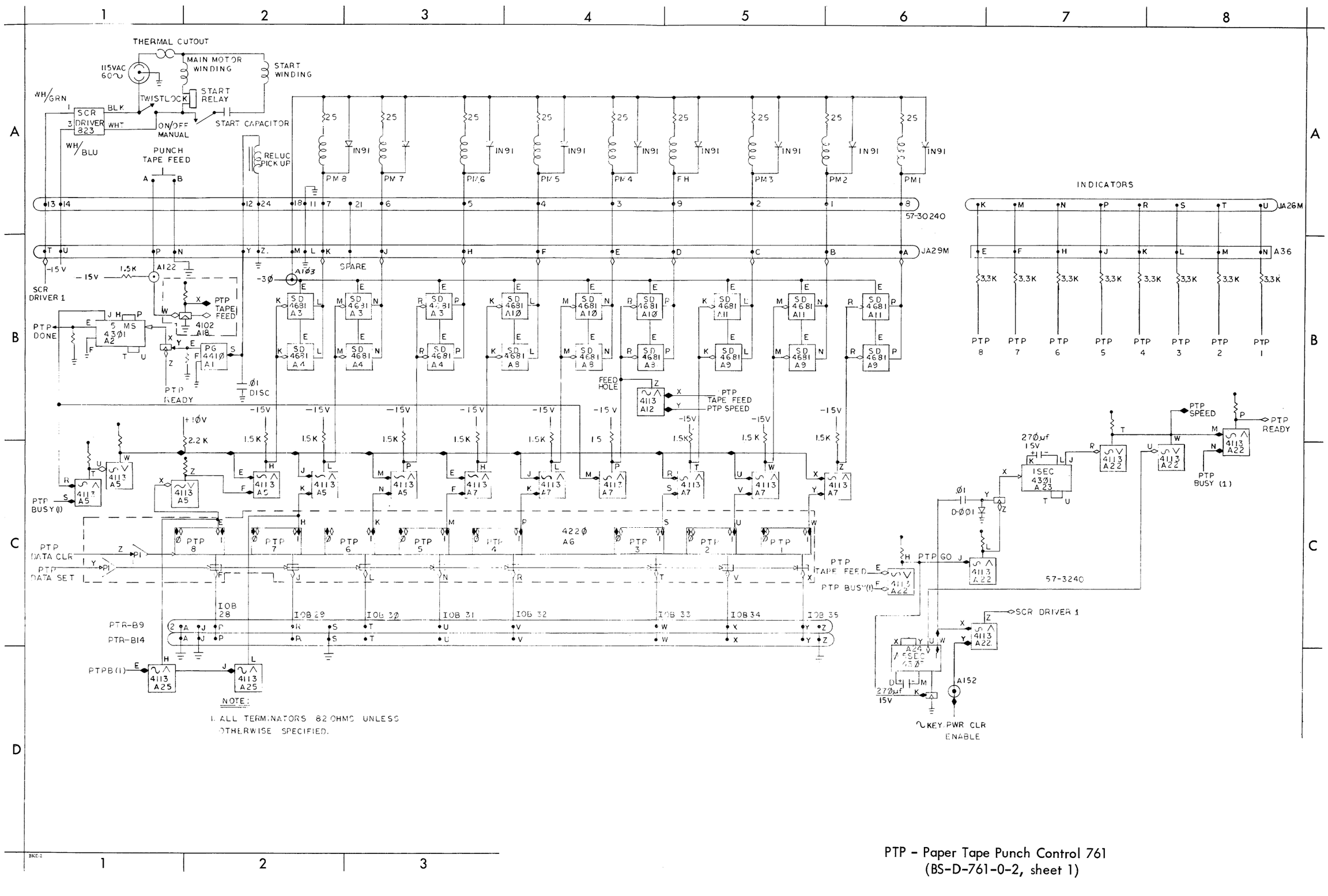
PTP - Paper Tape Punch Control 761
(BS-D-761-0-2, sheet 2)



NOTE:
UNLESS OTHERWISE INDICATED
TERMINATORS ARE 82 OHMS.

PTP - Paper Tape Punch Control 761
(BS-D-761-0-2, sheet 2)

PTP - Paper Tape Punch Control 761
(BS-D-761-0-2, sheet 1)



PTP - Paper Tape Punch Control 761
(BS-D-761-0-2, sheet 1)

TTI, TTO, TTY - Teletype Control 626
(BS-D-626-0-2, sheet 1)

TTI, TTO, TTY - Teletype Control 626
(BS-D-626-0-2, sheet 1)

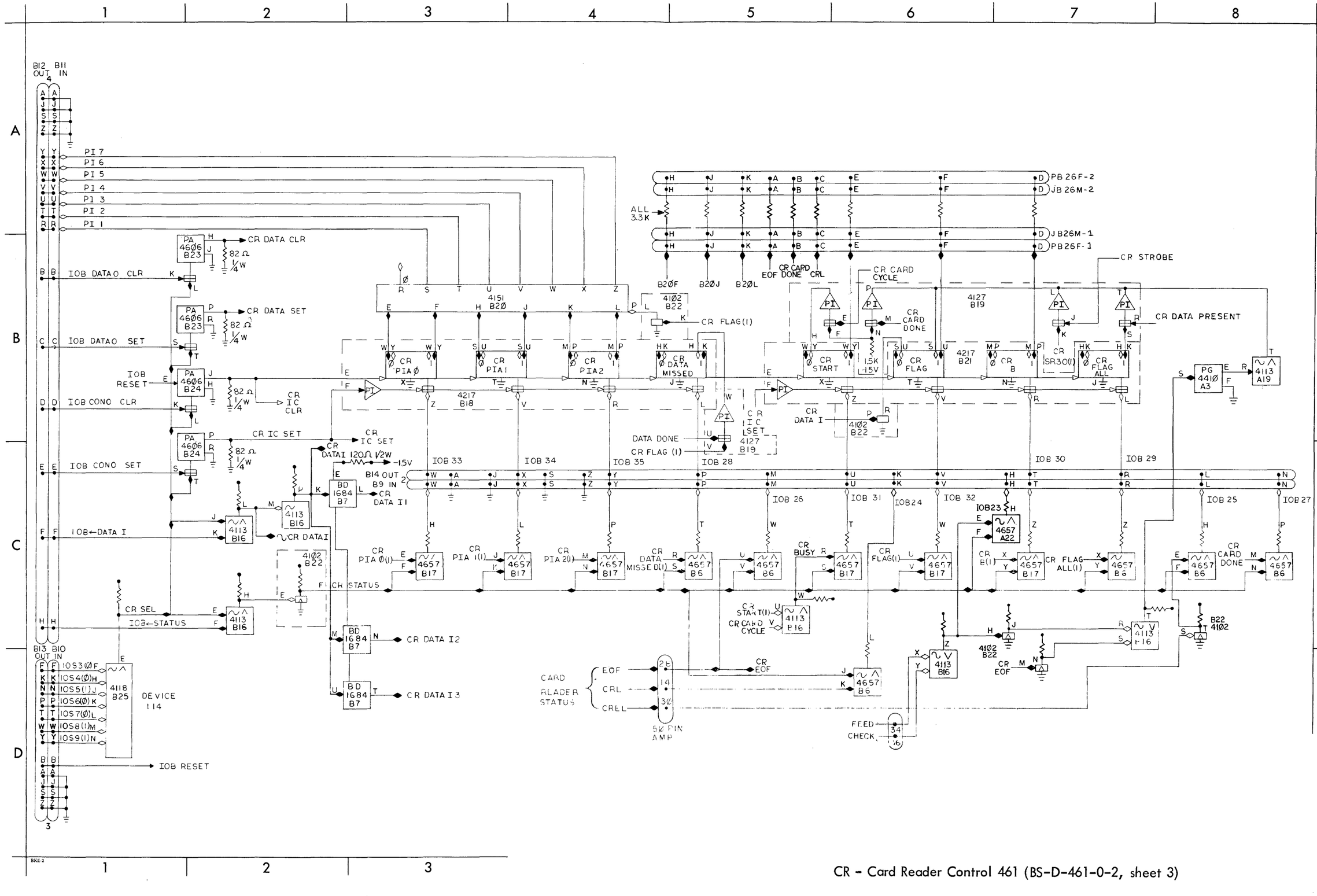
TTI, TTO, TTY - Teletype Control 626
(BS-D-626-0-2, sheet 2)

TTO TIMING

TTI TIMING

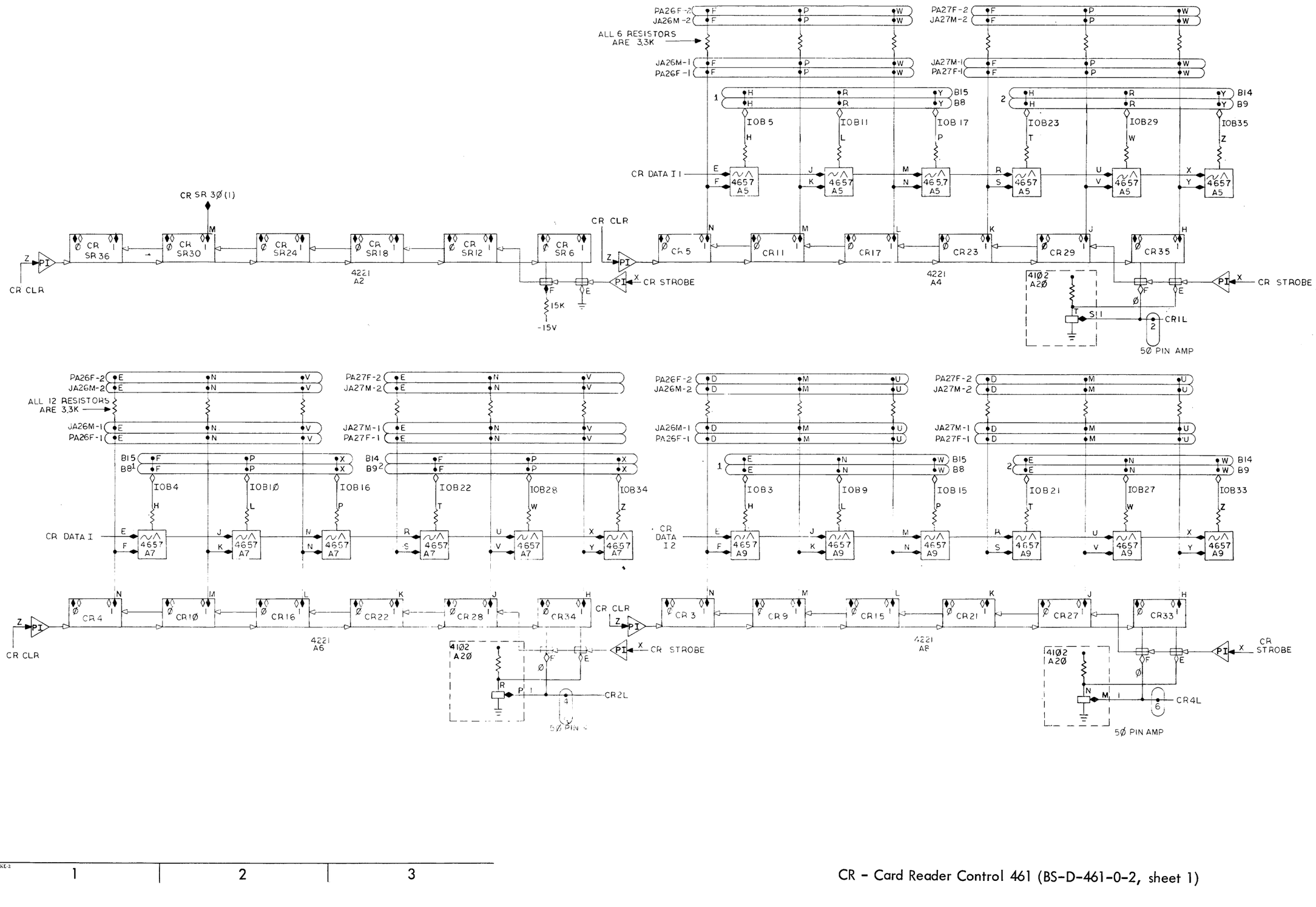
TTI, TTO, TTY - Teletype Control 626
(BS-D-626-0-2, sheet 2)

CR - Card Reader Control 461 (BS-D-461-0-2, sheet 3)



CR - Card Reader Control 461 (BS-D-461-0-2, sheet 3)

CR - Card Reader Control 461 (BS-D-461-0-2, sheet 1)



CR - Card Reader Control 461 (BS-D-461-0-2, sheet 1)

CR - Card Reader Control 461 (BS-D-461-0-2, sheet 2)

CR – Card Reader Control 461 (BS-D-461-0-2, sheet 2)

digital
EQUIPMENT
CORPORATION
MAYNARD, MASSACHUSETTS

Cambridge, Mass. • Washington, D. C. • Parsippany, N.J. • Rochester, N.Y. • Los Angeles
Palo Alto • Chicago • Ann Arbor • Pittsburgh
Denver • Huntsville • Orlando • Carleton Place
and Toronto, Ont. • Reading, England • Paris,
France • Munich, Germany • Sydney, Australia

F-67(CIR)

CIRCUIT INSTRUCTION MANUAL

PDP-6

F-67(CIR)

CIRCUIT
INSTRUCTION
MANUAL

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

COPY NO.

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

ILLUSTRATIONS

CHAPTER 1
INTRODUCTION AND DESCRIPTION

1.1 INTRODUCTION AND SCOPE

The purpose of this instruction manual is to aid personnel in maintenance, at the circuit level, of the DEC Programmed Data Processor-6. The manual contains complete descriptions of all circuits found in the Type 166 Arithmetic Processor, Type 162 Fast Memory, and in the control logic for four peripheral devices usually included in the installation: Type 760 Paper Tape Reader, Type 761 Paper Tape Punch, Type 626 Teletype Printer-Keyboard, and Type 461 Card Reader. Also included are all standard logic circuits used in the interfaces and control portions of Core Memories Types 161C and 163C; all special circuits associated with a core stack, such as the regulated power supply, core drivers, switching circuits, and sense amplifier, are described in the maintenance manual for the memory in which they are used. Troubleshooting information is included as it pertains to module replacement and repair; detailed information on system operation and troubleshooting may be found in the maintenance manuals for the PDP-6 arithmetic processor and memory. Descriptions of circuits unique to any equipment not listed above are included in the appropriate maintenance manuals.

Where possible, material common to circuits in a given class, such as basic inverter or flip-flop operation, supply voltages, and common jumper codes, is presented in the introductory remarks for each chapter. Timing characteristics common to each of the three DEC module series (1000, 4000, 6000) are also included here. The remainder of each chapter consists of material applicable to individual circuit types, such as circuit logic (if required), connector information, unique jumper codes, and all special circuitry.

Schematic diagrams for all circuits, with modified logic diagrams of a few complex modules, are grouped at the rear of the manual in order by type number. No figure references are made in individual unit descriptions, but references to applicable schematics are implied.

1.2 PDP-6 MODULES

The DEC Programmed Data Processor-6 is constructed of digital modules and power circuits. The module components are mounted on a dip-soldered epoxy board surrounded by an aluminum frame measuring (with some exceptions) 1/2 x 4-1/2 x 7 inches. Logic and power connections are made through 22-pin Amphenol plugs, connected to the circuits by flexible wiring to minimize strain.

DEC circuits can be categorized by function and by maximum operating frequency. PDP-6 circuits fall into seven functional groups:

Inverters
Diode gates and decoders
Flip-flops
Amplifiers
Delays
Pulse circuits
Power circuits

Most types are discussed in some detail; however, the first two groups (simple inverter and diode networks) are described collectively, since the reader is assumed familiar with basic solid state electronics.

A DEC module (any circuit in the first six classes) is assigned one of three maximum operating frequencies, 500 kc, 5 mc, or 10 mc---the 4000, 1000, and 6000 series respectively (although some 4000 series modules can operate at 1 mc). Module timing characteristics are commensurate with these speed categories. DEC standard pulse widths are 0.4 μsec for the 4000 series, 70 nsec for the 1000 series, and 40 nsec for the 6000 series. These pulses are 2.5v in amplitude. Tolerances are +2.3 to +3.0v and −2.3 to −3.5v. DEC signal levels are standard for all modules at ground and −3v. Tolerances are 0 to −0.3v and −2.5 to −3.5v.

To facilitate system wiring, four connector pins on each module are reserved for supply voltages: two pins, A and B, carry +10 vdc (for ease in submodular marginal checking), pin C supplies −15 vdc and pin D is grounded. Additional pins may be used for ground as necessary. All modules that require it have a built-in source of −3 vdc, which utilizes the 0.75v forward drop of the silicon diode. Four series diodes are used, biased to −15 vdc, usually through 560 ohms (figure 1-1). The resistor accepts enough current to regulate the supply under all load conditions.

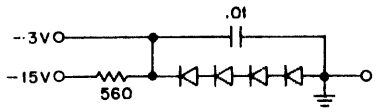


Figure 1-1 Standard −3 vdc Supply

Five PDP-6 modules, Flip-Flops 1250 and 6205, and Amplifiers 1664, 1665, and 6615, require special consideration. These types have three times the area of the standard DEC module and are serviced by two DEC mounting panels at each end--a packaging technique that permits a considerable reduction in space, in module types, and in external connections when the number of repetitive, interdependent circuits is large. Each pin designation on these five schematics is therefore two letters, the first indicating the connector (A, upper front; B, lower front; R, upper rear; S, lower rear), the second referencing a single pin of the selected connector. (On logic diagrams of the PDP-6 computer, letter designations of the

appropriate mounting panels replace the front connector prefixes; rear prefixes remain the same.) Two other modules, Flip-Flops 6203 and 6206, are double-length, standard-height types and are serviced by single connectors at both ends. Rear connectors have the usual R prefix.

A number of modules have provision for internal jumpering, which permits their use in a variety of applications without requiring additional external connections. Positions at which jumpers can be used are indicated on the schematics by dotted lines between unlettered terminals. Spare modules are delivered with all jumpers in place, and the technician must cut out the ones not wanted. On the UMLs, the type numbers for jumperable modules are followed by number or letter combinations that indicate jumper configurations (all such connections are shown on the logic drawings). Decoding methods for these symbols are given with the module descriptions.

On some circuit schematics, certain groups of passive components have a type number but no value designations; these are encapsulated elements whose values are called out elsewhere on the schematic. They appear on the modules as rectangular orange capsules.

PDP-6 modules containing only clamped loads, terminating resistors, or indicator lamps are considered self-explanatory and are not discussed here, although their schematics are included at the rear of the manual.

Replacement schematics for the circuits have drawing numbers of the form RS-X-N, where X is a letter indicating drawing size and N is the circuit type number, usually three or four digits. To the right of the drawing number is another, the revision number of the circuit schematic. To the left is a letter that indicates the revision of the circuit board. The revision letter is etched on the board and printed after the type number on the aluminum frame of the module.

1.3 PERTINENT DOCUMENTS

The following documents augment these discussions presented in this book.

a. System Modules Catalog, C-100

b. PDP-6 Arithmetic Processor 7F-67 (166)

c. PDP-6 Fast Memory Type 1621 Core Memory Type 161C, F-67 (162-161C)

d. PDP-6 Fast Memory Type 1621 Core Memory Type 163C, F-67 (162-163C)

Within this text constant reference is made to standard modules and configurations. All module information such as detailed circuit schematics can be found in the module catalog. References to PDP-6 applications of decoders, etc., can be further studied in the PDP-6 Arithmetic Processor 166 Manual.

CHAPTER 2
INVERTERS AND GATES

Most logical operations in the PDP-6 are performed with saturating inverters and diode gates. Because of their relatively simple design and because of the consistent use of inverters as gate buffer amplifiers, both groups are described in this chapter. For information on number of circuits per module, pin connections, jumpers, etc, refer to the DEC System Module Catalog, C-100. Many of these modules have clamped loads that do not appear at the output connectors but may be jumpered internally to inverter collectors. The presence of a jumper is indicated by a 1 in a jumper code of n bits (octal), where n is the number of clamped loads in the module. From left to right the bits correspond to the alphabetical order of the collector pins at the connector. An R is used in place of a number to indicate that all clamped loads are connected.

2.1 INVERTERS

An inverter can be used as a level gate or a pulse gate. Inverter transistors are pnp type, and operate either in saturation or in cutoff. The collector-emmitter impedance of a saturated transistor is very low; at cutoff, it is very high. If the emitter is grounded and the collector is connected to a negative load, the collector output is inverted with respect to the base input. Base input loading is supplied by the 3K resistor (figure 2-1); a saturating base current of 1 ma therefore flows through the transistor when −3 vdc is present at the input. A bypass capacitor at the input provides overdriving current which speeds switching. The more positive input to an inverter usually is derived from the collector of a saturated transistor (approximately −0.2v). Base bias at 0 vdc (input unterminated) is supplied by a divider between +10 and −15 vdc. This allows marginal checking inverter base leakage (simultaneously checking collector saturation voltage of the stage preceding the inverter) and inverter dc gain.

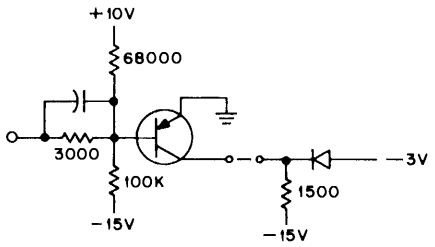


Figure 2-1 Standard Inverter

The diode in the clamped load limits the negative voltage at the inverter output, supplying the current needed to maintain the output at −3v. With a 1.5K load resistor, diode current is a maximum of 8 ma under no-load conditions and decreases to 0 as the current drawn from the external load increases to 8 ma.

Delay through a 4000-series inverter is 300 nsec (defined as the time between the point of 10% input change and 10% output change); the 1000-series delay is 20 nsec, whereas that of a 6000-series inverter is approximately 12 nsec.

2.2 GATES

The PDP-6 diode modules are of three basic types: simple networks for general purpose pulse and level gating; capacitor-diode gates; and decoders.

2.2.1 Diodes

The diode gate modules contain two or more diode logic gates, consisting of a number of diode-coupled inputs driving an inverter base. A clamped load is available for each inverter. The gating diodes provide either the AND or the OR function for a given assertion level; the transistor amplifies and inverts the diode output so that the diode-inverter combination is properly termed a NAND or NOR gate. In the negative NAND gate shown in figure 2-2, Q1 is driven into conduction only if A, B, and C are all negative (−3 vdc), allowing R1 to forward-bias the transistor emitter-base junction. With ground at any input, R2 reverse-biases the junction and cuts off the transistor.

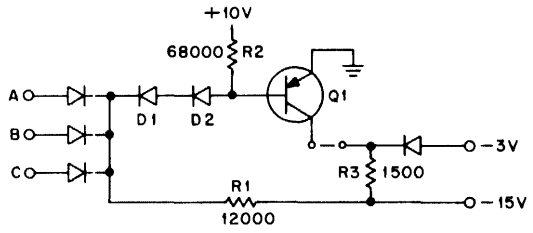


Figure 2-2 Negative Diode NAND Gate

If the source of a nominal ground input is the collector of a saturated transistor, the actual input voltage is slightly negative. The drop across the gate diodes further reduces the voltage applied to the inverter. To compensate for these reductions in input ground levels, two series silicon diodes, D1 and D2, are employed as level shifters between the gate proper and the inverter base. The shift is large enough to ensure positive bias at the base and hence reliable cutoff when any normal ground input is applied to the gate.

The negative diode NOR gate shown in figure 2-3 drives Q2 into saturation through input resistor R4 when −3 vdc is applied to D, E, or F. When none of the inputs are negative, R5 cuts off Q2 by reverse-biasing the emitter-base junction. A diode, D3, is always placed across this junction to prevent the transistor base from being driven too positive when no input connections are made. R5 may actually be two resistors connected to both +10v supplies, as is done in module 6116. This prevents undue sensitivity of the circuit to marginal voltages.

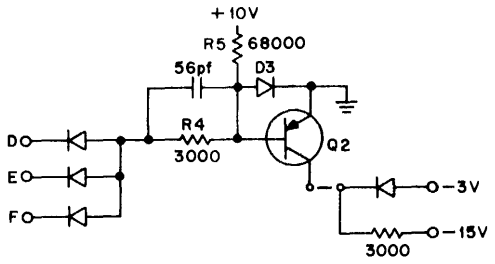


Figure 2-3 Negative Diode NOR Gate

2.2.2 Capacitor-Diode Gates

In the basic positive capacitor-diode gate shown in figure 2-4, a positive pulse or level change at A produces a positive pulse at C, provided a ground enabling level has been present at B for a certain interval before the pulse. This interval depends on the resistor-capacitor time constant, and is the time required for the common junction voltage to rise sufficiently close to ground. The gate is inhibited when the junction voltage falls close enough to −3 vdc that the input pulse cannot forward bias the diode. A clamped load often connects to the gating level input. In the basic negative gate the diode polarity is reversed, and the pulse input, enabling input, and output are negative. A complete capacitor-diode circuit provides pulse amplification in an inverter, which is associated with a single gate or shared among several. In all positive gate modules the inverter is at the pulse input; in all negative modules, at the pulse output. Thus, regardless of the polarity of the gating level, all modules accept a negative pulse or level change and produce a positive-going output.

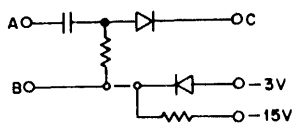


Figure 2-4 Basic Positive Capacitor-Diode Gate

The circuit at the left in the 4127 schematic is typical of negative capacitor-diode gates. A −3 vdc level applied to F enables the gate; a negative pulse or level change at E then generates a positive-going output pulse at H. The delay inherent in this circuit requires that the gating level be present at least 1 μsec before arrival of the pulse. When F becomes negative, the C1-D1 junction falls from ground to −3v, the fall time being determined by the time constant of C1-C2. After this interval, a negative pulse applied to E is sufficient to forward-bias D1 and pass through the diode to the load. The gate output is referenced by R5 to a dc level of −3v. D2 and R1 prevent the gate input from going more negative than −3v, preventing undue sensitivity of the circuit to noise at E. When D1 is forward-biased by a pulse at E, the negative pulse at the D1-R5 junction is coupled through C2 to the Q1 base. As the transistor turns on, output H rises to ground. At the trailing edge of the input pulse, D1 is cut off. The Q1 base is normally biased positive by R3-D3 to reduce noise susceptibility.

The circuit in the upper left of the 4128 schematic is typical of positive capacitor-diode gates. The pulse input P can be driven by a standard negative pulse or by a negative-going level change with a maximum fall time of 5 μsec. Each gate is enabled by a ground level that must be present at least 1.5 μsec before arrival of the pulse; however, for a transfer of 1s or 0s only, all off gates must be disabled for 4.5 μsec before readin. If the D2 cathode is tied to the base of a saturated transistor, it is near ground potential. A −3v to ground pulse or level change at the Q1 collector is coupled through C4 but can forward-bias D2 only in the presence of a ground level at F.

2.2.3 Decoders

The decoder modules contain special diode configurations that decode a given binary information group to an assertion at one of several possible outputs. The Type 1151 contains an 8-section diode matrix that is driven by the 1 and 0 outputs of three flip-flops. Each section is a 3-diode negative AND gate, which receives a unique combination of inputs representing one of the eight states of the three flip-flops. The assertion of a specific binary input configuration thus turns on a single output inverter. For example, when all flip-flops are 0, the output of the gate at the far left of the schematic, whose inputs are connected to the flip-flop 0 outputs, is at −3 vdc. The outputs of all other diode gates are at ground. At any time, therefore, all inverter outputs are negative except the one driven by the satisfied diode gate. The entire module can be gated on or off at the common emitter connection of the inverters, or the emitters can be jumpered to ground internally or grounded at the connector.

Types 4151 and 6151 are identical to the 1151 in all respects except speed. The 6150 is similar to the 6151 except for inverted logical sense (all inputs to a gate must be at ground to obtain a unique negative decoder output), internally grounded emitters, and a common enable input to the diode matrix.

The Type 6155 contains two binary-to-quarternary networks. Each output inverter is driven by a 3-input diode AND gate, which is in turn driven by the outputs of two flip-flops and a common enabling level. When all three inputs to a given gate are negative, the ground output of its associated inverter is uniquely asserted.

2.3 MAJORITY LOGIC

In the memory protection and relocation networks are two circuits that employ majority logic. Type 6131 is a dc adder which uses majority logic to develop the carry output; Type 6132 contains two majority gates which form part of the borrow chain in an arithmetic comparison.

In addition there must be a carry to the next more significant bit if two or more of the three inputs (two summands and a carry) are 1s. For subtraction, the minuend bit is complemented, and the carry output then represents a borrow to the next bit. If inverter or diode gates were used to develop the carry output, separate AND gates to decode all four possible input cases that necessitate a carry would be necessary, as well as an OR gate to form the output. Majority logic does not differentiate as to the source of the logic inputs; instead, all three are fed to precision resistors and the voltage at the junction point is compared to a fixed threshold by a difference amplifier. The summing resistors are paralleled by speed-up capacitors and the two-stage difference amplifier used in both modules operates far from saturation. Consequently the propagation delay for the carry is less than 10 nsec per bit.

2.3.1 6132 Majority Gate

This module consists of two gate circuits and appropriate internal supply voltages. The output of a gate is asserted whenever at least two of its three inputs are asserted. In the protection logic the A input (pins X and E respectively of the left and right gates) is an MA bit on 1, the B input is a PR bit on 0. These inputs are labeled $\overline{A}$ IN and $\overline{B}$ IN because they are made through inverters and hence require opposite assertion to C IN, which is the carry from the next less significant stage. The carry out is available at both polarities. Since these circuits are used in series, the carry output of the left gate is wired internally as the carry input to the other but is available at the connector via an internal jumper. If the A and B inputs to a series of gates represent 1s in a pair of numbers, C is the carry function. If one set of inputs, e.g., B, represents 0s, C is the borrow function of B-A.

The A and B inputs have ground assertion and drive standard DEC 6000-series inverters with clamped loads (for example Q1-Q4 in the left gate). The three majority logic resistors (R3-R6-R9) are tied together at the left base of Q2, the first stage of the difference amplifier. The carry input is asserted negative at R3 to correspond logically with the inverted signals at R6 and R9. The two emitters of Q2 are tied together to R8, a current sink to −15 vdc. When two or more of the inputs to the resistance summing

network are −3v, the sum point (left base of Q2) falls below −1.5v, cutting off the left side of Q2 and allowing the right side to conduct all current. Conversely when two or more inputs are at ground, the summing point rises above −1.5v turning on the left side and cutting off the right. Neither Q2 collector can saturate however because current sink R8 conducts a maximum of 1.5 ma which cannot bring either collector below ground.

The difference amplifier output stage is Q3 whose bases are driven directly by the Q2 collectors. The emitter current source for Q3 is R14 to +10 vdc; the collectors are clamped to 0 and −3 vdc by D2 and D3 on the left, D4 and D6 on the right. Coils L1 and L2 provide shunt inductive frequency compensation for the Q2 collectors to enhance propagation speed. C4 and C8 as well as the various capacitors associated with the series voltage supply at the right serve to keep voltage supply points free from transients.

2.3.2 6131 DC Adder

This module contains a majority gate, an exclusive OR network, and supplies for the necessary internal voltages. If bits from two summands are asserted ground at pins E and F, and a carry in is asserted ground at L, negative at T, a bit of the sum appears at outputs X and Y (ground and negative, respectively) and the carry out to the next more significant stage appears in both polarities at U and K. The negative sum out is also available at pin W isolated through a resistor to drive an indicator. A fifth input, pin H, receives a ground enable level. When the enable falls to −3 vdc, the F input is negated internally and consequently the sum output is identical to the E input (assuming there is no carry in, which would ordinarily be the case). In the computer, these circuits relocate memory addresses for the user mode. For this application input E receives an MA bit, F an RLR bit, the carry inputs come from the next less significant adder, and H receives the relocate enable level from executive mode control.

In the schematic the inverters for inputs E, F, and H are at the top left. Below these are two exclusive OR stages and an output current switch (difference amplifier). To the right of center is a difference amplifier for the majority gate and at the far right is the series-diode voltage source. Except for the input inverters and common internal supply, the sum and carry networks are completely independent. Inputs E and F drive Q2 and Q6, the latter paralleled by Q8 so that a negative level at H prevents negative assertion of the F input at the Q6 collector. Both inverter loads are clamped at −3 vdc by D1 and D2. L is the carry input of the majority gate, whose input for the adder is T.

The adder uses two exclusive OR stages to produce a sum in the same way as do the arithmetic register bits except that the output depends only on the propagation of level changes rather than pulsed flip-flop changes. The first stage produces a partial sum from the two summand inputs; the second stage then inverts this partial sum whenever there is a carry. In other words when neither or both summands are asserted, the sum output is equal to the carry input; otherwise the sum is the inversion of the carry. The first stage consists of transistors Q3 and Q5 with associated RC base feeds. The Q3 and Q5 collectors are

tied together so that the output at the D3 clamp can be −3v only when both are cut off. In this exclusive OR pair, the base of one transistor is driven through an RC network from the emitter of the other. Consequently when both emitters are at the same voltage both transistors are cut off; only when the two emitter drive signals differ can one of the pair saturate. The collectors are at −3v whenever the E and F inputs are identical, at ground when they differ.

The second exclusive OR stage is Q1 and Q7 with similar cross connection from emitters to bases. The first stage drives the Q1 emitter through an RC network; the emitter of Q7 receives the carry, asserted ground at T, through a similar RC net. When E and F are identical, the Q7 base is −3v so Q7 conducts whenever there is a carry. When E and F differ, the Q7 base is at ground and Q1 determines the sum as the complement of the carry. In order to prevent saturation in this stage the Q1-Q7 collector junction is not clamped at −3 vdc; only the input inverters and the first exclusive OR stage can saturate. Since emitter current sources R2 and R16 cannot allow current sufficient to bring the second-stage collectors more positive than −3v, saturation does not occur. The Q1-Q7 collector junction drives the left base of Q4, but the right base is held at −4.5v from the series-diode voltage source at the far right. When either Q1 or Q7 conducts, the left half of Q4 supplies all the current through R14 to −15v, pulling the sum 1 output, pin Y, down to the −3v clamp level. The right collector of Q4 floats, clamped at ground by D7, thus negating the sum 0 output at pin X. When Q1 and Q7 are both cut off, the right half of Q4 brings the X output negative, whereas the floating left collector allows Y to return to the ground clamp level.

Input inverters Q2 and Q6 also drive R11 and R10, two of the three majority logic input resistors. The third resistor R17 is driven directly by the negative carry input at pin L. The junction point of the three resistors drives the left base of Q9, the first of two difference amplifier stages that develop the carry output. The Q9 emitters are paralleled; current sink R23 accepts enough current to drive Q10 but not enough to saturate Q9. The right base of Q9 is held at −1.5v from the diode voltage supply. The difference amplifier formed by Q9 and Q10 is identical to that in the majority gate discussed in the preceding section. It develops the carry output at pin K (at the left Q10 collector), clamped at 0 and −3 vdc by D10 and D9. The complement is available at pin U from the right collector, clamped by D11 and D12.

CHAPTER 3
FLIP-FLOPS

Of the flip-flop types used in PDP-6, the 1260, 4706, 4707, 6203, 6205, and 6206 perform rather complex logical operations, so modified logic diagrams for these modules have been included with the schematics to make circuit identification and component location easier. In many flip-flop modules, jumpers select the outputs that appear at the connector or select the flip-flop states produced by a preset pulse. In either case the side of a flip-flop to which a jumper is connected is indicated by a bit in a jumper code of n bits (octal), where n is the number of jumperable flip-flops in the module. The bits left to right correspond to the alphabetical order of the flip-flop outputs at the connector. Other codes unique to particular modules are given in the module descriptions.

The flip-flop, or bistable multivibrator, is a two-state device used primarily for short term storage. A simple flip-flop can be constructed by interconnecting two grounded-emitter inverters (figure 3-1). When one inverter is cut off, its output is negative; this holds the other transistor on, which in turn holds the first transistor off. The state of the flip-flop can be changed by driving either the base of the conducting transistor or the collector of the nonconducting one with a positive-going pulse (although the latter method requires a more powerful driving source). It is this ability of the flip-flop to remain in the state to which it is driven, even after removal of the activating signal, which permits its use as a storage device.

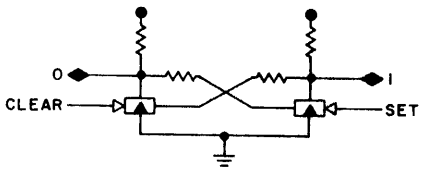


Figure 3-1 Basic Flip-Flop

It is often desirable to reduce flip-flop output impedance in order to drive heavier loads. For this reason, many flip-flop modules are equipped with buffer inverter or emitter follower amplifiers at the outputs. Delay networks are used in PDP-6 flip-flops when speed considerations prevent the use of capacitor-diode gates and intrinsic circuit delay is not sufficient to allow output terminals to be sensed while input terminals are being pulsed.

3.1 OPERATION OF COMPONENT CIRCUITS

To avoid repetition, six circuits basic to the operation of many PDP-6 flip-flops are discussed in this section. Although component values may differ in some instances when the same circuit is represented in several speed lines, the descriptions of these circuits are not dependent on operating frequency.

3.1.1 Unbuffered Flip-Flop

Figure 3-2 shows this circuit, with a type of input gating commonly used with 4000-series flip-flops, the positive capacitor-diode gate. The flip-flop is stable in either of two states: Q1 conducting and Q2 cut off, or vice versa. Its state can be changed by applying a positive-going pulse to the base of the conducting transistor or to the collector of the nonconducting transistor.

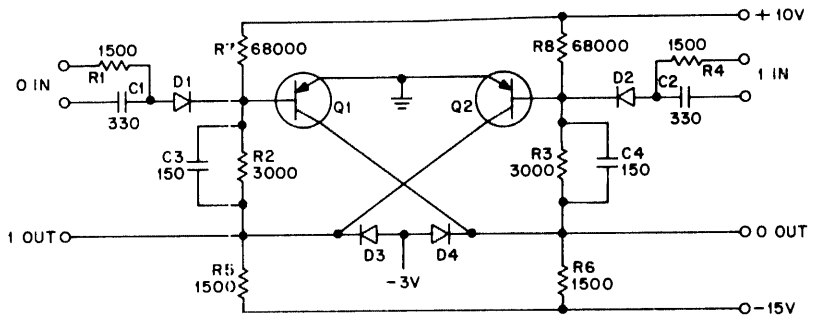


Figure 3-2 Unbuffered Flip-Flop with Positive Capacitor-Diode Input

Since the collectors of Q1 and Q2 are resistively cross-connected to the opposite bases, R2 and R3 hold the flip-flop in its new state after the disappearance of the trigger pulse. Base bypass capacitors C3 and C4 provide overdriving current to the transistors to speed switching. The flip-flop is said to be in the 0 state when the 0 output terminal is quiescent at −3 vdc and the 1 output is grounded. Q2 remains saturated and Q1 cut off until the flip-flop is set through capacitor-diode gate D2-C2-R4. This gate is enabled by a ground level at the R4 input, raising the D2-C2 junction to ground. A standard positive voltage step applied to the C2 input then forward-biases D2 (if a ground enable is present at R4) and turns off Q2. The sharp negative spike caused by the differentiation of the pulse trailing edge is discharged through R4.

As the positive-going trigger reverse-biases Q2, the resulting negative transition at its collector is coupled through the parallel combination of R2 and C3 to the Q1 base. As this point falls below the Q1 emitter, Q1 conducts and its collector rises to ground. This level is, in turn, coupled through C4 and R3 back to the Q2 base, allowing R8 to maintain Q2 at cutoff. The flip-flop is now in the 1 state. D3-R5 and D4-R6 are output clamped loads. The flip-flop is cleared back to 0 by enabling the C1-R1-D1 gate and applying a positive pulse to the Q1 base. Unbuffered flip-flop delay is approximately 50 nsec, 25 nsec, and 12 nsec for the 4000, 1000, and 6000 series, respectively.

3.1.2 NPN Emitter Follower

The npn emitter follower, shown in figure 3-3, with clamped loads at input and output, is often used as an auxiliary amplifier for positive-going set and clear trigger pulses where no signal inversion is desired. Its output impedance is low for positive-going pulses or level changes; current is supplied to the load by the transistor rather than by a load resistor.

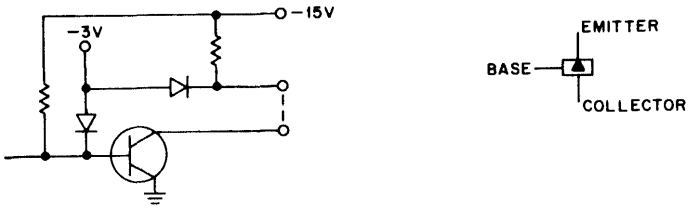


Figure 3-3 NPN Emitter Follower

Since, as a current amplifier, the npn emitter follower can operate with less drain on external trigger circuits than would a diode-clamped load configuration, it permits the use of remote trigger sources that may require long transmission lines and terminating resistors.

3.1.3 Flip-Flop with Output Buffers

Addition of output buffer amplifiers to the basic flip-flop substantially increases its driving capability and provides isolation from noise in the external load. When inverters are used as the amplifying element, the asserted output cannot be taken from the collector of the cutoff transistor. Instead the buffer inverter driven by the opposite collector provides the output (figure 3-4).

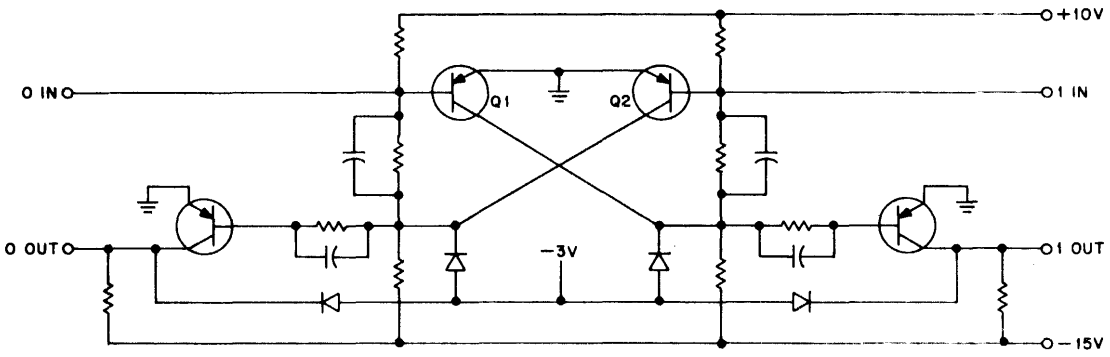


Figure 3-4 Flip-Flop with Output Buffer Inverters

The pnp output buffer transistor is saturated when turned on and therefore has considerable drive capability at ground assertion. At negative assertion, however, the drive capability is limited by the conductance of the clamped load resistor acting as a current sink. When increased drive capability is required at negative assertion, a push-pull output buffer is used. Figure 3-5 shows a basic flip-flop, Q1-Q2, which drives push-pull output buffer Q3-Q4 to develop the negatively asserted 1 output. When the flip-flop is set, Q2 conducts and Q3 is cut off, allowing Q4 to behave as an emitter follower driven by the −3 vdc level from the cutoff Q1 collector.

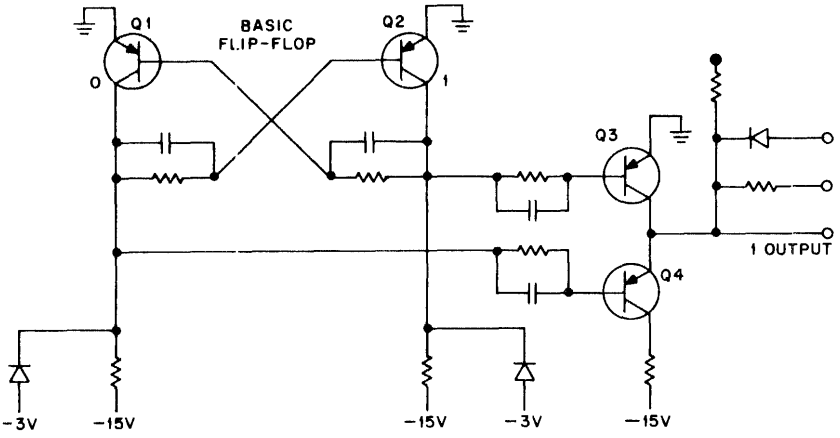


Figure 3-5 Push-Pull PNP Output Buffer

The current drive capability at the negative 1 output to the right of the figure is limited only by the Q4 collector resistance. Since Q4 never saturates, it can release the output for transition to ground level without requiring elimination of minority carrier storage caused by collector saturation. When the flip-flop is cleared, Q1 conducts and Q4 is therefore cut off; the Q2 collector load draws current through the Q3 base resistance to saturate Q3 and negate the 1 output at ground.

Push-pull complementary emitter followers (figure 3-6) are also used when the flip-flop is required to supply large amounts of current to the load at both logic levels. For ground level inputs, Q1 supplies current to the load through R3. The output point is then more negative than the Q1-Q2 base connection because of the forward drop of the Q1 emitter-base diode, and Q2 is therefore biased to cutoff. When the input is held at −3.75 vdc by D1-R2, Q2 conducts current from the load to −15 vdc through R4, and Q1 is cut off. R3 and R4 limit the load currents: this circuit can maintain a ground level up to 80 ma drain and a −3 vdc level up to 120 ma.

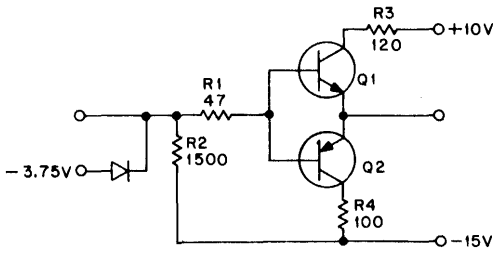


Figure 3-6 Push-Pull Emitter Follower

3.1.4 Pulse Amplifier

Pulse amplifiers (figure 3-7) standardize amplitude and width of carry pulses from one flip-flop in a register to the next. In the pulse transformer-inverter combination shown in figure 3-7, the pulse input point when quiescent is held to −8 vdc by the R1-D1 clamped load, through the low resistance primary of the transformer. The output terminal is at ground. The leading edge of a positive-going input pulse is coupled to the output as a sharp negative step (dots indicate relative winding polarities). A negative step of smaller amplitude is applied to the base of feedback transistor Q1 from the secondary center tap. The transistor amplifies and inverts this portion of the output pulse and returns it to the input, where it maintains ground potential even if the input pulse should prematurely disappear.

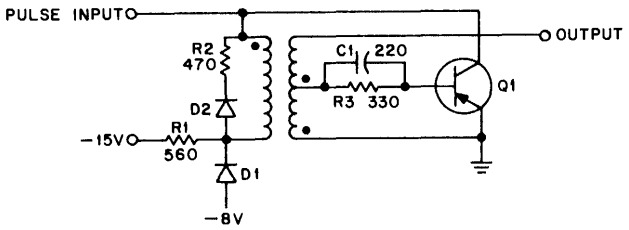


Figure 3-7 Pulse Amplifier

Output pulse width is therefore determined primarily by the pulse transformer and R1. The pulse ends when the transformer primary current reaches the value of current through R1. Current through clamp diode D1 is then reduced to zero, additional primary current increases the voltage drop across R1, and primary voltage begins to fall. The induced base drive to Q1 falls, reducing feedback, and the transformer voltages rapidly collapse. D2 and R2 prevent ringing by dissipating all stored energy in the first back swing after pulse completion.

3.2 1250 FAST MEMORY FLIP-FLOP

The Type 1250 is a quadruple module containing 24 flip-flops, interconnected to form three contiguous bits of eight flip-flop registers. In PDP-6, two groups of twelve Type 1250 modules make up a bank of sixteen 36-bit fast memory registers. Additional circuits include three diode readout gates and three noninverting digit amplifiers. A readout gate comprises eight 2-input negative AND gates with outputs ORed together, amplified and inverted. A digit amplifier is made up of two standard inverter amplifiers which supply incoming data to the register selected by the active write line.

Clear inputs are DEC standard 70-nsec positive-going pulses, applied separately to each register. Write signals are standard negative pulses which trigger the flip-flop collector through input buffer inverters. Buffer inverter emitters in each vertical column are gated by a single digit amplifier with output assertion at ground. Flip-flop registers can be sampled nondestructively and must be cleared before readin. The clear must precede the write by at least 100 nsec. Read signals are standard negative pulses. To sample one of the eight registers, the appropriate read line, ANDed with the 0 outputs of that register by the three diode networks, is pulsed. Transistors Q79, Q80, and Q81 invert the gate outputs, so the output of the sampled register appears at BZ, BL, and AE with negative assertion for 1s.

The operation of the 1250 Flip-Flop is identical to that of the basic flip-flop with one exception: clamped loads to −3 vdc are not used at the collectors of 1250 Flip-Flops. Instead, all flip-flop collectors are referenced through 1500 ohms to the parallel combination of R55 and R56, shown in the bottom right of the schematic. Since half of the 48 load resistors are at all times tied to grounded flip-flop collectors, load resistors for the off transistors operate into a voltage divider between −15 vdc and ground.

3.3 1260 SUBROUTINE CARD

This module contains three circuits, each comprising a flip-flop, a pulse amplifier, and two diode gates. Each circuit provides reentry to an asynchronous time chain by producing a single output pulse upon receipt of a restart pulse while the flip-flop is set (a logic diagram of this module precedes the schematic). The following describes the circuit including Q1 to Q6 and Q19. Flip-flop Q2-Q3 may be set either by grounding H or by applying a −3v pulse to K. While the flip-flop is 1 (−3 vdc at L, 0 vdc at H), a negative-going pulse or level change at J appears at E, amplified and standardized to 70 nsec duration. The flip-flop is cleared by grounding L or by applying a −3v pulse to the D1 or D2 cathode at F or E (a pulse at F clears the entire module). Note that the output pulse at E is applied back to the clear input of the flip-flop via D2 and Q1, so once the restart pulse has been received, no further output pulses can be produced until the flip-flop is again set.

The flip-flop used in the 1260 is the basic one discussed above. It is collector-gated with standard input buffer inverters, Q1 and Q4; the D5-D7-Q5 diode gate is also of standard design. The pulse amplifier (Q6-Q19-T1-T2) is similar to that in the Type 1609 Pulse Amplifier-Standardizer.

3.4 4214 QUADRUPLE FLIP-FLOP

The Type 4214 contains four flip-flops, each with both inputs and both outputs available at the connector. Each pair of flip-flops shares a common direct clear input at P or R. The module operates at any frequency up to 1 mc, using the basic flip-flop discussed above, but without internal capacitor-diode gates.

The common clear inputs require DEC standard 0.4-μsec positive pulses, which must precede readins by at least 1 μsec. The set and clear inputs to each flip-flop may be driven by positive capacitor-diode gates or by DEC standard 0.4 μsec positive pulses. Since all 1 and 0 outputs are available, the flip-flops may also be set or cleared at their collectors. Typical flip-flop output delay is 50 nsec.

3.5 4217 FOUR-BIT COUNTER

The Type 4217 contains four flip-flops, a pulse inverter-amplifier, positive capacitor-diode gates, and a negative dc supply. Depending upon external connections, the module may function as a counter or a buffer register with 1s transfer parallel readin. All input signals are applied to the flip-flops through positive capacitor-diode gates such as the C18-R37-D22 combination. The negative dc supply, consisting of D26 to D29 and R41, has a −0.75v tap at the D26-D27 junction and a −2.25v tap at the D28-D29 junction. Two capacitor-diode gates at each flip-flop have a common pulse input and are conditioned by the flip-flop outputs to form a complementing gate. Complement inputs for flip-flops A, B, C, and D are pins X, T, N, and H. Complement and clear signals are DEC standard 0.4-μsec positive pulses; the readin at F is a 0.4-μsec negative pulse.

The module may be connected as a 4-bit counter by tying the complement inputs of flip-flops B, C, and D either to the 1 outputs (up counter) or to the 0 outputs (down counter) of the preceding flip-flops. Count pulses are applied to pin X. For example, if the counter is initially clear and is connected for up counting, a pulse at X complements FFA to 1 because the gate is enabled by ground at the Q9 collector. The arrival of a second count pulse at X returns FFA to 0, causing its 1 output to rise to ground. This positive-going transition is coupled through the complement gate of FFB to the Q7 base, and sets FFB. Counting continues in this manner until the counter recycles to all 0s or is cleared by a pulse at E. The count may begin with any desired number, using the parallel readin gates and pulse input F. When the register functions as a down counter, the carry chain must be inhibited while the initial number is read in. A long time constant in the clear circuit prevents carry propagation when the register is cleared (this is necessary in the up count configuration).

3.6 4218 QUADRUPLE FLIP-FLOP

This module contains four flip-flops, a pulse inverter-amplifier, eight positive capacitor-diode gates, and a negative dc supply. Depending upon external connections, the Type 4218 may be used as a

shift register, or buffer register with jam transfer parallel readin. All input signals arrive at the flip-flops through positive capacitor-diode gates such as the D1-C11-R29 combination. The negative dc supply, consisting of D17 to D20 and R28, has a −0.75v tap at the D17-D18 junction and a −2.25v tap at the D19-D20 junction. The readin at P is a DEC standard 0.4-μsec negative pulse. Levels at E, H, S, and V gate the pulse at the set inputs of the flip-flops; levels at F, K, U, and X gate the pulse at the clear inputs. Gating levels must be present at least 1 μsec before readin for assertion (ground) and 2 μsec for negation. All four flip-flops are cleared simultaneously by a DEC standard 0.4 μsec positive pulse at R. A clear pulse must precede a readin by at least 1 μsec. Total flip-flop delay is less than or equal to 50 nsec.

For readin, complementary levels are connected to the two input gates of each flip-flop. The readin pulse thus loads (or shifts) 0s and 1s simultaneously, regardless of the prior states of the flip-flops.

3.7 4220 EIGHT-BIT BUFFER

This module contains eight flip-flops, each with positive capacitor-diode readin gates, a negative dc supply, and two pulse inverters, one to clear the flip-flops, the other to drive the gates. The gate driver inverter output is applied to an emitter follower whose low impedance output prevents cross coupling between capacitor-diode gates. For each flip-flop, the 1 input and a single output are available at the module connector. An output pin may, however, be connected either to the 1 or to the 0 output eyelet of its associated flip-flop with an internal jumper. The jumper code is an octal number representing an 8-bit binary number that identifies the output used for each flip-flop.

The clear and readin inputs require DEC standard 0.4-μsec negative pulses. A clear pulse must precede readin by at least 1 μsec. Level inputs are DEC standard signals, asserted at ground through positive capacitor-diode gates. Gating levels must be present at least 1 μsec before readin for assertion and 2 μsec for negation. Total flip-flop delay is less than or equal to 50 nsec.

3.8 4221 SIX-BIT SHIFT REGISTER

This module comprises six flip-flops, positive capacitor-diode gates, three pulse-inverter amplifiers, and a negative dc supply. The flip-flops are internally connected as a shift register and may be preset with a 1s transfer parallel readin. All signals are applied to the flip-flops through positive capacitor-diode gates such as the C1-R2-D42 combination. The negative dc supply, consisting of D34 to D37 and R5, has a −0.75v tap at the D36-D37 junction.

Clear, shift, and readin pulses at Z, X, and Y respectively are DEC standard 0.4-μsec negative pulses. The output of readin inverter Q15 drives npn emitter follower Q16, which provides a low impedance signal to prevent cross coupling between capacitor-diode gates. Pins R, S, T, U, V, and W are the

parallel inputs and are asserted at ground. Gating levels must be present at least 1 μsec before readin for assertion and 2 μsec for negation. A clear pulse must precede readin by at least 1 μsec. Total flip-flop delay is less than or equal to 50 nsec.

Flip-flops 1 to 5 have single output terminals (H, J, K, L, and M); each may be jumpered internally to either the 1 or the 0 output of the associated flip-flop. Both outputs of flip-flop 6 are available at the connector. An additional jumper provision allows the clear line to be connected either to the set or clear input of flip-flop 1. The octal number in the jumper code therefore represents a 5-bit binary number that indicates which flip-flop outputs appear at the connector. The number is followed by S or C to denote the connection of the clear line to flip-flop 1.

3.9 4225 EIGHT-BIT COUNTER

The Type 4225 contains eight flip-flops with complement inputs internally connected in a counter configuration, a single inverter used for presetting, and a negative dc supply. Both collectors of each flip-flop are available at the module connector and serve both as inputs and outputs. A series of jumper connections in the complement circuits of the register allows a choice of either binary or BCD counting (appropriate connections are indicated by B and D at the jumper eyelets). Flip-flops 4 and 8 have separate set and clear gates, rather than single complement gates, to facilitate alternate counter connections. For either mode, pin Y is the LSB count input and accepts a standard 0.4-μsec positive pulse or positive-going level change. The complement network for each flip-flop comprises two capacitor-diode gates with a common pulse input. The gates are conditioned by the flip-flop outputs. The preset inverter, Q17, accepts a negative pulse of 1 μsec minimum duration at Z. The pulse is inverted and applied through a capacitor-diode network to eight jumper eyelets. The register may be preset in any desired way by jumpering these eyelets to the appropriate flip-flop bases. The octal number in the jumper code represents the 8-bit number to which the module is preset. The number is followed by B or D to indicate binary or decimal operating mode.

3.10 4706 EIGHT-BIT TELETYPE RECEIVER

This module contains 14 flip-flops and a one-shot multivibrator (pulse shaper), with all gating implemented by positive capacitor-diode networks. Eight of the flip-flops make up a shift register; the remainder perform various control functions associated with the transfer of 8-bit Teletype characters from a unipolar line into the shift register. An 880-cps clock provides all timing at S. Usually the maximum transfer rate is 10 characters per second. The 100-msec period required for a single character transfer is divided into eleven equal unit intervals as follows:

START--one unit

8-bit character--eight units

STOP--two units (or more; prolonged STOP indicates idle line)

A timing diagram and a modified logic diagram accompany the schematic.

Between transfers, the Teletype receiver is inhibited by the STOP signal (ground) at T, holding the D26 cathode negative and inhibiting the ACTIVE set gate. Pin U must be at ground to enable the module. When START (−3v) arrives at T, the D26 cathode is grounded through inverter Q33, allowing R65 to enable the ACTIVE set gate. The following clock pulse at S sets ACTIVE, inhibiting the diode gate at the D24 cathode and generating a preset pulse through Q35 and its associated capacitor-diode gate. The preset pulse sets all shift register bits; it clears IN LAST UNIT and the MSB of the 3-bit counter (the other two are both 0 at this time). Since the counter is enabled when ACTIVE is 1, the next clock begins the first 8-count cycle. Every 4-count triggers the shift one-shot input gate, which is conditioned by the 0 state of IN LAST UNIT. The single pulse resulting at the Q24 collector is inverted by Q22, amplified by npn emitter follower Q21, and applied to the shift register input gates. Since 1 inputs at T are asserted at ground, the first shift transfers START into the register as a 0; the next eight pulses shift in the character. After seven bits have been read, the 0 in IN1 enables the set gates to IN LAST UNIT and FLAG: the eighth shift thus sets both flip-flops (FLAG lights an indicator mounted on the module). The 1 state of IN LAST UNIT inhibits the shift one-shot and enables the ACTIVE clear gate, C2-D31-R76, so the next 4-count clears ACTIVE. Since the STOP level (ground) is now present at T, the receiver is disabled until the arrival of the next START. For a standard 2-unit STOP code, jumpers C and D are present. IN LAST UNIT then enables the counter for one more cycle until the next 4-count clears it (this is the case shown in the timing diagram). For 1-1/2 units, B and D allow the next 0-count to clear IN LAST UNIT. For a 1-unit code the same 4-count that clears ACTIVE also clears IN LAST UNIT because of jumpers C and E. In any case the counter is inhibited when IN LAST UNIT clears, so it stops with the two less significant bits clear.

Transient inputs at T are prevented from activating the receiver by a diode gate which samples the outputs of all shift register bits and the pin T input. If the START level does not persist for at least five clock pulses, R64 enables an auxiliary ACTIVE clear gate, C23-R80-D32, which is pulsed by the first SHIFT, deactivating the receiver.

The flip-flops and positive capacitor-diode gates used in the 4706 module, as well as the npn emitter follower (Q21), are discussed at the beginning of this chapter. The one-shot multivibrator, comprising Q23 and Q24, is similar to the standard bistable multivibrator, or flip-flop, with the exception that no dc path exists from the Q23 collector back to the Q24 base. The device is therefore capable of only one stable state, in which Q24 conducts and Q23 is cut off. A positive pulse applied to the Q24 base momentarily cuts off Q24, driving Q23 into conduction. The positive transition at the Q23 collector is

coupled through C2 back to the Q24 base. The C2-R6 time constant primarily determines the duration of the negative level at the Q24 collector. As C2 discharges through R6, Q24 is eventually biased back into conduction and reverts to its quiescent state.

There is only one output terminal for each character flip-flop, and it can be connected to either the 1 or 0 side with an internal jumper. The jumper code thus includes an octal number equivalent to eight bits which indicate the outputs used. Following the number an F indicates a jumper that causes FLAG to be cleared when reception of a new character begins; an A indicates a jumper that allows ACTIVE to be sampled at pin M; and the letters associated with the STOP code jumpers indicate which of those are present.

3.11 4707 EIGHT-BIT TELETYPE TRANSMITTER

The Type 4707 Transmitter receives 8-bit characters from the PDP-6 I/O bus and sends out each character serially as a Teletype signal on a unipolar line. The module comprises 15 flip-flops and associated inverter and positive capacitor-diode logic. Ten flip-flops are connected as a shift register; the remainder perform various control functions associated with the character transfer. Timing is governed by a 220-cps clock input at T. Usually the maximum operating speed is 10 characters per second. The 100-msec cycle is divided into eleven equal units as follows:

START: one unit

8-bit character: eight units

STOP: two units (completing the cycle; longer indicates idle line)

A modified logic diagram and a timing diagram are included with the schematic. The module uses the basic flip-flop, capacitor-diode gate, and npn emitter follower discussed at the beginning of the chapter.

The transmitter is inactive until readin occurs at N, loading a character into the 8-bit OUT register and setting ENABLE, which in turn conditions the ACTIVE set gate through the D59-D61-D62 gate. The next clock sets ACTIVE, enabling the set input gate of a complementing flip-flop (FREQ DIV) and clearing OUT LINE to generate the START level at V. Subsequent clocks trigger the frequency divider, which generates a shift pulse through inverters Q23, Q22, and emitter follower Q21 on every transition to 0. The first shift moves the initial 1 from ENABLE into OUT8, shifts the character one place to the left, and sends out the first character bit by shifting OUT1 into OUT LINE. Each shift sends out a bit and clears ENABLE, so 0s follow the initial 1 through the register. The eighth shift places 1 in OUT1, followed by 0s in the remainder of the register and in ENABLE. This condition enables the ACTIVE 0 input through the large diode gate; thus the next and last shift clears ACTIVE while shifting 1 into OUT LINE.

The signal now at pin V is the STOP level, which usually must persist for at least four clock pulses. This time interval is supplied by a pair of STOP flip-flops that furnish a 4-count in the sequence 01, 11, 10, 00, provided jumpers A and B are present (this is the case shown in the timing diagram). The

0 transition in ACTIVE, besides setting FLAG (which lights an indicator mounted on the module), also sets OUT S1. The next three clocks continue the count, of which all steps except 00 inhibit the ACTIVE set gate in case a new readin has already occurred. The transmitter is reactivated by the first clock following the 00 count or a readin, whichever happens later. However, a negative level at R extends the inhibit indefinitely by holding OUT S1 on.

For a STOP code of 1-1/2 units, jumpers A and C enable the ACTIVE set gate at the same configuration, but the STOP flip-flops cycle through a 3-count: 01, 10, 00. For a 1-unit code, the absence of A enables the set gate on OUT S1(0), and C causes OUT S1 to set and clear on two clocks.

Either polarity may be selected at output V (and its opposite at W) by jumpering the desired OUT LINE output to the Q41 base. The jumper code thus includes a 0 or 1 to indicate the output used. An F indicates a jumper that allows FLAG to be cleared when a new character is received; the letters associated with the STOP code jumpers indicate which of those are present.

3.12 6203 SC AND FE FLIP-FLOPS

This module contains two 10-mc flip-flops, the ith bits of the 9-bit shift counter and floating exponent register. It is double size and is serviced by two standard 22-pin connectors (rear connector pin designations have the prefix R). A tap on the load resistance for the internal −3 vdc supply provides −8v for the pulse transformer primary clamp.

A logic diagram for the 6203 module, showing transistor numbers, is included with the schematic. In addition to standard inverter and diode gates, three special circuits are associated with the basic flip-flops and output buffers (all are discussed at the beginning of the chapter). Npn emitter followers Q11 and Q19 change SC state directly by grounding the appropriate flip-flop collector. The LC networks at the Q11 and Q19 bases delay the state change about 50 nsec from the leading edge of an incoming trigger pulse. The SC complement network, shown in the lower half of the logic diagram, operates entirely through this pulse delay to prevent race conditions. A pulse amplifier (upper left) is used for SC carry propagation.

Transistors Q12 and Q20 make up the basic SC flip-flop; Q8 and Q22 are its output buffers. A direct set input is provided at T, which triggers the flip-flop through npn emitter follower Q24; this input requires positive-going pulses from a pulse inverter, and includes the clamped load. SC may be cleared at RL through the pulse delay by a DEC standard 40-nsec negative pulse. The 1 and 0 outputs are at M and L; the resistor-coupled output at R may drive an indicator amplifier.

The FE flip-flop is unbuffered and may clear directly at the Q34 base by applying a 40-nsec positive pulse to RE. It may be set at < by a level or positive-going pulse to ground. The J output is resistor-coupled for connection to an indicator driver.

3.13 6205 ARITHMETIC REGISTER FLIP-FLOPS

This module contains four flip-flops, each the ith bit of one of the 36-bit processor registers, AR, MQ, MB, and MI. It also includes AR carry circuits. The module is triple size (in area) and is serviced by four standard 22-pin connectors. Pin designations are two letters, the first naming the connector (A, upper front; B, lower front; R, upper rear; S, lower rear), the second naming the pin. On the processor logic diagrams the front connectors are labeled by mounting panel. The internal −3 vdc supply for this module is derived from four series-connected silicon diodes, loaded to −15 vdc. A tap on the supply load provides −8v for the pulse transformer primary clamp. Three logic diagrams with transistor numbers for all flip-flops in the 6205 module accompany the schematic.

MI flip-flop logic, shown with MB, is considerably simpler than that of the other three, since MI is required only to store a memory bit and drive an indicator circuit. It is set by the 1 state of MB and a negative pulse at SN, through an input pulse inverter Q30. A negative pulse at SR clears the flip-flop through Q1; output point BS includes a series 3K resistor to drive the indicator.

Most of the input circuitry on AR, MB, and MQ consists of inverter networks that implement the many set and clear conditions required by logical, arithmetic, and transfer operations. All pulse inputs to the three flip-flops are bused to the rear connectors (R and S) from the arithmetic Bus Driver 6615, which produces 25-nsec pulses instead of the usual 6000-series 40-nsec pulses. To prevent race problems, flip-flop response time is carefully standardized at manufacture to ensure approximately 50-nsec delay between the input pulse leading edge and the signal response at the flip-flop output buffers. All input level conditioning is performed by the buffers (the basic flip-flop drives only the buffer), consequently input gating conditions cannot change until somewhat after the trailing edge of a pulse that changes the flip-flop state.

Trigger circuits on the 6205 differ from those on other modules: an emitter-driven pnp stage drives both basic flip-flop transistor bases through a pulse transformer. The clear and set triggers for all three flip-flops are identical; e.g., the clear trigger for AR (at the right in the schematic) consists of Q58 with pulse transformer T5. The Q58 base is clamped to −3.7v while its collector is loaded to −15v through the T5 primary. When one of the clear gates (at the top) is pulsed, it pulls the Q58 emitter toward ground so that Q58 saturates, placing approximately 13v across the T5 primary. The LC network at the Q58 collector introduces a short delay which adds to the transition time of input inverters, flip-flop, and output buffers to give the total flip-flop transition time. The T5 secondary simultaneously turns on Q74 through D29 and turns off Q75 through D31. Since both flip-flop transistors are pulsed, the output buffers provide simultaneously changing signals when the flip-flop is triggered.

The basic flip-flops, output buffers, and register clear inputs for AR, MB, and MQ are identical; all are 10-mc types using the push-pull pnp output buffer, and each has additional npn emitter-follower

input triggers that change flip-flop state by shorting the appropriate collector to ground. The emitter-follower pulse inputs at the front connectors are the only inputs that do not use the pulse amplifier trigger circuit described above. Clamped loads are included at the emitter-follower bases for the required external pulse inverters.

The AR circuits include an output pulse amplifier with additional networks for carry output gating and carry completion detection. The pulse amplifier consists of Q64 and pulse transformer T7 in the configuration described at the beginning of this chapter. If AR is 1, it delivers a 25-nsec negative pulse at AF when an input pulse appears at AH. The output is also triggered by a negative pulse at RH, if AR is 0 and MB is 1. These conditions are set up at two transistor AND gates, Q60-Q62, and Q61-Q63. A parallel output at AE is ORed with its counterparts on all other 6205 modules through D59 to detect the carry output; AE is therefore at ground potential only when carry propagation has terminated.

3.14 6206 MA, PC, AND IR FLIP-FLOPS

This module contains three flip-flops, the ith bits of the 18-bit processor registers, PC, MA, and IR. The module is double size and is serviced by a standard 22-pin connector at each end (rear connector pin designations have the prefix R). A tap on the load resistance for the −3 vdc internal supply provides −8v for the primary clamps of two pulse transformers. A logic diagram with transistor numbers is included with the schematic.

The MA flip-flop input circuits consist primarily of inverter networks that implement the memory address transfer logic. The diode comparison gate shown at the bottom of the logic diagram generates an output at K if the state of MA is the same as that of the corresponding memory address switch. Other MA circuits include a count net which complements the flip-flop through a delayed trigger circuit (Q15-Q23) when pulsed at P, and also generates a carry at N if MA contains 1.

The PC flip-flop circuits consist simply of a complement-carry network driving a pulse amplifier similar to that of MA, and a PC set gate. E is the count input for this flip-flop, and F is the carry output.

IR flip-flop circuits consist of a simple direct set network and two pnp-npn push-pull emitter-follower amplifiers (Q1-Q2 and Q10-Q11), one at each flip-flop output.

All 6206 circuits, including buffered and unbuffered flip-flops, npn emitter followers, push-pull emitter-follower amplifier, and pulse amplifier-standardizer, are discussed at the beginning of the chapter.

3.15 6227 EIGHT-BIT BUFFER

This module contains eight 10-mc flip-flops with no interconnections other than a clear line. Two inverters are provided for clearing: one, with input connection at P, clears a minimum of four and

a maximum of seven flip-flops; the inverter driven at R clears a minimum of one and a maximum of four. Clear signals at P and R are negative-going transitions resulting from level changes or DEC standard 40-nsec negative pulses; they are applied through Q17 and Q18 to the bases of the appropriate transistors. Positive-going levels or pulses applied directly to the flip-flop collectors can be used for individual setting and clearing.

Flip-flops with outputs T-S, V-U, and W-X may be cleared through either Q17 or Q18, depending upon internal jumper connections. Jumper identification for this module is therefore a 3-symbol code, one for each ambivalent flip-flop. P or R indicates the input connection; 0 indicates no connection.

CHAPTER 4
AMPLIFIERS

Amplifier types in the PDP-6 fall into two categories, pulse amplifiers and level amplifiers. The former are used not only to amplify short duration signals but also to shape and standardize them in amplitude and width; circuits for these modules are characterized by toroidal-core transformers, which generate pulses at relatively high currents with very short rise times. The level amplifiers are special-purpose modules usually associated with the in/out system and the memory bus; they function as solenoid, bus, and indicator drivers.

All PDP-6 pulse amplifiers except the Type 4606 operate in either the 5- or 10-mc speed range and must be driven by the collector of an inverter, which may be included on the module. The input to the base of this inverter must be a negative-going signal with an amplitude between 2 and 5v. For all 6000 series modules and for the Type 1609, the fall time of the negative pulse or level change must be less than 25 nsec; duration at 2v must be greater than 25 nsec. The remaining 1000 series modules require a fall time less than 50 nsec and a width greater than 50 nsec at 2v. Delay through a 5- or 10-mc pulse amplifier is approximately 25 nsec. The Type 4606 can be triggered by 2.5-4v level changes or pulses, of either polarity, having a rise (or fall) time less than 0.4 μsec and a duration at 2v greater than 70 nsec. For any PA, pulse amplitudes less than 0.5v cannot result in an output pulse. In general, a pulse amplifier produces output pulses characteristic of its speed line: 0.4 μsec, 70 nsec, and 40 nsec for the 4000, 1000, and 6000 series, respectively.

4.1 BASIC PULSE AMPLIFIER

To avoid repetition, this section describes a basic pulse amplifier that with minor variations forms a part of five modules. The schematic of figure 4-1 shows two input inverter configurations. Normally Q1 drives T1 directly. When Q1 is a high speed transistor (such as the MD94) with a breakdown rating below 8v, but the needed drive power to Q3 requires a −8v clamp at the T1 primary, common base stage Q2 is added to provide voltage standoff to Q1. The Q2 emitter (and Q1 collector) can go no more negative than −4.5v.

Under quiescent conditions, the transistor driving T1 terminal A is cut off so the T1 primary is held to −8 vdc by the D4-R4 clamp. Since no current flows in the T1 secondary at this time, Q3 is open circuited with its base and emitter at ground, and no current flows in the T2 primary. When A is pulled up to ground from its previous −8v level, the T1 primary inductance initially impedes current flow through it and R4 to the −15v supply, while the increasing magnetic field develops a sharp current pulse in the secondary (dots indicate the more positive winding polarities during the pulse). Q3 is immediately saturated

by this pulse, whose current amplitude is approximately proportional to the rate of change of the T1 primary current. As the R3-R4 junction rises toward ground potential, the primary current building toward the steady state maximum (10 ma in this case) proceeds more slowly, with a corresponding decrease in the T1 secondary current and voltage, until conduction through the Q3 base-emitter junction can no longer be maintained and Q3 cuts off. When the ground at A is removed, the reverse current spike generated by the T1 primary is damped out through R3 and D3. In addition to the inductance of the T1 primary and secondary, two other factors are primarily responsible for determining pulse width at the Q3 collector: the silicon V_{be} of Q3, and the emitter to ground potential established by R6. Increasing R6 decreases the pulse width and reduces T1 secondary current. Transformer T2 provides current amplification of the pulse generated at the Q3 collector. Positive output pulses are available at the T2 secondary when the negative terminal is grounded and vice versa. R5 and D5 damp the T2 primary backswing in the same manner as D3-R3.

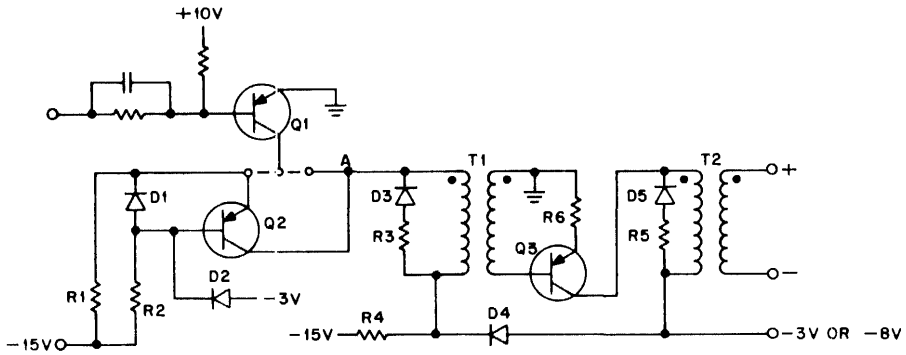


Figure 4-1 Basic Pulse Amplifier with Two Typical Driving Circuits

4.2 1607 PULSE AMPLIFIER

This module contains three of the basic pulse amplifiers (common-base input circuit) and three input inverters. The amplifiers, Q2-Q3, Q5-Q6, and Q8-Q9, are driven at H, L, and P. Their outputs are available at E-F, J-K, and M-N. Inverters Q1, Q4, and Q7 can be used in series for logic gating or with grounded emitters as individual triggering inputs. The triggering input to an inverter base is usually a DEC 70-nsec negative pulse, but the input requirement is satisfied by any negative transition meeting the specifications cited in the introduction to this chapter. The output generated by this signal is a DEC standard 70-nsec pulse, delayed by 25 nsec and capable of driving 16 units of pulse load.

Each amplifier has a separate 8v supply provided by a low-resistance voltage divider between −3 and −15 vdc (example: R11-R16). Capacitors such as C2 and C4 flatten the pulse top by maintaining a high rate of change of primary current for a longer time.

4.3 1608 PULSE AMPLIFIER

The Type 1608 contains two dual-output pulse amplifiers and two input inverters. The amplifiers are the basic type (common-base input circuit) with a duplicated second state. They are Q2-Q3-Q4 and Q6-Q7-Q8, and are driven at K and R. Outputs of the first are available at H-J and E-F; those of the second at N-P and L-M. Inverters Q1 and Q5 can be used in series for logic gating or with grounded emitters as individual triggering inputs. Input and output characteristics and all circuit parameters are identical to those of the Type 1607, except for the dual output stage that allows a single input pulse to produce a pair of output pulses of the same or opposite polarities. Since each pulse amplifier output is an independent transformer winding, transformers may be connected in parallel to increase the driving current, or in series to increase the pulse voltage amplitude for special purposes. Each output can drive 16 units of pulse load.

4.4 1609 PULSE AMPLIFIER

The six pulse amplifiers contained in this module differ from other PDP-6 pulse amplifiers by using feedback networks to transform very short-duration pulses into standard DEC 70-nsec signals. The Type 1609 is therefore useful as an interface between the 1000 and 6000 series. The amplifiers, Q1-Q2, Q3-Q4, Q5-Q6, Q7-Q8, Q9-Q10, and Q11-Q12, are driven at E, J, M, R, U, and X. Their outputs are available at H-F, L-K, P-N, T-S, W-V, and Z-Y. Pulse transformers in this module are referenced through suitable resistors directly to −15 vdc rather than to an intermediate negative supply. The internal −3 vdc source is loaded through 195 ohms to the −15 vdc supply, maintaining approximately 60-ma no-load current through the four series diodes. This relatively high current level reduces transients that occur on the −3v line when inputs are pulsed.

Pulse amplifier inputs must be driven from the collectors of one or more pulse inverters. The signal at the pulse inverter base must have a rise time less than 25 nsec; pulse width (at 2v) may run from 25 nsec up. When these input conditions are met, the output is a DEC standard 2.5v 70-nsec pulse that is negative if the positive terminal is grounded and vice versa. Outputs can drive 10 units of pulse load each and should not be used without terminating resistors. Although resistor values are ultimately dependent upon line length, 47 ohms is approximately correct when driving 1-5 units; 82 ohms is generally used to drive 6-10 units.

Consider the Q1-Q2 circuit as typical. It differs from the basic pulse amplifier in two respects: the T1 and T2 primaries are not clamped to −8 vdc, and the feedback circuit of Q1 has been added. To ensure a 70-nsec output from a pulse source as short as 25 nsec, Q1 is effectively placed in parallel with the driving source. A portion of the pulse amplifier output is fed back to the Q1 base by a separate T2 secondary winding. This signal saturates the transistor, maintaining the ground level at E for 70 nsec, when the circuit constants end the pulse. Capacitors C1 and C2 aid in flattening the output pulse top.

4.5 1664 MEMORY BUS CONTROL

This module contains four gated pulse amplifiers and a 7-input AND gate for positive signals (expandable to 9). It is quadruple size, with four 22-pin connectors, and has been designed specifically for use in the PDP-6 two-way memory bus system. The Type 1664 accepts a DEC standard 70 nsec negative pulse or a level change meeting the requirements for the 1000 series set forth in the beginning of this chapter. Each pulse amplifier output drives the center of a 93-ohm coaxial cable (50 ohms dc to ground) with a 100-nsec negative pulse.

Type 1664 modules are used in PDP-6 memory to decode the most significant bits of a processor memory address and, after priority system recognition, to implement various control and timing functions between the addressed memory and the processor having current access. PDP-6 can include a maximum of four processors and every memory must contain one 1664 for every processor. The address of a given memory is wired into its 1664s by jumpering the appropriate memory address lines (asserted at ground) at K or L, M or N, P or R, and T or U to four of the decoder gate inputs.

When 8K rather than 16K banks are used, an additional address line may be selected at V or W to accommodate a larger number of memories. Replacing 16 locations in core with a fast memory requires further access to the decoder at X or Y. The 8K and fast memory inputs may be gated separately and brought out to BC, or may be joined with the remaining decoder inverters, with output at BT. Two remaining inputs to the decoder indicate memory ready (BR) and a processor request for memory (RF). Since the latter is asserted negative, it is inverted before being applied to the decoder.

A negative output at BP therefore indicates a processor request for the memory to which the module is assigned; this signal is routed to the priority system, which grants access to the processor by enabling the level input to the four pulse amplifiers on the module--BW, AH, BU, and AM. Subsequent negative memory timing pulses at BY, AE, BS, or AK are placed on the memory bus at B, C, D, or E of either connector and may be gated with negative levels at BZ, AJ, BV, or AN to produce pulse outputs at BX, AF, BT, or AL; the negative level inputs may also gate information from the bus to the pulse output terminals.

The four pulse amplifiers on the 1664 module, of which the Q1-Q2-Q3 circuit is typical, differ considerably from other PDP-6 pulse amplifiers in that a blocking oscillator technique is used to provide the high energy pulses and virtually zero quiescent load required by the memory bus--a feature impossible to duplicate with conventional pulse transformer outputs. The blocking oscillator proper consists simply of T1, Q3, R4, R5, R7, and R8. The Q1 collector is the oscillator input; D1-D4 form a standard negative diode gate which drives the Q1 base through inputs BY and BW. The oscillator output at the Q3 collector is gated with BZ by a similar diode-transistor network to produce ground assertion at the Q2 collector, pin BX.

The 100-nsec negative pulse is produced by T1 and Q3 as follows: under quiescent conditions (Q1 cut off), approximately 3.6 ma flows from ground through R5, R4, T1, and the parallel combination R7-R8 to the 15 vdc supply. The transformer primary is the winding in series with 750 ohms; the secondary is placed directly across the Q3 base-emitter junction. Since the dc resistance of the T1 windings is low, both base and emitter of Q3 are at nearly the same potential (approximately −14.5 vdc), so that Q3 is cut off. When Q1 is gated on, its collector rises to ground, effectively shorting R5. The Q3 base voltage therefore becomes positive, saturating Q3. The Q3 collector is pulled up to approximately −2.8v if load impedance is 50 ohms. At this time there is approximately 12v across R8 and across the series combination of R7 and the T1 primary. Current in the primary sets up an increasing magnetic field, which is coupled to the T1 secondary to maintain the voltage at the Q3 base. A large amount of positive feedback is coupled across T1 in this manner, keeping Q3 in saturation while current flow increases through the T1 primary. As the rate of current increase lessens, being limited by R7, induced current in the T1 secondary starts to drop until Q3 goes out of saturation, whereupon the rapidly collapsing primary field accelerates Q3 cut-off and the pulse ends. Although a degree of interaction exists between R7 and R8, R7 functions primarily to determine output pulse width by limiting current through the T1 primary, whereas R8 limits output pulse amplitude. Pulse overshoot is damped in the conventional manner by D5 and R3.

4.6 1665 PULSED BUS TRANSCEIVER

This module is quadruple size and, like the Type 1664, is designed specifically for use in the PDP-6 two-way memory bus system. It comprises 18 blocking-oscillator pulse amplifiers, each with two-input negative AND gates at both input and output. The circuits of the Type 1665 Pulse Amplifier are identical with that of the 1664, although the two modules function differently in the system. Type 1665 modules control and amplify the two-way flow of data at either end of the memory bus: between the bus and a processor and between the bus and a memory. Since each module handles 18 bits, two 1665s are required for data at each processor, and two per processor at each memory. Furthermore, each memory requires an additional 1665 per processor to handle the receipt of addresses, but for this application only the output AND gates are used.

The Q1-Q2-Q3 network is a typical one. Q3 and D6 to D9 form a standard NAND conficuration for negative inputs; this gate is driven at BV and BX (the latter is common to all input gates on the module). The pulse amplifier output connects directly to the bus and to one input of a similar NAND with output at BW (the second input to this gate, BY, is common with corresponding inputs on all other pulse amplifier output gates). In a processor or fast memory, data is made available to the bus from a flip-flop register. Therefore a level at BV represents a bit to be transferred, and the transfer occurs when BX is pulsed. A data pulse on the bus at RB is brought in through the output gate at BW when a negative level

is present at BY. In a core memory, single data bits are represented by pulses from the sense amplifiers and the read is destructive, so information sent out must usually also be routed to interim flip-flop storage for later rewriting. Thus a pulse at BV is gated onto the bus by a level at BX, and a pulse going in either direction on the bus--that is, a pulse produced by the pulse amplifier or one coming in on RB--can be brought in to BW by a gating level at BY.

Both core and fast memories receive addresses through 1665s using only the output gates. A processor applies the address register levels to the bus, and these incoming levels are pulsed through the output gates at CY.

4.7 4606 PULSE AMPLIFIER

This module contains three pulse amplifiers which produce DEC standard 0.4-μsec, 2.5v pulses. Each circuit comprises a monostable multivibrator and an output pulse amplifier. It may be triggered at the appropriate input by 2.5-4v level changes or pulses of either polarity, having a rise time less than 0.4 μsec and a duration at 2v greater than 70 nsec. Maximum operating frequency is 1 mc.

Each pulse amplifier accepts three inputs: a positive pulse, a negative pulse, or a gated negative pulse; each input is one unit of pulse load. A level change of the same polarity can replace the pulse at any input. The gated pulse input is enabled by applying a DEC standard −3 vdc level to the gate level input; this level must be present for at least 1 μsec before the pulse arrives. When a signal meets the input requirements, the pulse amplifier delivers a positive-going output pulse if the negative terminal is grounded and vice versa.

Each amplifier, e.g., Q1-Q4, comprises a monostable multivibrator (Q1 and Q2), an input pulse inverter (Q3), and an output pulse amplifier (Q4). Capacitor-diode C1-D1 couples a negative input at E to the Q2 base; C7 couples a positive input to the Q3 base, and the resulting negative pulse at the Q3 collector is applied directly to the Q2 base. Thus an appropriate pulse at either input triggers the multivibrator with a negative pulse. The negative output pulse from the multivibrator is amplified by the Q4 circuit; the amplifier output is negative at H if J is grounded, positive at J if H is grounded.

L and K provide an additional, capacitor-diode input to the amplifier. A capacitor-diode gate couples an input pulse to its output when (and only when) an input gating level is asserted. For example, in order to couple a negative pulse at K to the Q2 base, a −3v level must be present at L. When L is at ground, the C5-R13 junction is near zero potential. Since the D6-C6 junction is quiescently at −3 vdc, a negative pulse at K cannot forward-bias D6. A −3v level at L, however, results in 0v across D6, so a negative pulse at K is coupled by D6 and C6 to the Q2 base.

Quiescently, Q1 is on; Q2 and Q4 are off. Base current for Q1 flows through R5 to the −3 vdc supply, holding Q1 in saturation. Voltage divider R2-R6 maintains a small positive voltage at the Q2 base,

keeping Q2 cut off. D2 clamps the Q2 collector to −3 vdc. Voltage divider R3-R4-R7 biases the Q4 base positive, holding it off. The Q4 collector potential is approximately −7v, as determined by voltage divider R10-R11. No current flows in the T1 primary, and no voltage exists across the secondary.

When a negative pulse appears at input E, C1 and R1 differentiate its leading edge, generating a negative pulse at the D1 cathode. D1 is now forward biased, so the signal appears at the Q2 base. As Q2 turns on, its collector voltage jumps from −3v to ground. This positive step is coupled by C2 to the Q1 base. Q1 now cuts off and its collector potential drops to −3v. R6 holds Q2 on, even though the input pulse has ended. The multivibrator remains in this state until the coupling capacitance from the Q2 collector to the Q1 base discharges. Discharge time is proportional to capacitance and is approximately 0.4 μsec. At the end of this interval, Q1 turns on, cutting Q2 off, and the multivibrator returns to its quiescent state.

The negative pulse generated at the Q1 collector turns on Q4, whose collector rises to ground, placing approximately 7v across the T1 primary. C4 flattens the pulse top by maintaining a high rate of change of primary current for a longer time (secondary output voltage is proportional to the rate of change of primary current). The pulse terminates when the multivibrator returns to its quiescent state, cutting off Q4. D5 and R18 clamp the T1 primary overshoot.

A positive pulse at F (inverted by Q3) or a gated negative pulse at K triggers the same chain of events to produce an output pulse across J and H. D8 and R15 form a clamped load to −3 vdc which can be connected with an internal jumper to the positive input line. The presence of a jumper is indicated by a 1 in a 3-bit number (written as a single octal digit) in which the bits left to right correspond to inputs F, N, and V.

4.8 6603 PULSE AMPLIFIER

This module contains three 10-mc pulse amplifiers of the basic type (common-base input circuit) and three input inverters. All circuit details and pin connections are identical with those of the Type 1607, the only differences being operating speed and the method of obtaining the −8 vdc source for the pulse amplifier primaries. The Type 6603 produces a DEC standard 40-nsec output pulse, delayed by 25 nsec and capable of driving 12 units of pulse load. Input signals to the base of the driving inverter must conform to the specifications outlines in the beginning of this chapter for 6000 series modules. A 4.7v zener diode, D24, is placed in series with the standard −3 vdc internal supply, resulting in a hard source of −7.7 vdc at the D24-R25 junction. The primaries of the pulse amplifier output transformers are connected to this point; first-stage transformer primaries are clamped to a point 0.75 vdc more negative.

4.9 6609 PULSE AMPLIFIER

This module contains six 10-mc pulse amplifiers of the basic type. They are driven at E, J, M, R, U, and X; their outputs are available at H-F, L-K, P-N, T-S, W-V, and Z-Y. Type 6609 amplifiers must be driven by the collectors of external pulse inverters. Signals at the bases of these inverters must conform to the specifications set forth in the beginning of this chapter for 6000 series modules. When a signal meets the input requirements, the output is a DEC standard 40-nsec pulse, delayed by 25 nsec and capable of driving 12 units of pulse load.

The module is identical in all other respects to the Type 1609, with the exception of the 1609 feedback circuit. The pulse transformer primaries are returned directly to the −15 vdc supply, rather than to an intermediate supply, and the primary overshoot of the output transformer is lower-clamped directly to −15 vdc.

4.10 6615 ARITHMETIC BUS DRIVER

Four Type 6615 modules are used in the arithmetic processor to gate arithmetic logic and control signals, which arrive at the front connectors, and to generate gated 25-nsec pulses which are fed to buses at the rear to drive the Type 6205 arithmetic register Flip-Flops, AR, MB, MQ, and MI. Each 6615 drives 18 Type 6205 Input Gates in parallel. Jumpering provisions at both input and output permit a variety of logical operations to be performed by this module; in practice, however, input jumpering is the same for all modules, and outputs are jumpered in only two ways. Four 6615 Pulse Amplifiers do not require jumpering at input or output; these implement the AR and MQ shift functions.

The 6615 module uses the basic pulse amplifier. The input transistor is a simple inverter circuit; the common base stage is not required since it is not necessary to mix from inverter collectors--all input pulses are negative. All transformer outputs are terminated with resistors at the 6205 input points.

The base of the carry detection inverter, Q31, is driven through R81 at pin E by a 9-diode negative OR gate whose inputs are tied to the carry outputs of nine AR flip-flops. The presence of a negative carry pulse at any input saturates Q31, resulting in a ground level at output F. Since this terminal is wired in parallel with the corresponding terminals on the other three 6615s, the four load resistors are selected so the value of their parallel combination approximates that of a standard clamped load (1500 ohms). Consequently when all four transistors are cut off, pin F drops to −3 vdc, indicating that the carry function is complete.

4.11 1669 INDICATOR DRIVER

This module contains nine inverters used as transistor switches in indicator lamp circuits. All emitters are grounded; inputs and collector outputs are available at the connector. Operation and

component values are the same as those of the basic inverter described in chapter 1, with one exception: due to the relatively slow circuit operating speed, no capacitors are required across the input resistors. The usual loads are GE 327 bulbs, connected between the collector output pins and −15 vdc. Nominal current output is 30 ma.

4.12 1684 BUS DRIVER

The Type 1684 is a noninverting amplifier used to drive heavily loaded logic lines at speeds up to 5 mc. The module contains four circuits, each comprising two inverters and an output stage that permits the bus to be driven at either logic level by a transistor rather than a clamped load. Inputs are pins K, M, U, and S, each input representing one unit of base load. Outputs at pins L, N, T, and R have a characteristic impedance of 22 ohms and can drive up to five units of base load each. Driving power can be increased to 30 units per output by connecting a 120-ohm resistor between C and E. Resistor coupled outputs at H, J, V, and W increase output impedance when series-driving a bus.

The bus driver circuit, e.g., that with input K and output L, operates as follows: When a ground level is applied to input K, the base of Q1 is held above ground by R2; this holds off Q1, so that its collector is at approximately −4 vdc. R3 then supplies turnon current to the Q2 and Q4 bases through R5 and R4. The Q4 collector is now slightly below ground (provided that output loading does not bring the transistor out of saturation). Since Q3 is cut off by the ground at the Q2 collector, L is at ground with an output impedance of 22 ohms.

When the input level changes to −3 vdc, Q1 saturates, turning off Q2 and Q4. D1 limits reverse bias to the Q2 base and quickly discharges C3. As the Q2 collector falls toward −15 vdc, Q3 turns on and its collector-base diode clamps the Q2 collector to approximately −4 vdc. The emitter of Q3 (for any load that permits the transistor to operate at saturation) is about −3.75 vdc. Because Q4 is cut off, the no-load voltage at output H is therefore −3.75 vdc. Output impedance is still 22 ohms, the value of R9. The −3.75 vdc supply is similar to the −3 vdc supplies of other modules except for an additional series-connected diode. However, unlike standard inverters, all current from negative outputs of this module must pass through R41, the resistor for the supply. Consequently, R41 limits current into negative outputs to approximately 20 ma (5 ma per amplifier). An external 2-watt, 120-ohm resistor can be added in parallel with R41 to increase output current available at −3 vdc to 120 ma.

4.13 4658 I/O BUS TRANSCEIVER

The Type 4658 is designed specifically for the PDP-6 two-way I/O bus system. The four circuits on the module can each drive up to twenty resistor inputs of capacitor-diode gates distributed along a 93-ohm coaxial cable. Access to each amplifier is through a two-element negative OR gate: data inputs are E, K, S, and W; a select level at L is common to all. Levels at these inputs must persist for at

least 2 μsec. Ground-asserted outputs at F, N, T, and X drive the level inputs of capacitor-diode gates; complements of these coaxial outputs at H, P, U, and Y are used to gate internal registers; outputs J, R, V, and Z usually drive Type 327 28v indicator lamps that are returned to −15 vdc. A ground level asserted at M resets the bus by providing a strong negative level at F, N, T, and X that rapidly discharges the CD gate capacitors. When data is transmitted in the other direction, i.e. when a bus line at F, N, T, or X is grounded externally, pin H, P, U, or Y is asserted negative.

The circuit of Q1 to Q4 is typical of the four amplifiers in the 4658 module. Q1 and Q2--an inverter-and emitter-follower combination--and the negative NAND gate circuit of Q3 have a common output at F. When pin M is at ground, R3 cuts off Q1 by holding its base at ground. With Q1 cut off, its collector is pulled negative toward −15 vdc by R5. This voltage, after current amplification by emitter-follower Q2, appears at F. The Q2 emitter is lower-clamped to −3 vdc, while the Q2 collector potential is stabilized by C2. A negative level at M turns on Q1, resulting in a ground at the Q2 base and therefore at F. D1 to D4 and Q3 form a standard NAND configuration for negative logic levels. When E and L are both negative, R4 pulls the D2-D3 junction negative, turning on Q3 and placing a ground on the bus at F. If either or both gate inputs are positive, Q3 is cut off and the R8-D6 clamped load pulls F down to −3 vdc. Level shifters D3 and D4 ensure good turnoff of Q3.

Inverter Q4 is driven at F, either by the Q1-Q2-Q3 circuit or externally via the I/O bus. When F is at ground, the potential at the D10-R14 junction is approximately −0.75v. Leval shifter diodes D7, D8 and D9 allow R11 to pull the Q4 base positive by about 2.25v, so Q4 is cut off. The −3 vdc level at the Q4 collector appears at H and J. When F is driven to −3v, R14 pulls the D9-D10 junction negative until it is clamped by the Q4 emitter-base junction, D7, D8, and D9 to approximately −2.5v. Since D10 is now cut off, the Q4 turnon current does not flow in the bus, but to the −15 vdc supply through R14. Turning Q4 on results in a ground level at H and J.

4.14 4681 SOLENOID DRIVER

This module contains three driver circuits that operate as switches, transforming DEC standard levels into outputs suitable for driving solenoids. Each driver can control an inductive load, such as a punch solenoid or a typewriter relay, supplying up to 500 ma at operating potentials up to −80 vdc. The switch inputs are K, M, and R; outputs are L, N, and P. Pin E is connected to the external load return voltage source.

A typical circuit operates as follows: When the input at K is −3 vdc, the solenoid driver is quiescent. Since the Q1 emitter is at ground potential, Q1 is saturated and its collector is at ground. The emitter of Q2 is at 2.25v (the forward voltage drop across D1 to D3). The Q2 base at ground is positive with respect to its emitter, so Q1 is cut off. Under these conditions, the load circuit is open:

no current flows in it and the output is at the load return voltage. When input K is grounded, Q1 cuts off and its collector drops toward −15v, turning on Q2. The load circuit is now completed, allowing current to flow. D4 connects the Q2 collector to the external negative supply, protecting Q2 from the high negative transient that results when the inductive load circuit is opened.

4.15 6684 BUS DRIVER

The 6684 contains four dual-purpose, noninverting bus drivers and associated biasing circuitry. Each driver provides DEC standard levels either to a cable terminated in 100 ohms or directly to a series of base and emitter loads. Circuit delay is approximately 50 nsec.

Each amplifier comprises two series-connected inverter stages driving a push-pull emitter-follower output stage. The load resistor for the second-stage inverter may be clamped to either of two voltages depending upon the output being driven: −3.75 vdc is used for direct driving at H, L, T, and Z; −6.75 vdc is used when driving 93 ohm cables via resistor-coupled outputs at F, N, R, and Y. Each direct output can drive 40 units of base load or four units of dc emitter load at 10 mc; each resistor-coupled output provides DEC standard levels at the cable termination that can drive ten units of base load or one unit of dc emitter load at 10 mc. A 4-digit jumper code specifies the bias used with each amplifier, e.g., 6684-3336.

Consider the circuit of Q1 to Q4 as typical. A −3 vdc level applied to the Q1 base saturates Q1; the resulting ground at the Q1 collector permits R9 to cut off Q2. The Q2 collector is now clamped by T5 and R13 to either −3.75 or −6.75 vdc, depending upon the position of the internal jumper. The negative level is coupled to the common base junction of Q3 and Q4 by R39. Since the amplifier output (the common emitter junction of Q3 and Q4) is referenced to −2.7 vdc by R17 and R25, the emitter-base junction of Q4 is forward biased, while Q3 is cut off. Emitter-follower Q4 therefore conducts current from the load through limiter resistor R3 to the −15 vdc supply; output voltage is either −3 or −6 vdc. When the input level at E returns to ground, Q1 cuts off and Q2 saturates. The ground level at the Q2 collector turns on Q3 and cuts off Q4. Emitter-follower Q3 now conducts current through limiter resistor R29 to the load at H or F, maintaining the output at ground.

CHAPTER 5
DELAYS

The modules described in this chapter provide adjustable delays for standard DEC pulses. Some units make use of delay line techniques to generate comparatively short delay intervals; others use monostable multivibrators to generate wider delay ranges and supply level outputs. The Type 4303 Delay can be extended indefinitely by repetitive triggering.

5.1 1304 DELAY

This module contains input pulse gate Q1, monostable multivibrator Q2-Q3, output level amplifier Q4, and output pulse amplifier Q5. When a DEC 70-nsec pulse grounds input X either directly or through the inverter, level output J switches from its quiescent ground level to −3v for a predetermined interval, then returns to ground. When the level rises at the end of the delay, the circuit generates a pulse at E or F, depending upon pulse polarity desired.

The delay interval is determined primarily by a capacitor connection (three capacitors are available internally, but an external one may be used). Using internal components, the delay may be varied from 0.25 - 500 μsec in three ranges. With U jumpered to T, potentiometer R5 varies the delay within any range. With no other external connection, C3 provides the minimum range. Other ranges are selected by jumpering H to N or M, thereby adding C4 or C5 into the multivibrator circuit. Delay ranges for these connections are as follows:

None	0.25 - 2.5 usec
H-N	2.5 - 35 usec
H-M	35 - 500 usec

If external delay control is desired, a potentiometer may be connected between S and T. Higher delay ranges may be added by connecting an external capacitor between L and K.

In the quiescent state, Q1, Q3, and Q5 are off; Q2 and Q4 are on. Q1 is held off by a ground at its base or -3 vdc at its emitter. A current of 2.5 ma flows from −3 vdc through D15, R20, the T1 primary and R7 to the -15 vdc supply; 5.5 ma flows from -3 vdc through D2, R21 and R7 to −15 vdc. The resulting 8 ma through R7 maintains the Q1 collector at about −4v. Since a steady current flows in the T1 primary, no voltage appears across the secondary. Current flowing from the Q2 base through R3 and the parallel combination R5-R6 (when T is connected to U) holds Q2 on.

When a pulse at X or Y triggers the 1304 Delay, the resulting 4v drop across the T1 primary is coupled to its secondary winding. Terminal 4 now goes negative as current through the primary increases linearly to 10 ma from its quiescent value of 2.5 ma. During this increase the voltage across

the primary at first remains nearly constant due to the clamping action of D2 and R21. When the T1 primary draws more than 8 ma, however, D2 is back biased, and R21 is removed from the primary circuit. As the primary circuit resistance is thus increased, the primary voltage decreases more rapidly.

D1 couples the negative pulse at T1-4 to the Q3 base; Q3 then turns on and its collector rises to ground. The low forward resistance of D7 and R11 shunts the T2 primary so no current flows in the secondary. The junction of D4 and R12 now rises to ground, turning off Q4. The output level at J then drops to -3v, indicating the start of the delay interval. The ground at the Q3 collector is applied through D5 to pin H where it is coupled to the Q2 base by C3, C4, or C5 (depending upon the connection made to H). Q2 now cuts off and its collector drops to −3v. R4 then draws base current from Q3, holding on Q3 even after the pulse at T1-4 ends.

The monostable multivibrator remains in this state, Q2 off and Q3 on, for the time interval required to charge the capacitance in the Q2 base circuit. The RC time constant, which determines this interval, depends upon the capacitors in use, and the resistance of R3 in series with the parallel combination R5-R6. When the time delay capacitors have charged to a sufficiently negative voltage, Q2 turns on. The rise in its collector voltage is coupled through R4 and C6 to the Q3 base: Q3 then cuts off, causing the T2 primary current to fall to its quiescent level. The resulting negative pulse in the T2 secondary is applied to the Q5 base, turning it on. The pulse amplifier comprising Q5 and T3 generates a DEC standard 70-nsec pulse across outputs E and F. C8 flattens the output pulse top. When the T2 secondary voltage drops to 0, Q5 turns off and the output pulse ends. As Q3 turns off, the junction of D4 and R12 returns to -3v, turning on Q4. Level output J therefore returns to ground, indicating the end of the delay.

5.2 4301 DELAY

Except for delay ranges and pulse widths, this module is identical to the Type 1304 Delay. Five internal capacitors enable delaying 0.4-usec input pulses by 2.5 usec to 200 msec in five ranges. Potentiometer R7 can be used to vary the delay within any range by jumpering U to T. With no other external connections, C4 provides the minimum range. Other ranges are selected by jumpering H to N, M, P, or R, thereby adding one of C5 to C8 into the multivibrator circuit. C4 provides a delay of 2.5 - 25 usec. The additional values of capacitance (C5 to C8) increase the range by successive factors of approximately 10. Circuit recovery time is 20% of the maximum delay in the selected range. If external fine control of the delay interval is desired, a potentiometer may be connected between S and T. Larger delay ranges can be achieved by connecting an external capacitor between L and K. All 4301 input and output pins, with the exception of two additional delay capacitor connections at P and R, are identical to those of the Type 1304.

5.3 4303 INTEGRATING DELAY

This module contains two monostable multivibrators, two difference amplifiers (slicers), three inverters, a positive capacitor-diode gate, and a negative dc supply. The 4303 circuit has flip-flop-type logic level outputs: in the 1 state, the 1 output is at -3 vdc, the 0 output is at ground; in the 0 state, output polarities are reversed. When any of three inputs is pulsed, the circuit assumes the 1 state; it returns to the 0 state after a selected delay, which begins with the termination of the input signal.

The multivibrators are Q2-Q3 and Q10-Q11; the slicers are Q5-Q6 and Q7-Q8. The inverters are Q1, Q4, and Q9; the capacitor-diode gate is C1-R1-D1. Voltage levels of −0.75, −1.5, −3, −4.5, −4.5, −5.25, and −6 vdc are tapped from the internal negative supply at the cathodes of D16, D15, D13, D11, D10, and D9. Three inputs are provided at pins K, S, and R. The K input may be either a DEC standard 0.4-μsec negative pulse or a negative level. The S input may be either a DEC standard 0.4-μsec positive pulse or a positive-going level change; this pulse is gated to the Q2 base by a ground at T. Input R requires a positive pulse such as the output of a positive capacitor-diode gate similar to C1-R1-D1. The 1 and 0 outputs are W and U.

The delay is variable from 3.4 μsec to 0.9 sec in five overlapping ranges. Connection of one of five internal capacitors determines the delay range. C7 is permanently connected and gives the shortest range. Connecting C6, C8, C9, or C10 (E, F, H, or J) to ground (D) increases the range by successive factors of approximately 10. Potentiometer R10 determines the delay within each range if Y is jumpered to X. Alternatively, an external potentiometer connected between Z and X may control fine delay.

In the quiescent state, Q1, Q3, Q4, Q5, Q8, Q9, and Q10 are off; Q2, Q6, Q7, and Q11 are on. Voltage divider R2-R3 shifts the ground level at input K positive, biasing Q1 off. Multivibrator Q2-Q3 is in its stable state, with Q2 held on by base current flowing through R5, and Q3 held off by voltage divider R8-R9. With the Q2 collector at ground, voltage divider R11-R12 biases Q4 off.

In the first slicer, Q5 is off and Q6 is conducting. D6 clamps the Q6 collector at −0.75v, holding Q6 out of saturation. The common-emitter connection of npn transistors Q5 and Q6 follows the voltage (−1.5v) at the base of Q6. Consequently Q5 is cut off. The other slicer transistors, Q7 and Q8 are on and off, respectively. The series combination of R13, R14, and potentiometer R10 draws sufficient current from the Q7 base to saturate that transistor, even if the potentiometer is set at maximum resistance. The saturation of Q7 holds the common emitter connection of Q7 and Q8 at −5.25v, the Q7 collector voltage. The more positive voltage (−4.5v) at the Q8 base holds it at cutoff. Q9 is therefore back biased through R19 and is also cut off. The voltage divider made up of R23 and the parallel combination R21-R24 holds Q10 off. Consequently, multivibrator Q10-Q11 is in its stable state, with Q10 off and Q11 on.

The circuit is triggered from the quiescent state when an appropriate input at K, R, or S drives the Q2 base positive. Multivibrator Q2-Q3 flips to the temporary state with Q2 off and Q3 on. Diode D4 clamps the Q2 collector to −3v. R11 and C5 couple this −3v level to the Q4 base, saturating it. The grounded collector of Q4 drives the bases of Q5 and Q7 positive. Q7 drives output multivibrator Q10-Q11 to its temporary state; Q5 forms part of the feedback loop that returns the signal to input multivibrator Q2-Q3, causing it to revert to its stable state.

As the Q7 base goes positive, Q7 turns off and Q8 turns on, so their common emitter connection rises to the Q8 base voltage. Q8 does not saturate because its rising collector voltage is clamped at −5.25v by the Q9 turnon. (Q9 does saturate and its collector drops to −5.25v). R20 couples the Q9 collector voltage to the Q10 base, turning it on. Multivibrator Q10-Q11 flips to the temporary state, with Q10 on and Q11 off. The output is now in the 1 state (W at −3v, U at ground).

In the feedback loop to input multivibrator Q2-Q3, R14 drives C7 (and any other capacitor that may be in parallel with it) toward ground. When this voltage reaches approximately −1.5v, Q5 saturates heavily, forward biasing both its base-emitter and base-collector junctions. The larger part of the Q5 base current flows to its collector, which is consequently more negative than its emitter. Q6 is thus cut off. The Q6 collector goes positive, forward biasing D5 and cutting off Q3. Q2 now turns on (provided it is not held off by a continuing positive level at R). This grounds the Q4 base, cutting off Q4 and initiating the discharge of the timing capacitors. Before the Q7 collector can return to its quiescent state, C7 (and any additional capacitor in parallel with it) must discharge through R13 and potentiometer R10. As C7 begins to discharge, Q5 turns off and Q6 turns on. D5 becomes back biased, disconnecting R16 and R17 from the Q3 base circuit. From this time on, even though the full delay interval is not yet over, the circuit can be retriggered by a positive pulse at S, or a negative input at K.

When the Q7 base becomes more negative than −5.4v, Q7 turns on, and Q8 turns off. The −6v supply back biases Q9 via R19, turning it off. The rise in voltage at the Q9 collector cuts off Q10, returning the output multivibrator to its stable state. The output returns to the 0 state (W at ground, U at −3v), indicating the end of the delay.

If the Q1 input is held at −3 vdc, C7 charges up to −1.5v and stays there, and the output remains in the 1 state. Return to the 0 state occurs a fixed interval after the removal of the negative Q1 input. When input pulses arrive at shorter intervals than the delay period, the output similarly remains in the 1 state, returning to the 0 state a fixed interval after the last pulse.

5.4 DELAY LINES

These modules contain one or more delay lines that provide relatively short delays for standard negative pulses, although the circuits can also be used to delay level transitions. With each line is an inverter driven by the delay line output; the collector, often the emitter, and occasionally the base of the inverter are available at the external connector. To simplify coaxial or twisted-pair connections, individual ground return pins are adjacent to all collector outputs. Every line has taps so that various delays can be selected up to the total length of the line. Modules in which the taps are available at the external connector contain one or two lines; those in which delays are selected by internal jumpers contain six. The jumper code is six digits which correspond left to right to the alphabetical order of the output pins. A 1 in a given digit indicates the shortest nonzero delay on the corresponding line, a 2 indicates the next longer delay, and so forth. The delay configuration and the pin connections or jumpers for selection in each module are shown in the circuit schematics. In every case, the delay across the output inverter must be added to the delay selected: 20 nsec in the 1000 series modules; 12 nsec in the 6000 series.

For the operation of a typical delay line, consider the Type 1310. A negative pulse at input X causes a ground pulse after a predetermined delay at inverter output E. A clamped load connected to E (or some electrically equivalent point) holds it negative while the inverter is off. When the circuit is quiescent, R1 furnishes current to hold Q1 off. Terminating resistors R2 and R3 prevent signal reflections from the ends of the delay line. R4 and R5 are used as attenuators when selecting a shorter delay, to match the attenuation of a signal traversing a greater line length. D1 isolates the input from reflections caused by mismatch at the output tap.

CHAPTER 6
PULSE CIRCUITS

The PDP-6 uses four pulse circuits, Pulse Generators 1410 and 4410, and Clocks 4401 and 4407. A pulse generator is similar to a pulse amplifier, since it produces a DEC standard pulse whenever its input is triggered, but it differs in that it can be triggered by slow, irregular voltage changes such as those produced by mechanical switching. The clocks generate trains of 0.4-μsec pulses: the 4401 frequency is adjustable, but the 4407 frequency is predetermined by the crystal oscillator it contains.

6.1 1410, 4410 PULSE GENERATORS

These two pulse generators have identical circuits and pin connections; they differ only in output pulse width: the 1410 output is a DEC standard 70-nsec pulse; the 4410 output is a DEC standard 0.4-μsec pulse. The following description uses the component designations for the 1410.

The module contains a Schmitt trigger circuit (Q1-Q2), an output pulse amplifier (Q5), and an RC filter (R1-R2-C1). It generates an output pulse whenever its input drops from a value more positive than −1.0v to a value more negative than −2.0v. If no filtering is required, the input is applied to S. However, if the internal filter is needed (e.g., when the circuit is used in conjunction with a mechanical switch), S and U are jumpered and the input applied to Z; the negative trigger input to Z is then obtained by mechanically switching K to Z. The combination of capacitor filtering and the hysteresis of the Schmitt circuit prevents the switch contacts from generating more than one pulse per closure. The output terminals are E and F; the output pulse is negative at E when F is grounded, positive at F when E is grounded.

For an initial ground input at S or Z, Q1 is off and Q2 is on. The Q2 emitter follower action holds the Q1 and Q2 emitters to about −1.8v. Divider R5-R6-R9-R10 determines Q2 base voltage. D2 isolates Q1 and Q2 emitters from the −1.0v level at the R12-R13 junction. The Q2 collector is quiescently at −4v. The current through the T1 primary is held steady at about 10 ma, and no voltage appears across the secondary. The Q5 base is therefore held at ground through the T1 secondary so that Q5 is cut off, with its collector at approximately −7.5v.

The circuit is triggered by a negative voltage applied to its input. Q1 begins conduction when its base becomes more negative than its emitter; i.e. when the input falls below −2.0v. As Q1 conducts, its collector voltage rises toward ground. R6 and C2 couple the rising Q1 collector voltage to the Q2 base, thereby cutting off Q2. This pulls the common emitter connection of Q1 and Q2 more positive, speeding the Q1 turnon by positive feedback. This method assures a fast change of state, independent of the fall time of the input signal. The T1 field collapses with the turnoff of Q2, inducing a negative pulse

at T1-4. The pulse turns on Q5, whose collector rises rapidly to ground, causing a pulse to appear across the T2 secondary, pins E and F. C4 flattens the pulse top by maintaining a more constant rate of change of current in the T2 primary over the pulse period.

When the Q1 base input again rises toward ground, the Q1 emitter follows until its base is at approximately −1v. At this point, D3 is forward-biased and clamps the emitter voltage. Further input voltage rise cuts off Q1; the resulting drop at the collector is coupled through R6 and C2 to the Q2 base, turning Q2 back on and returning the trigger circuit to its initial state. Current is now reestablished through the T1 primary; the resulting positive pulse at T1-4 only drives Q3 further into cutoff and the output pulse amplifier is unaffected. D1 and R11 prevent ringing in T1.

6.2 4401 CLOCK

This module comprises an astable multivibrator (Q1-Q2), a pulse amplifier-shaper (Q3), and an output pulse amplifier (Q4). The Type 4401 generates DEC standard 0.4-μsec pulses at E or F at any frequency from 5 cps to 500 kc. The interval is divided into five overlapping ranges; within each range, the output frequency is continuously adjustable. The output pulse train is inhibited by applying −3 vdc to a disconnect diode whose anode is connected to V.

The frequency range is determined by the amount of capacitance between T and V; one of five capacitors may be selected by an external jumper. Connecting T to M selects the range of 5-50 cps; connecting T to R, P, N, or U increases the range by successive factors of 10. Connecting an external capacitor between V and T will extend multivibrator frequency range. Potentiometer R4 adjusts frequency within the selected range when X and Y are jumpered. An external fine control potentiometer can be connected between Y and Z.

Diodes D1 to D6 determine multivibrator operating voltages: a tap at the D3-D4 junction holds the Q1 base at −2.25 vdc; −3 vdc for the Q1 and Q2 collector supply is provided at the D4-D6 junction. Since R4 varies the Q1 operating point over a wide range, a dual collector load is provided for Q1. R3 is principal load for low operating current; at higher currents, D5 conducts and the load is shared by R2.

Q1 and Q2 commutate at a rate that is a function of the RC time constant of the range-determining capacitor (one of C3 to C7) and the series combination of R1 and R4. An output pulse is generated during each cycle of the multivibrator at the Q2 turnoff. Assume that Q1 is turning off and Q2 is turning on. The Q2 emitter voltage follows the negative-going voltage at the Q1 collector. C3 (if T is jumpered to M) couples the negative transient to the Q1 emitter. This feedback rapidly triggers the multivibrator to its first state, Q1 off and Q2 on. The T1 secondary now generates a positive pulse at the Q3 base, which serves only to drive Q3 further into cutoff; the output pulse amplifier is therefore not affected.

The multivibrator remains in this state while C3 charges through R4, R1, Q2, and the T1 primary, pulling the Q1 emitter toward ground. When the emitter voltage becomes more positive than −2.25v (the

Q1 base potential), the transistor turns on and Q1 collector-emitter feedback triggers the multivibrator to its second state, Q1 on and Q2 off. Note that if V is held at −3 vdc, Q1 cannot turn on. The turnoff of primary current in T1 generates a negative pulse at T1-3. This pulse saturates Q3, grounding T2-2 and turning on Q4 with a negative pulse at the base. The output pulse amplifier (Q4, T2, T3) generates a DEC standard 0.4-μsec pulse across E and F. This pulse is positive at F if E is grounded, negative at E if F is grounded. C8 flattens the output pulse top by maintaining a high rate of change of T3 primary current for a longer time. The T3 primary overshoot is damped through R8 and D7.

The multivibrator remains in the second state while C3 discharges through R5, Q1, and the Q1 collector load. When the discharge current decreases to about 6 ma, the Q2 emitter becomes more positive than its base, so Q2 begins to turn on, turning off Q1 and starting a new cycle.

6.3 4407 CRYSTAL CLOCK

This module contains a crystal oscillator (Q1-Q2-Q3), an oscillator output amplifier (Q4), a Schmitt trigger (Q5-Q6), and an output pulse amplifier (Q7). It generates DEC standard 0.4-μsec pulses, either positive or negative, at a frequency determined by the oscillator crystal. This frequency may be between 5-500 kc: in the PDP-6 Teletype interface the crystal is series resonant at 14.08 kc. Negative pulses are obtained at E or positive pulses at F; the unused pin must be grounded.

Positive feedback from the collector to the emitter of Q1 sustains oscillations at the resonant frequency. This feedback is obtained through the common emitter junction of Q1 and Q3. Since the impedance of the crystal is minimum at resonance, positive feedback is maximum at this frequency. The parallel resonant circuit C1-C2-L1 helps to stabilize the gain of Q1 and makes it possible to tune the oscillator through a narrow band about the crystal resonant frequency. Zener diodes D1 and D2 isolate the oscillator from supply variations. Q3 collector output is applied through Q4 to the Schmitt circuit, Q5-Q6, which converts the oscillator output to a square wave.

Assume Q5 off and Q6 on. Q6 emitter follower action holds their emitters at approximately −1.8v. Divider R11-R13-R14-R16 determines the Q6 base voltage. Diode D5 isolates the emitters of Q5 and Q6 from the −1.0v level at the junction of R17 and R18. The Q6 collector rests at −4v. Current through the T1 primary is held steady at about 10 ma, and no voltage appears across the secondary. The T1 secondary holds the Q7 base at ground, so Q7 is cut off. The Q7 collector is, therefore, at approximately −7.5 v, with the T2 secondary quiescent.

The Schmitt circuit is triggered when the Q5 base falls below −2v. R13 and C4 couple the rising collector voltage of Q5 to Q6, cutting it off and pulling up the common emitter connection of Q5 and Q6. This positive feedback ensures a fast change of state, independent of input signal fall time.

As Q6 turns off, the T1 field collapses, inducing a negative pulse at T1-3. This negative pulse is amplified by Q7 and appears across the T2 primary, and hence across E and F. The pulse amplifier shapes the output to 2.5v amplitude and 0.4-μsec duration. C7 flattens the pulse.

When the Q5 base input rises toward ground, the emitter follows until it reaches approximately −1v. At this point D5 again conducts, clamping the emitter voltage. Q5 cuts off as its base rises above −1v. The resulting negative transition at the Q5 collector turns Q6 back on, returning the trigger to its initial state. Current flow is thereby reestablished through the T1 primary, resulting in an induced positive pulse at T1-3. This positive pulse, however, drives Q7 only further into cutoff, so the output pulse amplifier is unaffected. D4-R15 and D6-R20 prevent ringing in the transformer primaries.

CHAPTER 7
POWER CIRCUITS

The PDP-6 power supplies convert standard 110 vac to dc power. They include the Types 728 and 778 which supply the +10 and -15v required by the logic, and the Type 734 whose output, variable from 0 - 20v, may be used for marginal checking. A power control governs line voltage to the power supplies and to motors in peripheral equipment. The particular control used in any situation depends upon the requirements of the system; some units provide two-step control, some provide additional control functions. One control, Type 823, allows the logic to govern operation of the punch motor and is mounted directly on the motor. All other power supplies and controls are mounted on the rear plenum door of a standard DEC equipment bay, although in the arithmetic processor, the 734 Supply is mounted behind the console in/out indicator panel.

7.1 728 POWER SUPPLY

This unit supplies the power required by most of the PDP-6 modules. Outputs are +10vdc at 0 - 7.5 amp and -15 vdc at 1 - 8.5 amp. When both outputs are used concurrently, the available current is limited as follows: -10 vdc, between 0 and 7.0 amp, -15 vdc, between 1 and 8.0 amp; and combined, by the relationship

$$5i_{+10} + 6i_{-15} = 53,$$

where i is current supplied.

The +10 output is regulated to between +9.5 and +11v, the -15 output, between -14.5 and -16v. This regulation holds for minimum to maximum load in the presence of line variations from 105 to 125 vac. Output ripple is less than 350 mv.

T1 steps down the line voltage to 10-0-10 vac and 15-0-15 vac. D2 and D3 are connected to the 10v secondary taps as a positive full-wave rectifier. C2 and C4 filter out the ac component of the output. R1, in parallel with the 10v load, maintains the output within regulation tolerances even when no current is drawn externally. D1 and D4 are connected to the 15v secondary taps as a negative full-wave rectifier. C1, C3, C5, and C6 are filters.

The design simplicity of the Type 728 is made possible by the fact that the saturated-core resonant transformer (T1) incorporates inherent overload protection. Even with shorted outputs, only a limited current can be drawn. This self-limiting feature obviates series impedance elements at the filter inputs. Thus the dc output impedance is kept low, rendering regulating devices unnecessary.

7.2 734 VARIABLE POWER SUPPLY

This unit furnishes dc power for marginal checking of the PDP-6 modules. For a nominal 110 vac input, output voltage is continuously variable from 0 - 20 vdc (no load). Maximum output voltage drops 3v at full rated load of 2.5 amp.

Line power at 110 vac is stepped down by transformer T1 (only half of the secondary is used), which drives an M5 Variac. By adjusting the position of the Variac tap, any voltage within the range of 0 - 20 vac is available (clockwise rotation increases voltage). The Variac output is applied to a bridge rectifier, whose diodes are oriented so that the dc output at the D2-D4 junction is positive with respect to the D1-D3 junction. Parallel capacitor C1 filters the output; R1 improves voltage regulation for small load currents. A slow-blow 5-amp fuse at the positive output protects the supply against overload. The output voltage is indicated on a 0-30 vdc meter.

7.3 778 POWER SUPPLY

The Type 778 is a dual floating 15 vdc power supply, capable of supplying 8.5 amp from each channel. The two supplies each contain a transformer, a full-wave silicon rectifier, and three filter capacitors.

In each unit, a resonant transformer steps down the line voltage to 15-0-15 vac. The diodes are connected as negative full-wave rectifiers. The simple design of this supply is made possible by use of saturated-core resonant transformers that provide inherent overload protection; even with shorted outputs only a limited current can be drawn. This self-limiting feature obviates series impedance elements at the filter inputs. Thus the dc output impedance is kept low, making regulating devices unnecessary.

The outputs remain between -14.5 and -16v over variations in load from minimum to maximum and in line voltage from 105 - 125v. Maximum peak-to-peak ripple is 1.0v; required line frequency regulation is 60 ± 0.5 cps.

7.4 811 POWER CONTROL

This unit controls power turnon and turnoff in various peripheral devices. An 811 in the arithmetic processor supplies line voltage to the reader, punch, and keyboard-printer whenever computer power is on (the power control output to the punch is actually applied to the 823 SCR Power Control).

As long as the circuit breaker is closed, K1 applies line power to the control output through contacts A and B. The unit includes a two-position switch to allow a choice of local/remote control and two additional relays K2 and K3 to provide a means of control by remote signals. All relay contacts are normally open. The K1 coil is energized by 115 vac, K2 and K3 by -15 vdc. Wiring connections for each application are made at the 10-terminal Jones strip, and a diagram of the configuration is included in the appropriate system maintenance manual.

As an example of a typical application, consider the situation in which terminal 3 is connected to 8, 6 to 7, and 7 to 1. Then the position of the switch that connects 1 to 3 provides for local power turnon by energizing K1 either directly if 7 is wired to 9, or by means of an external power switch if 7 is connected to 9 through such a switch. With no connection to terminal 2, the other switch position provides for remote turnon so that −15v applied to 5 energizes K2 bypassing the switch. Relay K3 allows delayed turnoff if desired. At turnon, the −15 vdc remote signal charges C1 through D4; at turnoff, C1 energizes K3 momentarily, delaying the K1 release.

7.5 823 SCR POWER CONTROL

The Type 823 is an ac power control capable of high speed response to low power logic signal inputs. The controlling element is a high speed reed relay designed to form the collector load for any standard DEC pnp stage among the 4000-series inverter and diode modules. The relay in turn triggers a silicon controlled rectifier which accomplishes the primary power switching function.

The silicon controlled rectifier is a four-layer pnp device that behaves much like a gas thyratron. At reverse-bias voltages below the avalanche value, the SCR blocks the flow of current. Under forward-bias conditions the SCR also blocks current flow, but it can be switched into a high-conduction state by a small positive current from gate to cathode. The SCR can thus control very large amounts of power in response to a low power signal. Once it is in the high-conduction state, it continues to conduct indefinitely, even after removal of the gate signal, until the anode current is interrupted or diverted. It then regains its forward-bias current blocking capability.

In the Type 823, the SCR works in conjunction with a 4-diode bridge, which is in series with the punch motor across the power line. One side of the 115-vac 60-cycle power line is connected to AC IN. Under control of the SCR, power flows through the bridge to AC OUT, through the punch motor, and back to the ac line. Shown at the left is K1, the low power reed relay that operates from a logic level input to control the trigger input to the SCR gate.

Assume initially that the SCR is blocking forward current flow. R2 holds the gate close to cathode potential. When the logic provides ground level at INPUT, K1 energizes and its contacts allow positive gate current to flow through R1 into the SCR gate, to trigger the high-conduction state. During positive half-cycles across the power line, current flows from AC IN through D2, the SCR, D3, and the motor back to the line. During negative half-cycles current flows through the motor and returns to the line through D1, the SCR, and D4. When the logic releases the ground level at INPUT, K1 removes the gate current through R1, and the SCR reverts to its forward current blocking condition when the instantaneous line voltage swings through zero.

7.6 829 TWO-STEP POWER CONTROL

This unit is used in situations that require a two-step application of power such as in a core memory, where core drive circuits must not receive power until after the logic does but must be turned off first. The 829 is used in the Type 161 Core Memory and in some arithmetic processors. Wiring connections for use with the memory are shown in the memory control drawings, for use with the processor in the console drawing.

The left secondary of the transformer shown at the left supplies a 60-cycle signal which may be applied to a pulse generator for use as a real-time clock. The other secondary drives a diode bridge that supplies unregulated filtered 15 vdc for the relays. Line power applied to the control input is coupled to the outputs through contacts 2 and 3 of K3 and K4. K1, a fast on/slow release relay, picks up K3, which couples the input to the upper output. K2, a slow on/fast release relay, energizes K4 to provide power at the lower output.

If TP7 is connected to TP8, turning S1 to local energizes K5, which in turn energizes both K1 and K2. Through the upper contact sets, K1 energizes K3 immediately, K2 energizes K4 after a delay of 3 - 5 sec. The lower contact sets provide the same type of operation for any desired external switching function. For example, if TP17 or TP20 is connected to signal ground (at the logic), TP16 and TP19 remain at ground for several seconds after turnon to enable a power clear function. When S1 is turned off, K2 drops out immediately but K1 holds K3 on for several seconds.

If S1 is in remote, the power control can be turned on by an external switch connected across any of the TP pairs 1-2, 3-4, or 5-6. Of course remote turnon can also be accomplished by applying an external −15v to TP1, 3 or 5, directly to TP7, bypassing the local/remote switch.

7.7 832 TWO-STEP POWER CONTROL

This unit is used in situations that require a two-step application of power such as in a core memory, where core driving circuits must not receive power until after the logic but must be turned off first. K3 connects line power to the upper output, K2 to the lower output. Al three relays operate on 115 vac. Energizing K1, a fast on/slow release relay, in turn energizes K3. If power is desired at the delay on/fast off output, closing S2 allows K2 to be energized whenever K1 is energized. For typical use terminal 1 is connected to 3, and 5 to 6. Then turning S1 to local picks up K1, and picks up K2 through S2. Line voltage appears immediately at the upper output, but, since K2 operates slowly, there is a 3 - 5 sec delay before power appears at the lower output. When S1 is turned off, the lower output goes off immediately but the upper output is delayed. For remote control, S1 is turned to remote and the power control is turned on and off by an external switch connected across terminals 1 and 2.

7.8 834, 835 ONE-STEP POWER CONTROLS

These units provide single-step power application by means of a remote switch closure. The 835 is the main power control for the arithmetic processor, and the wiring connections for that application are shown at the upper right in the console drawing. The two power controls are identical except that the 835 contains a filament transformer whose 60-cycle output drives a pulse generator to provide a real-time clock.

For standard applications terminal 1 is connected to 3, and 5 to 6. Turning S1 to local energizes K1, whose contacts couple the line voltage from the control input to the output. For remote control S1 is turned to remote and K1 is then controlled by an external switch connected across terminals 1 and 2.

CHAPTER 8
MODULE REPAIR

All DEC modules are exhaustively checked for faulty or marginal operation, both prior to their incorporation into the computer and during operation as a functional part of the system, to make malfunctions due to defects of workmanship or components as unlikely as possible. But module failures sometimes do occur, however seldom, and some provision must be made to restore them to proper operation. After locating a faulty module, the user may choose from two courses of action: he may return the module to DEC or attempt to repair it himself.

DEC recommends return of all defective modules since the facilities available at the plant ensure that all possible causes of failure can be investigated and eliminated. However, if there are appropriate test facilities in the field, well-staffed with engineering and test personnel competent in computer maintenance, the user may wish to try his hand. This chapter is a guide to module repair in the field. Replacement components may be obtained either directly from DEC or commercially. Equivalency charts on the schematics show the JEDEC numbers corresponding to DEC part numbers. Almost all non-inductive components except distributed-constant delays are available through local electronic part distributors.

8.1 TOOLS AND TEST EQUIPMENT

In addition to standard shop tools, including a 6v soldering iron with isolation transformer, the following are recommended:

Multimeter	Simpson model 260A, Triplet Model 630NA, or equivalent
Subminiature alligator clips	Mueller Type 30 or equivalent
Oscilloscope	Tektronix Type 581 (50 mc) with Type 82 dual-trace vertical preamp, or equivalent
Long-lead probes	Tektronix P-6002 or equivalent
Current probe	Tektronix P-6016 or equivalent
Module puller	DEC Type 1960 (supplied without charge)
Module extender	DEC Type 1954 (supplied without charge)
Pigtail module extender	Modified DEC Type 1954 (below)

In complex installations or in multiple installations, an independent bench module tester may be desirable. A tester for DEC plug-in modules can be made up from a Type 722 Power Supply, a

commercial variable signal generator (10 mc maximum), standard DEC signal-generating modules (e.g., Pulse Generator 1410, Variable Clock 6401) and suitable switching circuits. A desirable addition to the tester is a 734 Power Supply, which permits marginal checking of plug-in units at the bench.

A pigtail module extender can readily be fabricated from a standard DEC Type 1954 Extender. Disconnect the small wire leads to terminals A, B, and C of the extender, and solder 8-foot leads to the three terminals. Solder alligator clips to the free ends of the leads. This permits convenient marginal testing of an individual module while it is in place (through the extender) without disturbing the remainder at the mounting panel.

8.2 REMOVAL AND REPLACEMENT

It is advisable to turn off system power before extracting or inserting a module. Modules can be removed conveniently with the DEC Type 1960 Module Puller. Carefully hook the small flange of the puller over the center of the module rim and gently pull the unit out of the mounting panel. Use a straight, even pull to avoid damaging the plug connections or twisting the etched circuit. Since the puller does not fasten to the module, prevent it from falling by holding the rim in one hand as you pull it from the mounting panel.

When replacing a module, always position it so that the component side of the board is to the right and the printed wiring to the left. The aluminum rim extends along the bottom edge beyond the plug. When a module is properly installed, this extension fits into a matching slot in the mounting panel. Should the module be upside down, the extension prevents it from making contact with the mounting panel socket. Carefully slide the module between the guide ridges embossed on the mounting panel surfaces until the plug makes contact. If plug and socket are properly aligned, gentle pressure is sufficient for full insertion. Do not force the connection if plug and socket are not aligned. Occasionally, slight movement of the module within the guide ridges may be necessary to match the plug with its connector.

The tape reader printed circuit cards (mounted on top of the reader chassis) are removed by grasping the top of the card with the thumb and index finger of one hand and the edges of the card with the thumb and fingers of the other hand. A gentle pull disengages the card from its socket. The card then slides straight up and out of the mounting panel. A card is inserted by guiding it into the two panel slots and allowing it to slide downward until it contacts the socket. A slight downward push fully engages the connections. Component sides of these cards are to the right when observed from the reader front.

Connections to power supplies and controls are made both by wiring at barrier terminal strips and by cables terminating in plugs. Although the barrier strip wiring is color coded, these markings may be absent from the chassis of the power unit. Before removing or replacing a power unit, clearly mark all unlabeled connections both on the unit to be removed and on the spare to be installed. After disconnecting the unit, release it by removing the Phillips-head mounting screws on both sides.

A large number of DEC modules are equipped with jumpers to increase their versatility in performing logic functions. Nomenclature for such a module on the logic diagrams includes a code defining jumper locations for that module in the given mounting panel position. The codes differ among module types and are noted in the circuit descriptions. Spare modules are supplied with all jumpers in place, indicated usually by an R. Before replacing a module remove all jumpers on the spare except those required and mark the spare with the correct code.

8.3 MODULE TROUBLESHOOTING

Malfunctions within a single module may be located in two ways. The first, in situ or active circuit troubleshooting, involves use of the module extender, the pigtail extender, the oscilloscope, and small two or three-instruction exercise loops like those described in chapter 9 of the arithmetic processor manual.

The second method is bench troubleshooting, involving use of a suitable multimeter, the DEC equipment listed in section 8.1, an oscilloscope, and standard bench test equipment such as an in-circuit transistor and diode checker, a regulated bench power supply, etc. A bench tester may be built up from DEC modules to provide active-circuit troubleshooting (signal tracing) independently of the computer. Information on system design with DEC modules, helpful in assembly of a bench tester, is in the Digital Modules Catalog (A-705A) available from DEC without charge.

8.3.1 In-system Troubleshooting

The extender permits troubleshooting of many modules while the system is operating. There are, however, exceptions to use of the extender: the high operating speed of the 6000-series modules may not allow them to function properly on the extender, due to the added inductive coupling of the extender leads. Moreover, no extender is provided for double- and triple-size modules; if these cannot be repaired at the bench, return them to DEC. To use the extender turn off system power, remove the module and insert the extender in its place, plug the module into the exposed end of the extender and finally, turn the power back on. To apply marginal voltages to an individual module, use the pigtail extender instead, and furnish pins A, B, and C with appropriate marginal check voltages via the three clipleads.

Active troubleshooting can often be simplified if knowledge of PDP-6 logic enables the user to exercise the questionable module with the computer. A repeat switch and speed control on the console allow repetition of the key functions with a delay adjustable from 8 sec down to 3.4 μsec.

CAUTION

Although DEC circuits are designed with internal safeguards which prevent damage from opening or shorting the output terminals on a single unit, they are not proof against all accidental shorts which might be produced while testing the unit on an extender. Care must also be exercised when testing terminals on the wiring side of the racks.

8.3.2 Bench Troubleshooting

If a simple inspection fails to reveal the cause of the trouble, multimeter resistance readings can often isolate it. Resistance readings allow a rough check on emitter-base diodes of transistors (it is essential to determine the internal battery polarity of the multimeter; this polarity is sometimes opposite to the normal polarity of the leads). Note that although incorrect resistance readings are a sure indication that the transistor is defective, correct readings give no guarantee that the transistor is good; it may have other troubles. Several inexpensive in-circuit transistor and diode checkers are available commercially--those generally provide a more reliable indication of diode or transistor quality.

Damaged or cold-soldered connections can also be located with a multimeter. Set the multimeter to the lowest resistance range and connect it across the suspect junction. Poke at wires and components around the connection with a probe or hand. Often the response time of the multimeter is too slow to detect rapid transients produced by intermittent connections. Current interruptions of very short duration can be detected by placing a 1.5v battery, in series with a 1.5K resistor, across a connection and observing the voltage across the resistor with an oscilloscope.

Use a 6v soldering iron with an isolation transformer to remove or replace defective components. Avoid excessive heat which can cause damage to components and might even delaminate the etched wiring. Semiconductors require special care--wherever possible, attach a copper alligator clip or other heat sink to the lead being soldered, to reduce the amount of heat transferred to the component.

digital
EQUIPMENT
CORPORATION
MAYNARD, MASSACHUSETTS

9114 PRINTED IN U. S. A. 1-5/66

DATA CONTROL TYPE 136

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

DATA CONTROL
TYPE 136
INSTRUCTION MANUAL

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

COPY NO. 236

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

ILLUSTRATIONS

TABLES

ENGINEERING DRAWINGS

ENGINEERING DRAWINGS (continued)

SECTION 1

INTRODUCTION AND DESCRIPTION

INTRODUCTION

The purpose of this instruction manual is to aid personnel in the installation, operation, and maintenance of the DEC Data Control Type 136.

Anyone who wishes to learn the details of the system logic can save considerable time by first noting the conventions used on the engineering drawings. These conventions are explained in Appendix 2. It is recommended that this section of the manual be scanned before reading Section 2.

Abbreviations

The following abbreviations are used throughout this manual.

CCT	Character Counter
C(DA)	Contents of Data Accumulator
CHAR	Character
CLBD	Clobbered
CLR	Clear
CONI	Condition In
CONO	Condition Out
CP0, CP1	Command Pulse 0, 1
DA	Data Accumulator
DASH	Data Accumulator Shift
DATAI	Data In
DATAO	Data Out
DB	Data Buffer
DC	Data Control
DEC	Digital Equipment Corporation
DEV	Device

GV	Give
IC	Control Status Register (IC is mnemonic for Initial Conditions)
I/O	Input/Output
IOB	Input/Output Bus
IOS	Input/Output Selection
IOT	Input/Output Transfer
IR	Instruction Register
LT	Left
MCV	Marginal Check Voltage
PA	Pulse Amplifier
PI	Priority Interrupt
PIA	Priority Assignment
PDP	Programmed Data Processor
RQ	Request
RT	Right
SCT	Shift Counter
STRB	Strobe
TK	Take

Pertinent Documents

The following publications serve as source material and complement the information in this manual.

Programmed Data Processor-6 Handbook, F-65 - contains programming information for the Data Control Type 136.

System Modules Catalog, C-100 - describes the functional operation of the elements of the various logical modules comprising the data control. It is suggested that this handbook be referenced when reviewing the data control logic diagrams in Appendix 2 of this manual.

PDP-6 Installation Manual, F-68 - gives environmental, power, and cabling requirements as well as other information for installation of the 136 Data Control in a PDP-6 system.

PDP-6 Maintenance Manual, F-67 - covers topics not repeated in this manual which are essential to efficient maintenance of the data control when used with the PDP-6.

GENERAL DESCRIPTION

The Data Control Type 136 is a programmable buffer unit through which the PDP-6 Processor can control up to six input-output devices (Figure 1-1). Of the six I/O devices, two are unidirectional 6-bit devices, two are bidirectional 6-bit devices, and two are variable character-length devices. The variable character-length devices can handle data words composed of three 12-bit characters, two 18-bit characters, or one n-bit character where n can be any integer from 1 to 36.

Purpose of Data Control System

Besides saving processor time by matching the format and timing requirements of the various asynchronous I/O devices to the PDP-6 processor, the data control also frees the processor from the necessity of exercising detailed program control over the micro-operations required for the various in-out transfers. Because most of the required sub-operations are built into the data control logic, the programmer need not be concerned with the detailed byte manipulations required by the various I/O devices. The CONO DATA CONTROL instruction need only specify the device to be used, the number of bits per character, and whether an output or input operation is to be performed.

Programmed Operations

The details of data control programming are described in Section 4; the present paragraph briefly lists the primary variables of data control operations which are controlled by a CONO DATA CONTROL instruction in the processor program.

Device Selection

A 3-bit field of the CONO-instruction effective address specifies to which of the six possible I/O devices a given output or input sequence is to apply.

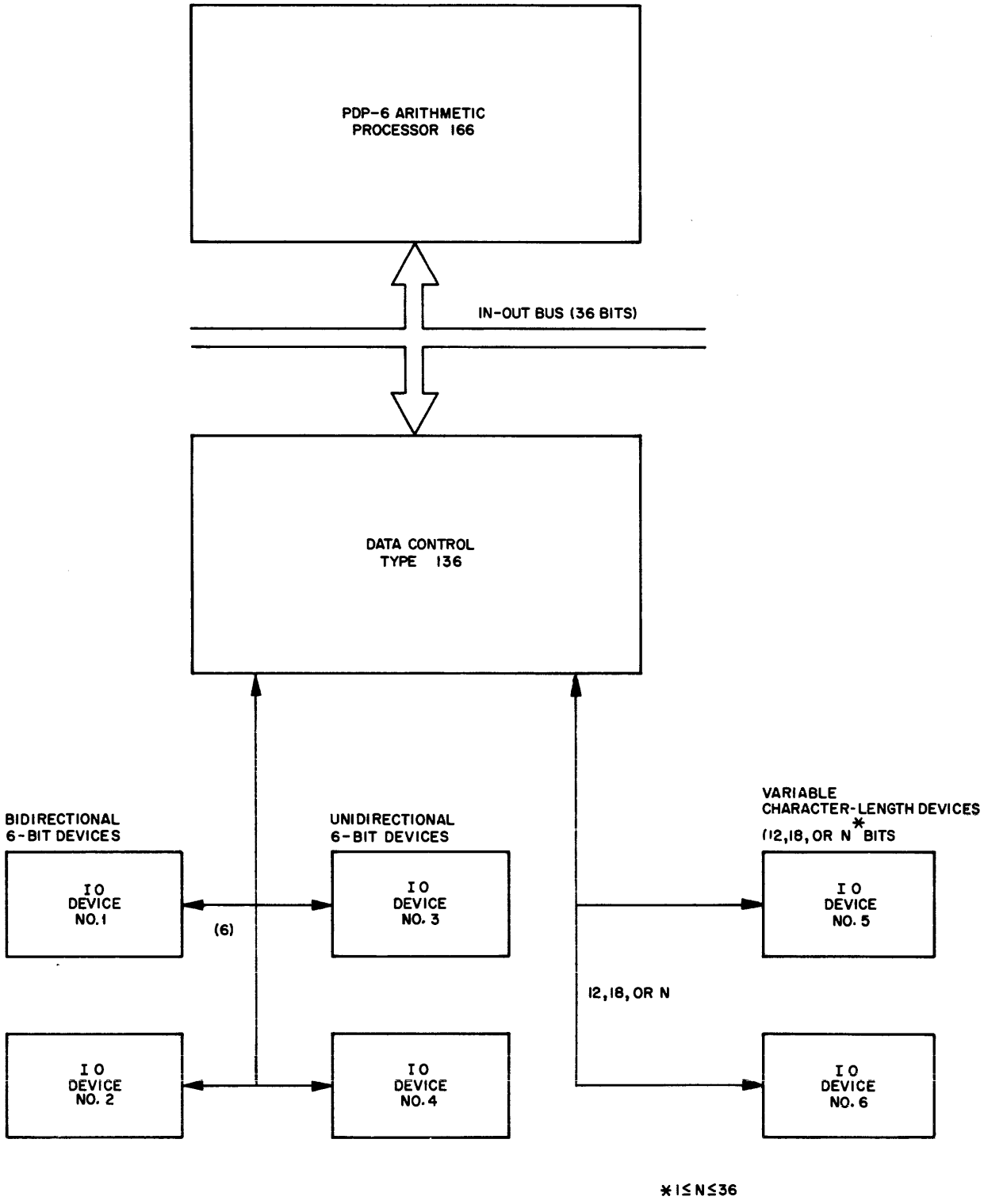


Figure 1-1 System Application

Input/Output

A single bit of the CONO-instruction effective address determines whether the data control is to treat the operations that follow as an output or input sequence.

Character Mode

A 2-bit field of the CONO-instruction effective address determines whether the data control is to transfer out (or read in) characters of 6, 12, 18, or n bits. Once the character mode has been specified, the built-in logic of the data control internally programs the requisite number of data shifts.

Priority Interrupt Assignment

A 3-bit field of the CONO-instruction effective address can assign a priority interrupt request channel to the data control. If such assignment is made, a PI channel can be chosen at any of seven ranked priority levels. This permits the data control to request a priority interrupt break with the Arithmetic Processor Type 166.

Logical Organization of Data Control

The data control contains a 16-bit control-status register called IC (Initial Conditions) and two 36-bit data registers, the data accumulator DA, and the data buffer DB (Figure 1-2). The internal logic of the data control governs the transfer of data through these registers and between them, the processor, and the I/O devices.

The data control internal logic is controlled by signals from two sources: the processor, and the IC register of the data control itself. Eight control inputs are sent to the data control from the processor (See Input Signals from Processor in Section 2). During CONI DATA CONTROL instructions, the processor can sample the contents of all 16 IC bits (see Figure 1-2).

Bidirectional Output and Input

The capability for both left and right data shifts is built into the data accumulator, DA. This enables the data control to operate with bidirectional input-output devices for example, DECtape (formerly Microtape) Control 551 and DECtape Transport 555. Because the direction of tape

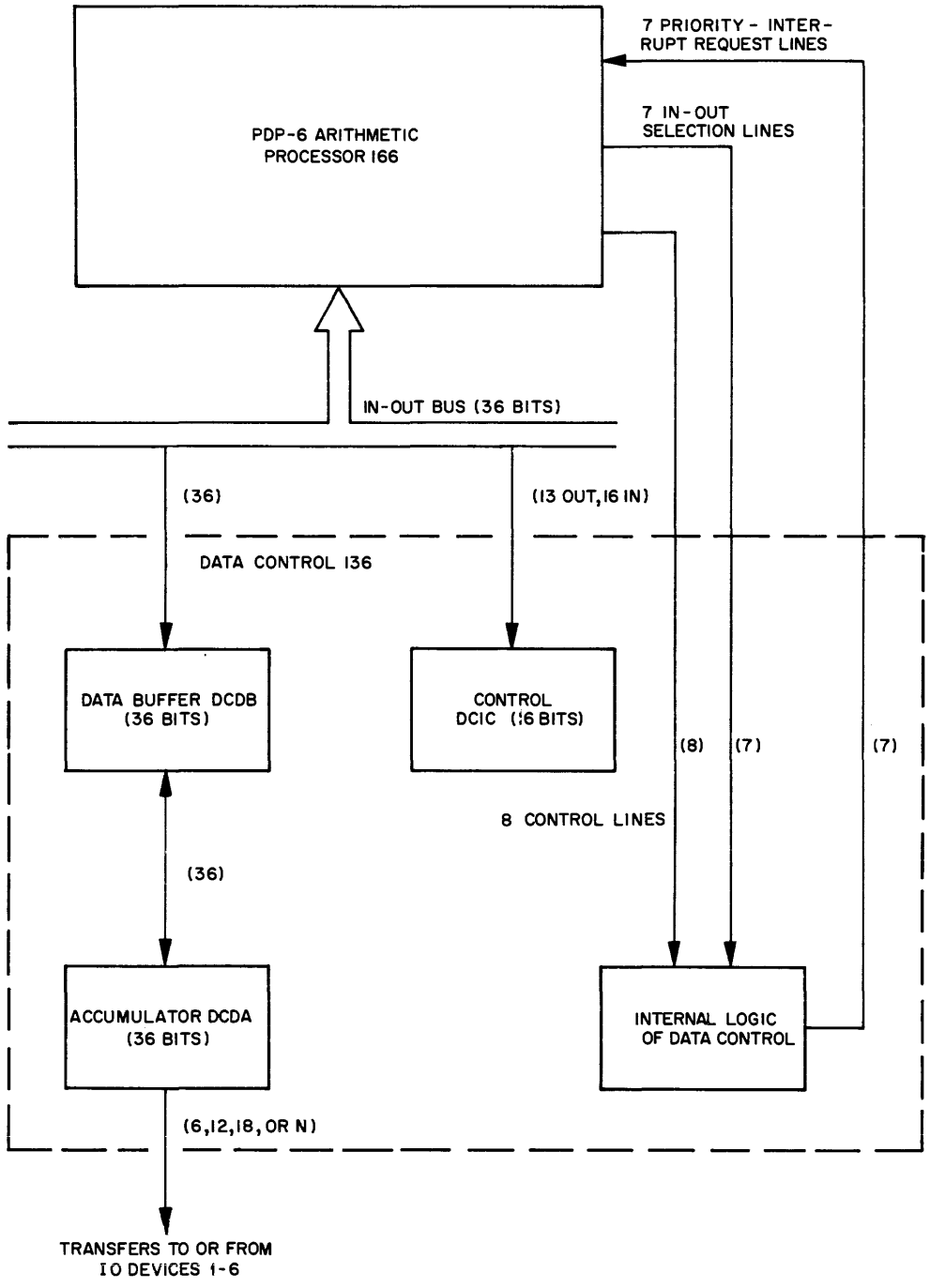


Figure 1-2 Logical Organization of Type 136 Data Control

motion determines the sequence of data bytes within a data word, the data control's bidirectional accumulator is a particularly useful feature. Depending on the current direction of tape motion, bytes can be shifted in or out from either end of the DA. The bidirectional accumulator permits correctly ordered data words to be read or written immediately while the DECtape moves in either direction. This frees the processor program from the necessity of reshuffling the bytes within a data word to remedy an inversion.

System Operating Specifications

APPLICATION	PDP-6 Input-Output Buffer
TIMING	Asynchronous
INPUT-OUTPUT DEVICES	
Number of Devices Controlled	1-6
Unidirectional 6-Bit Devices	0-2
Bidirectional 6-Bit Devices	0-2
Variable Character Length Devices	0-2
Character Modes Available (Variable Character Length Devices Only)	Three 12-bit characters per word Two 18-bit characters per word
Character Modes Available (cont'd)	One n-bit character per word $1 \leq n \leq 36$
BUFFER STORAGE	
Media	500 kc flip-flops
Quantity	Two 36-bit registers (one a bidirectional shift register)
STATUS-CONTROL REGISTER	16 bits, 13 programmable

MAXIMUM DATA RATE OF DATA CONTROL LOGIC

(Ordinarily not relevant. System speed normally limited by I/O device data rate.)

Character Mode	Minimum Interval Between Characters	Characters per Second
6-bit	3 μsec	333K
12-bit	4 μsec	250K
18-bit	5 μsec	200K
36-bit (n-bit)	3 μsec	333K

PROGRAMMABLE FUNCTIONS

Selection of I/O device from up to six devices
Selection of character mode from four available modes
Selection of input or output mode
Selection of priority interrupt request channel

ERROR BIT	One lost-data flag (described in Section 2)
PRIORITY INTERRUPT CHANNELS	7 ranked channels available. Any one, or none, may be used

Physical Characteristics

CONSTRUCTION	One standard all steel construction DEC cabinet, containing four logic racks
MODULES	Standard DEC system plug-in units, series 1000, 4000, and 6000
POWER EQUIPMENT	Power Supply unit 728; Power Control unit 811
LOGIC	Solid-state. Transistors and crystal diodes utilizing static logic levels (0 vdc and -3 vdc)
DIMENSIONS	
Height	69-1/2 inches
Width	22-1/4 inches
Depth	27-1/8 inches
CLEARANCE	
Front Doors	8-3/4 inches
Rear Door	14-7/8 inches
WEIGHT	73 pounds

Power Requirements

LINE VOLTAGE INPUT	105 to 125 volts, 60 cycle, single phase Power plug - Hubble Twist-Loc 3-prong, 30-ampere, 250-volt
CURRENT CONSUMPTION	5 amperes surge, 30 amperes normal
POWER CONSUMPTION	345 Watts
HEAT DISSIPATION	1175 BTU per hour

SECTION 2

THEORY OF OPERATION

Tables A1-1 through A1-6 are referenced throughout this and subsequent sections. These tables are contained in Appendix 1 on foldout pages.

Engineering drawings pertinent to the 136 Data Control are also referred to in this chapter. They are included in Appendix 2 and listed in the table of contents. The entire engineering drawing number is given with each reference so that either the reduced drawings in Appendix 2 of this manual or those that were supplied with the equipment may be consulted. Each block schematic number is followed by a comma and letters and/or numbers which refer to zones on the drawing. See Appendix 2 for an explanation on DEC logic symbols and codes.

PROCESSOR INTERFACE

These paragraphs describe the interchange of data and control information between the PDP-6 and the data control.

Cabling - Processor to Data Control

There are four 18-conductor cables from the PDP-6 processor to the I/O control units. Each cable is wired to two 22-terminal Methode plugs connected in parallel. The remaining four terminals, A, J, S, and Z, are used for grounding the shielding of the 18 conductors. Two of the four 18-conductor cables from the processor comprise the 36-bit I/O bus. These two cables are connected to Methode plugs 1A1-1B1 and 1A2-1B2 respectively. The 18 more significant data bits IOB0-17 enter the data control through Methode plugs 1A1-1B1. Methode plugs 1A2-1B2 receive the remaining 18 data bits IOB18-35 and also (via bus lines IOB20-35) the 16 IC control register bits IC20-35.

The remaining two 18-conductor cables between the processor and the data control are used for control information. One of these two cables is connected to Methode plugs 1A4-1B4. This cable carries the 14 IOS selection lines (described below under Device Selection Decoding) as well as the two ungated input signals, POWER ON and IOB RESET.

The second 18-conductor control cable enters through Methode plugs 1A3-1B3. This cable carries the six gated command signal inputs from the processor to the data control (IOB CONO SET, IOB CONO CLEAR, IOB DATAO SET, IOB DATAO CLEAR, IOB STATUS, and IOB DATAI; all described under Input Signals from Processor). The same cable also includes the seven priority interrupt request lines from the data control to the processor.

The specific terminal assignments of all four of the 18-conductor cables connected to Methode plugs 1A1-1A4 are listed in engineering drawing CD-D-166-0-IOMB in Appendix 2.

IC Register as Control Interface

The 16-bit control-status register, IC, is the principal interface for the exchange of control information between the processor and the data control. Most of the processor control information that governs the data control operation is transferred through the I/O bus and stored in the IC register. During the CONO instruction the processor determines the state of 13 of the 16 IC bits. The resulting state of IC determines the operations carried out by the data control logic during subsequent output or input instructions.

The IC register also serves to store control information moving in the opposite direction - from the data control to the processor. Since the processor samples the contents of all 16 IC bits with a CONI instruction, the state of DB RQ, status bit number 26, can readily be used to synchronize processor operations to appropriate phases of the data control output or input sequence. In a similar manner DA RQ, status bit number 25, synchronizes the operations of the I/O devices to the data control output or input sequence (see Table A1-1).

Device Selection Decoding

The device selection code portion of the IOT instructions determines which I/O device is to be utilized by a given instruction. The command levels produced by the IOT instruction must actuate only the selected I/O device, and must not be allowed to affect other non-selected devices. The gating function is performed by the device selection levels. These levels are applied to the data control and other I/O devices through the I/O selection lines.

The 14 I/O selection lines IOS3-9 represent both states of the seven bits of the device code contained in bits 3-9 of the processor instruction register, IR. (Assertion is at ground.) Seven

of the 14 IOS lines, one from each bit in IR3-9, are applied to an IOS decoder in the data control. The specific input connections made determine the selection code for a given data control unit. The decoder is shown in Appendix 2 in system block diagram SD-D-136-0-DCSD, D1, and in drawing BS-D-136-0-DC1, C-D1. The first data control incorporated in the PDP-6 installation is assigned the device selection code 200 octal.

Input Signals from Processor

Various input levels and pulses are applied to the data control by the processor. These include two ungated signals (not associated with any IOT instruction and applied to all I/O devices and control units), and six gated command signals applied only to the specific unit specified by the device selection code of the current IOT instruction.

Ungated Signals

There are two ungated input signals, the POWER ON level, and the IOB RESET pulse (see column 1, BS-D-136-0-DC1.

POWER ON - The POWER ON line drops to -15 vdc whenever computer power is turned on. This line is connected to the remote terminal of the LOCAL/REMOTE switch on the power control for each device. Normally, the LOCAL/REMOTE switches are left in the REMOTE position, so that the -15 vdc POWER ON level causes all peripheral equipment to be turned on when computer power is turned on.

IOB RESET - This line supplies negative pulses to clear the control registers and data buffers of all equipment attached to the I/O bus. The IOB RESET pulse is generated when computer power goes on or when the operator presses the I/O RESET key on the computer console.

Gated Command Signals

The six command lines are applied to the data control through Methode plugs 1A3-1B3 (drawing BS-D-136-0-DC1, column 1). These lines are connected to all control units, but are gated by the selection level from the IOS decoder so that they can affect only the control unit for the addressed device. The six command lines carry four output command pulses and two input command levels.

Output Command Pulses - There are two pairs of negative command pulses which implement the basic output instructions. Each of these two pairs includes a clear pulse followed by a set pulse.

The CONO instruction (see Section 4) produces an IOB CONO CLEAR pulse followed 1 μsec later by an IOB CONO SET pulse. The principal function of the CONO CLEAR is to clear the data control IC register and shift counter. The CONO SET then loads the 13 low order IC register bits from the I/O bus. The processor places information on the bus for 2.5 μsec. The clear occurs at the 1 μsec point; the set occurs at the 2 μsec point.

Similarly, the DATAO instruction (see Section 4) produces an IOB DATAO CLEAR pulse followed 1 μsec later by an IOB DATAO SET pulse. These pulses clear the data control data buffer, and then load it from the I/O bus. The processor again places information on the bus for 2.5 μsec, and again the clear is at the 1 μsec point and the set at the 2 μsec point.

Input Command Levels - There are two negative command levels which implement the basic input instructions. These command levels gate the input information that is held by the data control onto the I/O bus for 2.5 μsec. The processor reads the information from the bus onto its I/O register at the 2 μsec point. The IOB STATUS level gates the contents of the entire IC register onto the I/O bus during the execution of each CONI instruction. Similarly, IOB DATAI gates the contents of the data buffer onto the bus during DATAI instructions.

Priority Interrupt Request Lines

There are seven priority interrupt request lines running from data control to processor. The data control requests a priority interrupt by placing a ground on one of these seven PI request lines. To assign a priority to the data control, a CONO instruction loads a 3-bit number into the PIA (priority assignment) field of the IC register, IC33-35. (Refer to Priority Interrupt Assignment Bits IC33-35).

DATA BUFFER AND DATA ACCUMULATOR

The data control contains two 36-bit data registers, the data buffer register DB, and the data accumulator DA. Both the data buffer and the data accumulator are shown in engineering

drawing BS-D-136-0-DAB1. Each of the 12 4230 flip-flop modules shown in simplified block form is composed of logical elements identical to those illustrated in the detailed block schematic at column 8 of the figure. Each of the 12 4230 modules contains 3 bits of the data buffer DB, and 3 bits of the data accumulator DA. The 12 modules are arranged in 6 pairs of 2 modules each; each pair contains 6 bits of DB and 6 bits of DA. The 6 DA bits are connected to form a bidirectional 6-bit shift register.

Note that the bits contained in each module pair do not represent adjacent bit positions. Rather, each bit in a given module pair represents a bit position six bits less significant than the next bit to the left. Therefore, when the accumulator is operated as a conventional bidirectional shift register, each shift of one bit position to the left or to the right on the six module pairs causes the entire contents of the accumulator to be shifted six bits to the left or the right.

Data Buffer (DCDB)

The data buffer DB is a 36-bit storage register with no shift requirement. The various operations it performs in response to applied input signals are described below.

I/O Bus Transfers

Processor output data can be parallel transferred from the 36-bit I/O bus into DB. This transfer is produced by the application of a negative-going pulse, DC DB $\longleftarrow$ IOB, to the terminal Y ("LOAD BUFFER") inputs of the 12 4230 modules. For input, the 36-bit data word in the buffer can be read into the processor via the I/O bus. The data is placed on the bus by applying a negative level, DC DB $\longrightarrow$ IOB, to the terminal K ("BUFFER BUS") emitter-follower inputs.

Buffer-Accumulator Transfers

When the DC DB $\dashrightarrow$ DA pulse is applied to the W terminals, it jam transfers the contents of DB into the data accumulator. Note that the apparently converse pulse DC DA $\dashrightarrow$ DB (applied to the X and T terminals) not only jam transfers the contents of the accumulator into the buffer, but also clears the accumulator.

Buffer Clear

The buffer is cleared by DC CLR DB at terminal E.

Data Accumulator (DCDA)

The data accumulator DA is a 36-bit bidirectional shift register capable of successive 6-bit shifts in either direction. The various operations it performs in response to applied input signals are described below.

Accumulator Output

The accumulator contents are always available at output terminals J, N, and R of each module. Ground levels at these terminals correspond to ones in the corresponding bits of the accumulator. These accumulator output levels are applied to the output inverter buffers shown in engineering drawing BS-D-136-0-DAB2, A-B. During output operations, the I/O devices sample the inverter outputs at appropriate phases of the device operating cycles (negative levels at the inverter outputs correspond to accumulator bits containing ones).

Devices 3 and 4 receive only the six high order bits DA0-5. The bidirectional devices, I/O devices 1 and 2, receive the six high order bits during normal tape motion, but during reverse tape motion these devices receive instead the six low order bits DA30-35. The variable character-length devices, I/O devices 5 and 6, receive data in bytes of 12, 18, or n bits ($1 \leq n \leq 36$) (refer to Data Control Outputs to I/O Devices).

Accumulator-Buffer Transfers

The application of DC DA $\rightarrow\!\!\!\!\!\!\rightarrow$ DB to the X and T terminals of the 12 4230 modules causes the jam transfer of C(DA)* into DB and clears DA.

Accumulator Dash Data Inputs

Data can be read in from I/O devices 1-4 to the six low order (right-end) bits of the accumulator, DA30-35. In addition to this right-end readin, the bidirectional I/O devices, devices 1 and 2, can also read data into the six high order (left-end) bits of the accumulator, DA0-5.

*C(DA) means contents of data accumulator.

Data transfer from I/O devices 1-4 into the low order bits is carried out as follows. Data is presented to the six terminal S inputs shown in engineering drawing BS-D-136-0-DAB1, B. The data inputs are designated DASH DATA IN 30 RT through DASH DATA IN 35 RT. A ground DASH DATA IN level corresponds to a 1 input from the corresponding bit position of the selected I/O device. The detailed block schematic in column 8 of BS-D-136-0-DAB1 shows how the data is read into the accumulator applying a DASH LT pulse to the U terminals of all 12 4230 modules. This pulse strobes the input data from the I/O device into the six low order bits of the accumulator, and simultaneously shifts the previous contents of the entire accumulator six bit-positions left. (See page 2-5, second paragraph).

During reverse tape motion, the bidirectional devices, I/O devices 1 and 2, read data into the six high order bits of the accumulator, DA0-5. This operation is exactly similar to the low-order-bits transfer described above. The terminal H data inputs (BS-D-136-0-DAB1, C) are designated DASH DATA IN 0 LT through DASH DATA IN 5 LT. The data is strobed in by presenting a DASH RT pulse to the Z terminals of all 12 4230 modules. This pulse reads the input data from the I/O device into the six high order bits and shifts the accumulator right six bit-positions.

Accumulator Dash Strobe Inputs

The two variable-character-length devices, I/O devices 5 and 6, can read data into the accumulator in bytes of 12, 18, or n bits ($1 \leq n \leq 36$). These devices always send each input byte of less than 36 bits into the low order end of the accumulator.

The data transferred in from devices 5 and 6 does not enter the accumulator in the same way as the data from I/O devices 1 through 4. If the IC30-32 bits select I/O device 5 for a read-in sequence, that device sends the data control a TK CHAR 5 pulse (take a character from device 5) when it is ready to send data into the data control. This pulse (BS-D-136-0-DASH, C1) generates DASH STRB LT and DASH STRB RT pulses. Similarly, if IC30-32 select I/O device 6 for a readin sequence, the TK CHAR 6 pulse generates DASH STRB LT and DASH STRB RT pulses.

Input data levels from device 5 are applied to the negative capacitor-diode gates shown in drawing BS-D-136-0-DAB2, 7C; negative levels correspond to 1 bits. The number of input

terminals used, 12, 18 or n, depends on the size of the input bytes that device 5 sends to the data control. For bytes of less than 36 bits the active input terminals are always those at the low order end of the accumulator.

The input data levels are strobed into the accumulator by device 5 DASH STRB pulses. If a negative input level is applied to a given capacitor-diode gate, the pulse inverter output of that gate generates a positive pulse at DASH STRB time, thereby setting the flip-flop to the 1 state. Input levels from device 6 are strobed into the accumulator by the same process.

Accumulator Shifts

The entire contents of the accumulator are shifted left six bit-positions by applying a DASH LT pulse to the U terminals of all 12 4230 modules. Application of a DASH RT pulse to the Z terminals produces a 6-bit right shift.

For I/O devices 1-4 the contents of the accumulator are always shifted left six bit-positions whenever a character is read into the six low order bits of the accumulator. This is necessary to prevent the destruction of the data previously read in. When reading in data from the bi-directional devices in reverse tape motion, C(DA) (the contents of DA) are shifted right six bit-positions as each new character is read in.

Devices 5 and 6, the variable character-length devices, are read in at the DASH STRB pulse but do not start to shift C(DA) until 1 μsec after DASH STRB. The accumulator is then shifted left six bit-positions every microsecond until there is room for the next byte of input data to be read in. For 12-bit bytes, two shifts are required; 18-bit bytes take three shifts; n-bit bytes are not shifted at all. For operations involving devices 5 or 6, no shifts ever follow the final character of the data word.

For output operations of all six I/O devices, the direction and number of accumulator shifts is the same as for input. Data is normally transferred out to the I/O devices from the high order (left-end) accumulator bits, and C(DA) are shifted left for each character that is transferred out. In reverse tape motion, the bidirectional devices 1 and 2 transfer out data from the low order (right-end) accumulator bits, and shift right as each new character is transferred out.

CONTROL STATUS REGISTER (DCIC)

The internal logical operations of the data control are controlled largely by the 16-bit control-status register IC (mnemonic for Initial Conditions) and the 2-bit shift counter SCT. The IC register is composed of 16 identical 500-kc flip-flops. However, the conditions controlling these flip-flops and the uses to which they are applied are so diverse that each of the nine data fields (which together comprise the 16 IC bits) may be regarded as an essentially separate register. Thirteen of the 16 IC bits, IC23-35, are shown on engineering drawing BS-D-136-0-DC2, A-B. The remaining three IC bits, IC20-22, make up the 3-bit character counter shown in drawing BS-D-136-0-DC1, B6-7. The nine data fields making up the IC register are listed in Table A1-1, a foldout sheet in Appendix 1.

The 2-bit shift counter SCT0-1 is a separate auxiliary register, entirely internal to the data control. Because it can neither be controlled nor sampled by the processor, it is not considered to be part of the IC register. However, because the logical conditions governing the shift counter operations are closely interrelated with the conditions governing the operation of the character counter, the two counters are described together below. The shift counter is shown in BS-D-136-0-DC1, B5.

Character Counter IC20-22 and Shift Counter

The character counter (CCT 0-2) and the shift counter (SCT 0-1) are shown in drawing BS-D-136-0-DC1, B. Decoder nets which determine when the counters have reached the assigned count (DC CCT DONE and DC SCT DONE) are shown immediately above the counters. The control nets which govern the counting and clearing of the counters are also shown in the same drawing at zone C7-8.

Function

The character counter keeps track of the number of characters of a given data word which have been read in or transferred out. It completes its count at a number one less than the number of characters in the data word. The resulting DC CCT DONE (Character Counter Done) level serves two functions. First, during operations by I/O devices 5 or 6, it prevents the final

character from being shifted left. Second, and even more important, it is the character counter done level which enables the final command pulse of the readin or transfer out sequence to terminate the sequence by setting the DA RQ flip-flop.

The shift counter is only used by the variable character-length devices. When devices 5 or 6 are operating in the 12 or 18-bit mode, the shift counter keeps track of the number of 6-bit left shifts which have been made after the readin or transfer out of a given character. The DC SCT CONT level (which is asserted until the shift counter completes its assigned count) controls the generation of the repetitive DASH LT pulses that shift all non-final 12 or 18-bit characters left to make room for the next character.

CP1 Pulses

The CP1 pulse strobes all stepping and clearing of both counters. The generation of this pulse is described later in this section. A CP1 pulse is generated whenever any I/O device sends a command pulse to the data control. A CP1 pulse is also generated by the DASH LT pulse whenever the accumulator contents are shifted left.

Counter Done Levels

The character mode in which the data control operates (see Character Mode Bits IC28-29) determines the correct final count for both the shift counter and the character counter. The final count of the shift counter corresponds to the number of shifts required per character (after the command pulse that reads the character in or transfers it out). The final count of the character counter is always one less than the number of characters in the data word. The shift counter done level DC SCT DONE is generated by the decoders in BS-D-136-0-DC1, A-B5; the decoders in zone A-B7 of this drawing generate the character counter done level DC CCT DONE. Operation of these two sets of decoders is summarized in Table A1-2.

Shift Counter Continue

The shift counter is stepped and a DASH LT pulse is generated at each CP1 pulse which occurs while the DC SCT CONT (Shift Counter Continue) level is asserted negative. The level rises to ground if either the shift counter or character counter finishes its assigned count, that is, at the assertion of either SCT DONE or CCT DONE (see BS-D-136-0-DC1, C8).

Character Counter Continue

The character counter is stepped at each CP1 pulse which occurs while the DC CCT CONT (Character Counter Continue) level is asserted negative. The level is asserted only when the shift counter is done (BS-D-136-0-DC1, C8). This insures that the character counter is stepped only by those CP1 pulses corresponding to actual command pulses from the I/O devices, and not by the CP1 pulses resulting from repetitive DASH LT pulses in 12 or 18-bit operation. The latter class of CP1 pulses are counted by the shift register only; they represent merely shifts within a given character and not new characters. When the DC CCT DONE level is asserted, the DC CCT CONT level ends.

Clear Counter Levels

The circuits which clear the shift counter and character counter are also shown in BS-D-136-0-DC1, C7. Both shift counter and character counter are cleared by the initial data control clear pulse, DC IC CLR. The character counter is also cleared by the first CP1 pulse which occurs after the character counter completes its count. There is one further possibility of clearing the character counter. Under certain circumstances it may be desirable to clear the character counter by an externally generated clear pulse from one of the I/O devices. The negative capacitor-diode gate permits the option of connecting any of I/O devices 1-4 for an external character counter clear.

The shift counter is always cleared at the first CP1 pulse occurring after either the shift counter or the character counter is done.

Error Bit IC23

The DATA CLBD (Data Clobbered) error bit is set to 1 by any CP0 pulse generated when DA RQ contains 1. This error condition indicates either that an output command pulse has been sent to the data control while DA is empty, or else that an input command pulse has been sent to the data control while DA is still loaded with the previous data word.

The latter condition is the more serious one; its detection is the primary purpose of the data CLBD error bit. If an I/O device should attempt to read in data before the previous data word

were advanced from the accumulator to the data buffer, data would be lost. This error condition could arise only through an equipment malfunction or a programming error at the I/O device.

Data Control Status Bits IC24-26

The three status bit flip-flops DBDA MOVE, DA RQ, and DB RQ synchronize the processor and I/O device operations to the data control as well as governing the internal operations of the data control itself.

DBDA MOVE IC24

The DBDA MOVE (data buffer data accumulator move) flip-flop is shown in BS-D-136-0-DC2, A2. Its output is applied to input terminal M of diode gate 1B7 (BS-D-136-0-DC1, C3); a negative-to-ground transition at the output of that gate initiates all transfers (in either direction) between the data buffer and the data accumulator. A transfer can be initiated in either of two ways. When the DA RQ flip-flop is in the 1 state, the 0 to 1 transition of DBDA MOVE causes a rising edge at the output of gate 1B7, thus triggering pulse amplifier 1A9 and producing a DA $\rightarrow$ DB or DB $\rightarrow$ DA transfer. Similarly, when the DBDA MOVE flip-flop is in the 1 state, the 0 to 1 transition of the DA RQ flip-flop causes the transfer.

DA RQ Flip-flop IC25

The data accumulator request flip-flop DA RQ (BS-D-136-0-DC1, A2) indicates the status of the accumulator both to the I/O devices and to the internal logic of the data control. At most phases of the data control operating cycle, the state of DA RQ indicates whether or not the data accumulator is fully loaded with data. During output operations, a 0 in the DA RQ indicates that the accumulator is loaded. During input operations, however, a 1 in DA RQ indicates that the accumulator is loaded, and a 0 indicates that the accumulator is empty and free to receive data.

For either output or input operations, a 0 in DA RQ indicates that the data control is ready to act on commands from the I/O devices (to transfer out the data word in DA during

output, or to read in a new data word to the empty DA during input). Conversely, during both output and input operations, a 1 in DA RQ indicates that the accumulator is not ready to execute a transfer.out or readin sequence (refer to Table A1-3).

DB RQ Flip-flop IC26

The data buffer request flip-flop DB RQ (BS-D-136-0-DC1, A3) indicates the status of the data buffer both to the I/O devices and to the internal logic of the data control. At most phases of the data control operating cycle, the state of DB RQ indicates whether or not the data buffer DB is loaded with data. The immediate meaning of the state of DB RQ is the same as that of DA RQ described above. Either flip-flop in the 0 state during output, or in the 1 state during input, signifies that the corresponding register is loaded with data. When DB RQ contains a 1 during output (indicating that the buffer is empty), the data control is ready to accept output data from the processor. Moreover, when DB RQ contains a 1 during input (indicating that the buffer is loaded) the data control is ready to read data into the processor.

Status-Bit Configuration Tables (Tables A1-4 and A1-5)

The configurations of the three status-bit flip-flops after each phase of the data control output operations are summarized in Table A1-4. Input configurations are similarly summarized in Table A1-5. These two tables are frequently referred to in the paragraphs below describing the data control output and input cycles. Once the data control logic has been learned, these two tables and the system flow diagram, FD-D-136-0-FD1, can serve as a ready-reference summary of the data control operating cycle.

Input/Output Flip-flop IC27

The input/output flip-flop, DC IN/OUT, is shown in engineering drawing BS-D-136-0-DC2, A5. It indicates to the data control logic whether the current operation is an input operation or an output operation. The outputs of the in/out flip-flop are applied to input terminals Z and T of pulse amplifiers 1A10 (BS-D-136-0-DC1, C5). The state of the flip-flop determines whether the dual purpose transfer pulse DC DB $\leftarrow\!\!\!\!\rightarrow$ DA produces a DC DA $\longrightarrow$ DB pulse which jam transfers C(DA) into DB during input cycles, or a DC DB $\longrightarrow$ DA pulse, which jam transfers C(DB) into DA during output cycles.

Character Mode Bits IC28-29

The two character-mode flip-flops CH MODE 0 and CH MODE 1 are shown in BS-D-136-0-DC2, B4. The states of these flip-flops determine the character mode in which the data control operates (refer to Table A1-2). The outputs from the character mode flip-flops are applied to the decoder nets shown in BS-D-136-0-DC1. Thus they determine the final contents of the shift counter that produce the DC SCT DONE level, and the final contents of the character counter that produce the DC CCT DONE level.

Internal Device Address IC30-32

The three internal device address flip-flops, DC DEVICE 0, DC DEVICE 1, and DC DEVICE 2, are shown in BS-D-136-0-DC2, 5-6. The 3-bit binary number contained in these three flip-flops indicates to the data control logic which of the six subsidiary devices is to be used for a given input or output operation.

The contents of the three internal device address flip-flops correspond directly to the I/O device numbers. When the flip-flops contain 010 (2 octal) I/O device 2 is selected; 100 (4 octal) selects I/O device 4, and so on. This 3-bit internal device address addresses an I/O device only from the selected data control. It should not be confused with the 7-bit device selection code used by the processor to address the selected I/O device directly, nor with the second 7-bit device selection code used by the processor to address the selected data control; levels 1-4 are first sent through the buffer-inverter circuits shown in BS-D-136-0-DC2, A5-6.

Priority Interrupt Assignment Bits IC33-35

The three priority interrupt assignment flip-flops, PIA-0, PIA-1, and PIA-2, are shown in BS-D-136-0-DC2, B7-8. The processor can assign a priority-interrupt request channel to the data control by means of a CONO instruction which sets these three bits to some number from 1 to 7. If all three PIA bits are left 0, no priority interrupt channel is assigned (see Section 4).

Until the contents of PIA0-2 are changed by another CONO from the processor, the data control can request a break on the assigned channel. The data control requests a break on the assigned channel when the DB RQ flip-flop is set to the 1 state. If this occurs during

an output sequence, it indicates that the DB register is empty and is therefore ready to receive more data from the processor. If it occurs during an input sequence, it indicates that the DB register is loaded with data and is therefore ready to read data into the processor. Both output levels from each of the three PIA flip-flops are applied to the binary-to-octal decoder 1B8 (BS-D-136-0-DC2, B7-8). When inactive, the seven PIA outputs from this decoder are negative levels. When the DB RQ flip-flop is set to 1, the PIA line corresponding to the assigned channel is grounded.

DEVICE INTERFACE

This section describes the interchange of data and control information between the data control and its subsidiary I/O devices.

Cabling - Data Control to I/O Devices

There are 16 22-conductor coaxial cables from the data control to the six I/O devices. Devices 1 and 2, the bidirectional devices, require two cables each. Unidirectional devices 3 and 4 have only one cable apiece. Devices 5 and 6, the variable character-length devices, are each assigned five cables. The 16 cables are wired to 22-pin Methode plugs in rack positions 1D10 through 1D25. The specific terminal assignments of all 16 cables are shown in engineering drawing ID-D-136-0-EDC.

Data Control Outputs to I/O Devices

The data control sends the I/O devices both control information and data levels.

Control Information

Output control information is sent to I/O Devices 1 through 6 through the six Methode plugs shown in BS-D-136-0-DASH, D. All six devices receive the state of the DA RQ flip-flop; furthermore each device receives its own device select signal. The negative 1 output of the DA RQ flip-flop is amplified by the 6684 Bus Driver 1B21 (BS-D-136-0-DC2, A3). The amplified output is then sent to the six I/O devices through the W terminals of the six Methode plugs (BS-D-136-0-DASH, D). When DA RQ contains a 1, the output is quiescent in the assertion state shown (W terminals negative). This indicates to the selected I/O device that

the data control is not ready. When the DA RQ output is active, the W terminals are at ground. This indicates to the I/O devices that the DA RQ flip-flop contains 0 and that the data control is ready.

The data control sends DC DEVICE SEL levels to the six I/O devices so that a given device responds to data control operations only when the internal device address in IC30-32 selects that device; refer to Internal Device Address IC30-32. The negatively asserted DC DEVICE SELECT level for each device is applied to the X terminal of the corresponding Methode plug (BS-D-136-0-DASH, D).

Data-Level Outputs

The ground assertion 1 outputs of the 36 accumulator flip-flop are applied to the output buffer-inverters shown in BS-D-136-0-DAB2, A-B. During output operations, the I/O devices sample the inverter outputs at appropriate phases of the device operating cycles. Negative levels at the inverter outputs correspond to accumulator bits containing ones. The outputs from the six high order accumulator bits, DA0-5, are applied to all six I/O devices. The output data levels are sampled only by the selected device. If bidirectional I/O device 1 or 2 is selected, the high order (left-end) accumulator bits are sampled only during normal tape motion. During reverse tape motion output cycles, the bidirectional devices 1 and 2 receive the six low order bits CA30-35. All 36 outputs are available for connection to the variable character-length devices 5 and 6. Data may be transferred out to these two devices in bytes of 12, 18, or n bits. For 12 or 18-bit characters I/O devices 5 and 6 always sample data from the high order end of the accumulator. For n-bit ($n < 36$) single-character data words, devices 5 and 6 sample from the low order end of the DA.

Inputs to Data Control from I/O Devices

Besides providing input levels that read data into the data control, the I/O devices also send command pulses to the data control. During output cycles, these pulses indicate to the data control when the selected I/O device is ready to accept another data character. During input cycles, command pulses from the I/O device let the data control know when a new data

character is ready to be read in. The command pulse inputs from the I/O devices govern the generation by the data control of internal command pulses, which in turn cause the data control to sample and shift at appropriate intervals.

Command Pulses from I/O Devices

Command pulse control information to the data control is not the same for all six I/O devices; the specific command pulses sent to the data control depend upon which I/O device is selected.

Bidirectional Devices 1 and 2 - The bidirectional devices send the same command pulses to the data control during both the output cycle and the input cycle. However, the command pulses from these devices do vary depending upon whether the device is functioning in normal tape motion. For both output and input cycles in normal tape motion, a TK/GV CHAR LT (Take or Give a Character Left) command pulse is sent to the data control.

During an output cycle, this pulse indicates to the data control that the I/O device has sampled the output levels from the six high order accumulator bits, DA0-5, and that the data control may shift the accumulator contents left six bits, thus bringing the next character of the data word into position to be sampled by the device. During an input cycle, the TK/GV CHAR LT command pulse indicates that the I/O device has applied a 6-bit input character to the low order accumulator DASH DATA IN RT inputs (described under Data-Level Inputs from I/O Devices below) and that the data control may shift the contents of the accumulator left six bit-positions, and at the same time strobe the input character into the six low-order bit-positions which are made available by the shift.

For both output and input cycles in reverse tape motion, the bidirectional devices send a TK/GV CHAR RT (Take or Give a Character Right) command pulse to the data control.

Unidirectional Devices 3 and 4 - For both output and input cycles, devices 3 and 4 send a TK/GV CHAR LT command pulse to the data control. This pulse serves exactly the same function for unidirectional devices 3 and 4 as it does the bidirectional devices 1 and 2 during normal tape motion.

Variable Character-Length Devices 5 and 6 - The variable character-length devices send the data control a GV CHAR command pulse during the output cycle, but send a TK CHAR command pulse during the input cycle. The GV CHAR command pulse indicates to the data control that the device has accepted the character, and that unless that character was the last in the output data word, the data control must shift the accumulator left by one character length in order to bring the next character into the high order bit positions.

During input operations, devices 5 and 6 send the data control a TK CHAR command pulse when they have presented an input character to the low order strobe data inputs of the accumulator (refer to the following paragraph). This indicates to the data control that it may strobe the character into the low order accumulator bits and then begin to shift the accumulator contents left to make room for the next character that will be read in (unless the character was the last one in the input data word).

Data-Level Inputs from I/O Devices

The paths by which the I/O devices transmit data into the data control also vary depending upon which device is selected for input and (for the bidirectional devices 1 and 2) depending upon which direction of tape motion the device is using.

Dash Data In Right - Devices 1 through 4 read information into the low order (right-end accumulator bits through the DASH DATA IN RT networks shown in BS-D-136-0-DASH, B4-8. These input networks are used by bidirectional devices 1 and 2 only during normal tape motion. The I/O devices apply six input data levels (called DATA IN RT levels) to the Methode plugs shown in BS-D-136-0-DASH, C4-8. Negative levels correspond to 1 bits.

If one of the DC DEVICE SELECT levels 1-4 is asserted negative, data from the corresponding device is gated into the six diode gates 1D2-1D4, and appears at the output terminals of these gates as the six data levels DASH DATA IN 30 RT through DASH DATA IN 35 RT. Since the gates invert the data levels, ground DASH DATA IN RT levels correspond to 1 bits from the devices. The DASH DATA IN RT outputs from the diode gates are strobed into the six low order (right-end) accumulator bits DA30-35 by the same DASH LT pulse that shifts the accumulator contents left. The accumulator readin is described earlier in this section under Accumulator Dash Data Inputs.

Dash Data In Left - During reverse tape motion, bidirectional devices 1 and 2 read information into the high order (left-end) accumulator bits through the DASH DATA IN LT networks shown in BS-D-136-0-DASH, A5-8. The I/O devices apply six DATA IN LT input levels to the Methode plugs 1D10 and 1D12. Negative levels at the gate inputs correspond to 1 bits read in from the I/O devices; the gate outputs are inverted and thus generate ground DASH DATA IN LT levels to represent 1 bits from the devices.

Variable Character-Length Data - The variable character-length devices 5 and 6 send data into the data control through the negative capacitor diode gates shown in BS-D-136-0-DAB2, C-D. The number of active inputs depends on the length of the input data byte; for bytes of less than 36 bits, the data is always sent into the low order bits of the accumulator. Negative inputs represent 1 bits; inactive inputs are not connected. Data from I/O device 5 is gated into the accumulator by the DASH STRB LT (DEV 5) and DASH STRB RT (DEV 5) pulses; data from I/O device 6 is gated into the accumulator by the (DEV 6) DASH STRB pulses.

Resulting Internal Command Pulses

Three sets of internally generated command pulses are produced by the data control in response to the incoming command pulses from the I/O devices. These pulses are: the dash strobe pulses; the dash left and dash right pulses; and the CP0 and CP1 pulses. These pulses strobe data into the accumulator, initiate the accumulator shifts, and control the shift and character counters that indicate to the data control when it has become to the end of a character or data word.

Dash Strobe Pulses

The dash strobe pulses, DASH STRB (DEV 5) and DASH STRB (DEV 6) gate in the variable character-length input data for I/O devices 5 and 6. The pulse amplifiers which generate the DASH STRB (DEV 5) pulse and those which generate the DASH STRB (DEV 6) pulse are shown in BS-D-136-0-DASH, D1. Each of these pulses is generated by two pulse amplifiers and applied to two independent lines, DASH STRB LT (which strobes data through capacitor-diode gates into the 18 high order accumulator bits) and DASH STRB RT (for the 18 low order bits of DA). For either device, the DASH STRB RT and DASH STRB LT lines are always pulsed together .

Dash Left and Dash Right Pulses

The DASH LT pulse always causes the accumulator contents to be shifted left six bit-positions. Furthermore, during input cycles of I/O devices 1-4, DASH LT strobes the input data carried by the six levels DASH DATA IN 30 RT through DASH DATA IN 35 RT into the six low order (right-end) bit-positions of the accumulator (the six bit-positions made available by the left shift).

The DASH RT pulse always causes the accumulator contents to be shifted right six bit-positions, and during input cycles strobes the DASH DATA IN 0 LT through DASH DATA IN 5 LT levels from bidirectional I/O device 1 or 2 into the six high order (left-end) accumulator bit-positions.

Conditions When Dash Right and Dash Left Are Generated (I/O Devices 1-4) - The pulse amplifiers that generate the DASH LT and DASH RT pulses are shown in BS-D-136-0-DASH, A2-3. Pulse amplifier 1A21 generates DASH RT whenever I/O device 1 or 2 is selected and takes or gives a character in reverse tape motion.

Pulse amplifier 1A18 generates the DASH LT pulse under two sets of conditions; one set of conditions applies to I/O devices 1-4, the other applies to I/O devices 5 and 6. The first set of DASH LT conditions, which is implemented by the negative capacitor diode gate at input terminal F of the pulse amplifier, is almost exactly similar to the DASH RT conditions. This set of conditions causes a DASH LT pulse to be generated when a character has been transferred out to or read in from selected I/O device 1-4. (If bidirectional device 1 or 2 is selected, the device must be in normal tape motion to produce a DASH LT.)

Condititons When Dash Left is Generated (I/O Devices 5 and 6) - The second set of input conditions which causes the generation of a DASH LT pulse is less self-evident. The variable character-length devices 5 and 6 can read in or accept data in bytes of 12, 18, or n bits. The n-bit bytes are single-character data words which are not shifted, and thus require no DASH LT pulses. However, each 12-bit character which is read in or transferred out (except the final character in the data word) must be followed by a 12-bit left shift of the accumulator contents. Each non-final 18-bit character is followed by an 18-bit left shift. To implement

these left accumulator shifts, it is necessary, when operating in 12-bit mode, to generate two DASH LT pulses for each character except the last. When operating in 18-bit mode, three DASH LT pulses are needed. Each DASH LT pulse produces a 6-bit left shift.

Circuits Used to Produce Dash Left Pulse - There are four sets of circuits involved in producing the correct dash left shift operations: 1) Pulse amplifier 1A18 is the final circuit which generates the required DASH LT output pulses; 2) the shift counter keeps track of the number of shifts executed; 3) the character counter indicates when the final character of the data word is being transferred out or read in; 4) finally the CP0-CP1 (Command Pulse 0 - Command Pulse 1) network produces the CP1 pulses which initiate the DASH LT pulses and step the shift counter.

The conditions for generating DASH LT during output and input cycles of I/O devices 1-4 are implemented by the capacitor-diode gate at input F of pulse amplifier 1A18 (BS-D-136-0-DASH, A2). Terminals K and L of this same pulse amplifier receive the inputs which govern the generation of DASH LT during output and input cycles by I/O devices 5 and 6. Pulse amplifier 1A18 generates a DASH LT pulse at the arrival of each CP1 pulse at terminal K so long as the DC SCT CONT (Shift Counter Continue) level is asserted at terminal L.

As soon as either the shift counter or the character counter completes its assigned count (indicating that the contents of the accumulator have been sufficiently shifted) the SCT CONT level rises to ground. This prevents any further CP1 pulses from generating a DASH LT, and also resets the shift counter. For a more detailed description of the shift counter operations and logic levels refer to Character Counter IC20-22 and Shift Counter.

DASH CP0 and CP1 Pulses

The DASH CP0 and CP1 pulses (Command Pulse 0 and Command Pulse 1) are generated by the network shown in BS-D-136-0-DASH, B2-3. The CP1 pulse is simply a CP0 pulse delayed 1 μsec by 1310 Delay Line 1A22. The CP1 pulse is the signal that steps the shift counter and character counter and that initiates the appropriate number of DASH LT signals during device 5 and 6 operations with 12- and 18-bit characters. The CP0 pulse is used only for the single purpose of setting the DATA CLBD error flip-flop.

Generated by Device Command Pulses - A CP0 pulse and a CP1 pulse are generated whenever any I/O device sends a command pulse to the data control. Any command pulse from I/O devices 1-4 must produce either a DASH LT or a DASH RT; either of these pulses triggers CP0 and CP1. The variable character-length device command pulses also initiate CP0 and CP1. When a character is transferred out to device 5 or 6, the GV CHAR 5 or GV CHAR 6 pulse directly produces CP0 and CP1. When a character is read in from I/O device 5 or 6, the TK CHAR 5 or TK CHAR 6 pulse first generates a corresponding DASH STRB (DEV 5) or DASH STRB (DEV6) pulse which then triggers pulse amplifier 1A18 to produce CP0 and CP1.

Repetitive Shifts - In addition to following every command pulse from the six I/O devices, CP0 and CP1 may also be produced by DASH LT pulses which are not themselves initiated by command pulses. As mentioned previously, each DASH LT pulse produces a CP1 1 μsec later. Note, however, that the DASH LT pulse itself can in turn be produced by CP1 so long as the DC SCT CONT level is asserted negative (BS-D-136-0-DASH, A2).

The result of this reciprocal causal relationship is a loop. The external command pulse from the I/O device which occurs when a character is first transferred out or read in (for example, GV CHAR 5) causes the generation of the first CP1 pulse for that character. This first CP1 pulse in turn produces a DASH LT pulse (the first DASH LT of the current character) and also constitutes the first shift counter step signal of the current character; CP1 steps the SCT to 01. The first DASH LT pulse initiates a second CP1 (after a delay of 1 μsec). This second CP1 pulse steps the shift counter to 10 (2 octal) and at the same time produces a second DASH LT pulse.

If 12-bit characters are being transferred out to the I/O device, the shift counter is done after being stepped by the second CP1. The second DASH LT can then produce a third CP1 (delayed 1 μsec), but that third CP1 cannot get through to trigger pulse amplifier 1A18. (It is no longer gated through by the DC SCT CONT level. That level ceased to be asserted when the shift counter reached its assigned count, 2 octal, at the second CP1 pulse.) Consequently, no third DASH LT pulse is ever generated. The third CP1 does not step the shift counter, but instead resets it. If 18-bit characters are being transferred out to the I/O device, or being read in from it, the process is similar, except that the shift counter is not done until it reaches a count of 11 (3 octal).

Final Characters Not Shifted for I/O Devices 5 and 6 - During input cycles involving devices 5 and 6, the last character of the data word (the first and only character in n-bit mode, the second character in 18-bit mode, or the third character in 12-bit mode) must not be shifted. If it were shifted, data would be lost out the left end of the accumulator (or for n-bit input cycles at least displaced). During output cycles involving devices 5 and 6, it would of course do no harm to shift the empty accumulator after the last character is transferred out, but this would be useless and is not done.

The logic which implements the data control shift operations prevents any shift pulses from being generated by the readin or transfer out of the final 12, 18, or n-bit character. The command pulse from I/O device 5 or 6 concurrent with the transfer out or readin of the final character does produce a CP0 and CP1 pulse. However, the CP1 is prevented from triggering a DASH LT by the removal of the DC SCT CONT level when the character counter reaches its final assigned count.

The final count of the character counter is always one less than the number of characters in the data word; refer to Table A1-2. The character counter is done at the count of zero for n-bit characters, at the count of one for 18-bit characters, and at the count of two for 12-bit characters. When it reaches its final count in 12- or 18-bit mode, it produces a DC CCT DONE (Character Counter Done) level, thereby terminating the negative DC SCT CONT level and preventing the generation of further DASH LT pulses. In n-bit mode, there is only one character to a data word. Both the character counter and the shift counter are done at the initial zero count. Consequently, no DASH LT pulses are generated.

During output or input cycles involving I/O devices 1-4, completion of the character counter count does not prevent generation of a DASH LT or DASH RT pulse. Such a shift pulse is generated concurrently with the final (sixth) character of the output or input data word. The sixth and final shift pulse is harmless for output cycles, and necessary during input cycles.

OUTPUT CYCLE

The simplified output cycle described in the present section consists of a CONO (Conditions Out) instruction followed by a series of DATAO instructions. Although for practical operations

other routines are employed (refer to Section 4 for a description of sample programs), the input signals to the data control, and the resulting internal cycle, are identical in both cases; the only difference lies in the intervening instructions executed by the processor.

CONO Out Instruction

The CONO Out (Conditions Out) instruction clears the IC register and the accumulator (preserving the contents of the accumulator in DB), and then sets the contents of the IC register to control codes such as those shown in Table A1-6. This prepares the data control for the following DATAO instructions, enabling it to dispatch the data out in appropriate bytes to a selected I/O device, and if required, enabling it to generate a priority interrupt request when more data is needed from the processor. Also see Table A1-7.

CONO Clear Pulse

The processor initiates the data control output sequence by sending the data control on IOB CONO CLR pulse (I/O Bus Conditions Out Clear). This pulse is represented by the initial operation (CONO CLR) of the output flow chart, engineering drawing FD-D-136-0-FD1. It occurs 1 μsec before time 0 of the output timing sequence shown in TD-D-136-0-TD1. The data control receives IOB CONO CLR over one of the I/O bus command lines (BS-D-136-0-DC1, A1); it clears all flip-flops in the IC and DA; the accumulator contents are preserved in the DB.

The IOB CONO CLR triggers pulse amplifiers 1A8 and 1A9, which generate negative and positive DC IC CLR pulses (Data Control Initial Conditions Clear). The positive DC IC CLR from PA 1A0 clears the shift counter and the entire IC register (drawings BS-D-136-0-DC1, B4-7 and BS-D-136-0-DC2, A-B). The negative DC IC CLR from 1A8 is applied to pulse amplifier 1A10 (BS-D-136-0-DC1, C5) which generates DC DA $\leftrightarrow$ DB. At the conclusion of DC IC CLR, the three status bits, DB DA MOVE, DA RQ, and DB RQ are all in the 0 state; refer to line 1 of Table A1-4.

IOB Reset Pulse

The clearing function described above can be initiated not only by IOB CONO CLR, but also by IOB RESET (BS-D-136-0-DC1, A1). The latter command line is pulsed whenever computer

power is turned on, and whenever the operator presses the I/O reset key on the computer console. The IOB reset triggers pulse amplifiers 1A8 and 1A9, generating negative and positive DC IC CLR pulses in the same manner as IOB CONO CLR. However, IOB RESET also triggers pulse amplifier 1A10 (BS-D0136-0-DC1, B2), generating an additional clear pulse, DC CLR DB, which clears the data buffer.

Device Selection Gating

Because IOB RESET is intended to clear all I/O control units, it causes an unconditional triggering of pulse amplifiers 1A8, 1A9, and 1A10. It is not gated by any device selection code. In contrast, IOB CONO CLR, and in fact all of the eight control line inputs to the data control except IOB RESET and POWER ON, are intended only for one specific addressed control unit, the unit selected by the device selection code in bits 3 through 9 of the CONO instruction. Consequently, these signals are all gated by the output of diode net 1A6 (BS-D-136-0-DC1, D1).

CONO Set Pulse

One μsec after IOB CONO CLR (at time 0 of the output sequence in TD-D-136-0-TD1) the processor sends an IOB CONO SET pulse (I/O Bus Conditions Out Set) to the data control (BS-D-136-0-DC1 B2). The CONO SET is gated by the data control device selection level and then triggers pulse amplifier 1A8 (BS-D-136-0-DC1, A2). The resulting DC IC SET output pulse causes the parallel transfer of the information on the I/O bus into bits 23 through 35 of the IC register. It is assumed for the present example that this sets the IC register flip-flops to the states indicated in Table A1-6. (See also line 2 of Table A1-4.)

First DATAO Instruction

At the completion of the CONO Out instruction, the data control is ready to respond to the next processor instruction, a DATAO (Data Out) instruction. The purpose of this instruction is to transfer a 36-bit data word from the processor into the data control data buffer, from there to transfer it to the data control accumulator, and then finally to shift it out into the selected I/O device.

DATAO Clear Pulse

The first processor signal to reach the data control during the DATAO instruction is the IOB DATAO CLR (I/O Bus Data Out Clear) pulse (BS-D-136-0-DC1, B1). This pulse is represented by the DATAO CLR operation on the output flow chart (FD-D-136-0-FD1). The DATAO CLR, is gated by the device selection level and then triggers pulse amplifier 1A10 (BS-D-136-0-DC1, B2). The resulting DC CLR DB output pulse from PA 1A10 produces two effects. First, it clears the data buffer register DB. Second, DC CLR DB clears the DB RQ flip-flop through diode gate 1B6 (BS-D-136-0-DC2, B3). The latter operation indicates that the DB register is in use, or at any rate is about to be used, and that succeeding instructions must not attempt to transfer data into it until the DB RQ is again set (indicating that DB is once more free to receive data). The state of the three status bits after the DATAO CLR and DC CLR DB pulses is indicated in line 3 of Table A1-4.

DATAO Set Pulse

One μsec after IOB DATAO CLR, the processor sends an IOB DATAO SET pulse (I/O Bus Data Out Set) to the data control (BS-D-136-0-DC1, C1). This pulse is represented by the DATAO SET operation in flow chart FD-D-136-0-FD1. After being gated by the device selection level, DATAO SET is applied to pulse amplifier 1A9.

I/O Bus Loads DB - Pulse amplifier 1A9 produces a 1-μsec negative output pulse, DC DB ⟵ IOB, which is applied to the Y inputs of the 4230 modules shown in BS-D-136-0-DAB1, D. The leading (negative-going) edge of the DC DB ⟵ IOB pulse loads the 36-bit output data word which the processor has applied to the I/O bus into bits 0 through 35 of the data buffer.

Set DB DA MOVE - Besides gating the Y inputs of DB, the DC DB ⟵ IOB pulse is also sent to input terminal S of inverter 1B9 (BS-D-136-0-DC2, B2), the output of which is in turn applied to input M of capacitor-diode 1B10. One μsec later, the positive-going trailing edge of the pulse sets the DB DA MOVE flip-flop momentarily to the 1 state.

FD-D-136-0-FD1, the output flow diagram, shows the 1-μsec width of the DC DB ⟵ IOB pulse as the delay between the transfers IOB ⟶ DB and DB DA MOVE ⟵ 1. Line 4 of Table A1-4 shows that the DATAO SET pulse and the leading edge of DC DB ⟵ IOB have

no immediate effect on the configuration of the three status bits. Line 5 shows the change in the state of these status bits 1 μsec later, immediately after the positive-going edge of DC DB ⟵ IOB sets the DB DA MOVE flip-flop.

Generation of DC DB ⟷ DA

The setting of the DB DA MOVE flip-flop causes the sudden appearance of a negative input at terminal M of negative diode 1B7 (BS-D-136-0-DC1, C3). Since the DA RQ flip-flop is already in the 1 state (see line 4 of Table A1-4), the 0 to 1 transition of DB DA MOVE (line 5 of Table A1-4) causes diode 1B7 to generate a positive output pulse, thereby triggering 4604 Pulse Amplifier 1A9 (BS-D-136-0-DC1, C4) which generates a 1-μsec negative output pulse, DC DB ⟷ DA. The double arrow indicates that the pulse occurs whenever there is any transfer (in either direction) between the data buffer and the accumulator (line 6 of Table A1-4).

During this first DATAO instruction, DA RQ is in the 1 state, and it is the transition of the DB DA MOVE flip-flop from 0 to 1 that produces the DC DB ⟷ DA pulse. This is represented by the left branch of the loop in the output flow diagram FD-D-136-0-FD1.

Transfer C(DB) Into DA - During the output sequence, the in/out flip-flop is in the out state (IC27 = 1). Consequently the negative-going leading edge of the 1-μsec pulse DC DB ⟷ DA is gated by DC OUT, and pulse amplifier 1A10 (BS-D-136-0-DC2, C5-6) generates DC DB ⟷ DA. Application of this pulse to the terminal W inputs of the 4230 modules shown in BS-D-136-0-DAB1, D causes the jam transfer of the contents of the data buffer into the accumulator. The 36-bit output data word from the processor is thus stored in the accumulator, ready to be shifted out in a series of bytes to be selected I/O device. The data buffer is free to receive another 36-bit output data word at the execution of the next DATAO instruction (line 6 of Table A1-4).

Other Effects of DC DB ⟷ DA (Leading Edge) - Besides transferring C(DB) into DA, the leading edge of the 1-μsec DC DB ⟷ DA pulse produces two other important results. The application of this negative-going leading edge to input terminals W and Y of inverter 1B19 (BS-D-136-0-DC2, A2-3), causes DB DA MOVE to be reset from 1 to 0, and DB RQ to be set from 0 to 1. The resulting configuration of the three status bits is shown in line 6 of Table A1-4.

Reset of DA RQ - Because the DB RQ flip-flop is again in the 1 state, indicating that the data buffer is free, the processor can immediately execute another DATAO instruction. There is, however, one more operation remaining to be executed in the current, first DATAO instruction. One μsec after DB RQ is set, the positive-going trailing edge of the DC DB ⟷ DA pulse resets the DA RQ flip-flop through terminal P of inverter 1B9 (BS-D-136-0-DC2, B2). The final configuration of the status bits at the end of the first DATAO instruction is reflected by line 7 of Table A1-4.

Further Operations

The reset of the DA RQ flip-flop is represented by the last output operation in the left column of the system flow diagram (FD-D-136-0-FD1). From that point on (any time after line 7 of Table A1-4) the actual transfer-out sequence to the external I/O device can begin. The transfer-out sequence is represented by the right-hand column of the system flow diagram; it is described in Output Transfer Sequence below.

Concurrently with the transfer-out sequence, the data control can, of course, start responding to a second DATAO instruction from the processor. The second DATAO instruction can be commenced any time after the setting of DB RQ to 1 (that is, at any time after line 6 of Table A1-4). However, before the processor executes a second DATAO, it must by some means be notified that DB RQ has in fact been set. The two customary programming methods that let the processor know that DB RQ = 1 (that is, data control is ready to accept more data) are described in Section 4. The second DATAO executed by the processor differs from the first in that the DA RQ flip-flop contained 1 at the commencement of the first DATAO (line 2 of Table A1-4), and in contrast contains 0 (line 7 of Table A1-4) at the beginning of the second DATAO.

Second DATAO Instruction

The data control responds to the second DATAO instruction executed by the processor in a slightly different manner than it responded to the first DATAO. In particular, the jam transfer of C(DB) into DA must wait for the I/O device to complete the first transfer-out sequence.

DATAO Clear and Set Pulses

As with the first DATAO instructions, the first processor signal to reach the data control during the second DATAO instruction is IOB DATAO CLEAR. In following the second DATAO instruction on drawing FD-D-136-0-FD1, jump from the bottom of the diagram (at the conclusion of the first DATAO) back to the point just above the DATAO CLR operation. The initial phase of the second DATAO instruction, down to the branch point, is identical with the first DATAO. Note, however, that the three status bits (shown in line 8 of Table A1-4) are not in the same configuration as they were at the same stage of the first DATAO (line 3 of Table A1-4). The DA RQ flip-flop contains 0, indicating that the accumulator is not yet available to receive data during the current instruction.

As in the first DATAO, 1 μsec after the IOB DATAO CLR the processor sends the data control an IOB DATAO SET. This pulse again produces a 1-μsec DC DB ⟵ IOB pulse, the leading edge of which loads the 36-bit output data word from the I/O bus into the data buffer. The leading edge of this pulse has no effect on the status bits (Table A1-4, line 9), but 1 μsec later, the trailing edge sets the DB DA MOVE flip-flop to the 1 state (line 10 of Table A1-4).

Generation of DC DB ⟷ DA

During the first DATAO instruction, the 0 to 1 transition of DB DA MOVE triggered the generation of the pulse DC DB ⟷ DA; this corresponded to the left FIRST PASS ONLY branch of the FD-D-136-0-FD1 output flow diagram. However, during the current second DATAO instruction, the 0 to 1 transition of DB DA MOVE has no immediate effect on the contents of DB. The DA RQ flip-flop does not contain 1, but 0 (see Table A1-4, line 9), when DB DA MOVE is set to 1. Consequently, the system halts with a 1 in the DB DA MOVE flip-flop (at line 10 of Table A1-4). No further operations can be carried out until the word in the accumulator is completely transferred out into the selected I/O device.

When the I/O device transfer-out sequence is complete, and the accumulator is again free to receive data, the final operations of the second output cycle can be carried out. As the last data byte is transferred out from the accumulator into the I/O device, the DC CCT DONE (Character Counter Done) level gates the final DASH CP1 pulse of the current data word through negative capacitor-diode 1B10 (BS-D-136-0-DC2, B2), thereby setting the DA RQ flip-flop to 1, and indicating the end of the transfer-out sequence.

Completion of Second DATAO Cycle

The setting of the DA RQ flip-flop to the 1 state causes the sudden appearance of a negative input at terminal N of negative diode 1B7 (BS-D-136-0-DC1, C3). Since DB DA MOVE = 1, (see line 10 of Table A1-4), the 0 to 1 transition of DA RQ (Table A1-4, line 11) causes diode 1B7 to generate a positive output pulse, thereby triggering pulse amplifier 1A9 (BS-D-136-0-DC1, C4). The pulse amplifier then generates a DC DB $\leftarrow\!\!\!\!\!\!\rightarrow$ DA pulse (represented by the right-hand "A" branch of the output flow diagram at the left of FD-D-136-0-FD1).

The remaining operations of the second DATAO cycle exactly correspond to the final operations of the first DATAO cycle. The leading edge of DC DB $\leftarrow\!\!\!\!\!\!\rightarrow$ DA resets DB DA MOVE to 0 and sets DB RQ to 1. The trailing edge resets the DA RQ flip-flop. The last three configurations of the status bits during the second DATAO (lines 11, 12, and 13 of Table A1-4) are identical to the final configurations of the first DATAO (lines 5, 6, and 7 of Table A1-4).

Additional DATAO Instructions

At the completion of the first DATAO instruction, DB is clear so that the processor can execute a second DATAO immediately. However, the accumulator is still loaded at the completion of the first DATAO instruction, and it remains loaded until the I/O device can receive its contents. This means that after the second DATAO instruction loads the data buffer, the processor must wait for the completion of the first transfer-out sequence to the I/O device. The data control cannot accept a third DATAO instruction until the accumulator is free and the contents of DB have been advanced into the accumulator. The interval between successive DATAO's is governed by the rate at which the I/O device can clear the data control accumulator.

When the last data word has been advanced into the accumulator and transferred out to the I/O device, the DA RQ flip-flop is once more set to 1, indicating that the accumulator is empty and ready to receive data.

Output Transfer Sequence

As soon as DA RQ is reset to 0 (line 7 of Table A1-4) the selected I/O device is free to initiate the output transfer sequence by sending a command pulse to the data control. The output transfer sequence is summarized in the right hand portion (headed by the "B" connector) of FD-D-136-0-FD1, the system flow diagram.

For I/O devices 1-4, the only branch point in the output sequence is that following the CP1 pulse. If the character counter is not done, the output transfer sequence loops back to point B and continues. If the character counter is done, the character counter is reset, and DA RQ is set. The setting of DA RQ terminates the transfer-out sequence, and the output cycle then proceeds to point A in column 3 of the system flow diagram.

Output cycles of I/O devices 5 and 6 are executed in exactly the same manner for n-bit mode (single-character) cycles. For 12-bit and 18-bit mode operations, there is a nested subordinate loop after the CP1 pulse (the left-hand loop). This loop executes the repetitive shifts described in Resulting Internal Command Pulses. Successive DASH LT pulses are produced until the shift counter is done. This requires one return through the left-hand loop for both the first and second character of the data word in 12-bit mode, and two returns through the left-hand loop for the first character in 18-bit mode.

When the character counter is done, the command pulse associated with the final character produces the final CP1 pulse, thereby resetting both the shift counter and the character counter and setting DA RQ. As in the case of output cycles of devices 1-4, the setting of DA RQ terminates the transfer-out sequence and the output cycle again proceeds to point A.

INPUT CYCLE

The operations making up the data control input cycle are listed in Table A1-5, and are shown in the center portion of the system flow diagram, FD-D-136-0-FD1.

Input Cycle CONO Instruction

The CONO instruction which sets up the data control for the input cycle is executed in the same manner as the output cycle CONO. There are, however, two differences in the coding of the effective address. First, for input operations, the in/out bit is 0. Second, the status bits are set to a different initial configuration. The DB DA MOVE flip-flop must be set to 1, and both DA RQ and DB RQ are 0. (See Table A1-2.)

First Readin Sequence

As soon as the IC register is set up, (line 2 of Table A1-5) the selected I/O device is free to initiate the readin sequence by sending an input command pulse to the data control. The readin sequence is summarized by the same portion of the FD-D-136-0-FD1 flow diagram (headed by the "B" connector) which illustrates the transfer-out sequence. The data control performs exactly the same shift operations during the readin sequence as during the transfer-out sequence. The only difference between the two sequences is that for I/O devices 5 and 6 the DASH STRB internal command pulses are generated only during the readin sequence, but not during the transfer-out sequence.

Completion of First Readin Sequence

At the completion of the first readin sequence, DA RQ is set to 1. This causes the input cycle to advance to point A in column 3 of the system flow diagram, FD-D-136-0-FD1. For the first input cycle, the data control follows the left branch of the loop, the branch immediately below the "A" connector. The contents of the accumulator are advanced to the data buffer (line 4 of Table A1-5), and DB RQ is set to 1, indicating to the processor that the data control is ready to read in data. The processor is then free to execute the first DATAI of the input cycle (from line 7 of Table A1-5). One μsec later DA RQ is reset to 0 (line 5 of Table A1-5). This permits the I/O device to initiate a second readin sequence.

Second Readin Sequence

This second sequence is exactly similar to the first. If the processor has not executed a DATAI instruction before the completion of the second readin sequence, the first data word read in will still be stored in the data buffer. Consequently, the second data word read in (stored in the accumulator) should not be advanced to the buffer until the processor reads in the first data word.

Completion of Second Readin Sequence

At the completion of the second readin sequence, the DB DA MOVE contains 0 rather than 1 as it did at the end of the first readin sequence. Consequently, the 0 to 1 transition of DA RQ cannot carry the input cycle to point A as it did after the first readin. Instead, the data

control must follow the left hand branch of the loop. This means that the DA —J→ DB transfer cannot be initiated by the setting of DA RQ, but must instead wait for the 0 to 1 transition of DB DA MOVE. This transition does not occur until the end of the first DATAI instruction executed by the processor.

First DATAI Instruction

During the DATAO instruction, the processor sends a 2.5-μsec negative level IOB DATAI into the data control (BS-D-136-0-DC1, A1). This level is gated by the data control device selection level, and then (as DC DB ⟶ IOB) the gated level is applied to the K terminals of the data buffer (BS-D-136-0-DAB1, D). The negative level at the K terminals places C(DB) on the I/O bus for sampling by the processor. The leading edge of this level (that is, the 0 to 1 transition; see the last line at the left-hand of TD-D-136-0-TD1, the system timing diagram) resets the DB RQ flip-flop to 0, thus indicating that the buffer is empty (or about to be empty) and that the processor should not send any further DATAI instructions until the buffer is reloaded (Table A1-5, line 7). The 1 to 0 transition at the trailing edge of the DC DB ⟶ IOB level sets the DB DA MOVE flip-flop to 1 (Table A1-5, line 8).

Subsequent Operations

When DB DA MOVE is set to 1, the input cycle advances through the left side of the loop below point A. The data word in the accumulator is advanced into the buffer and DB RQ is set to 1 (Table A1-5, line 9). The 1 in DB RQ indicates to the processor that the buffer is loaded, and that the data control is ready to read in a second data word. The processor is therefore free to execute a second DATAI instruction. One μsec later, DA RQ is reset to 0 (Table A1-5, line 10). The selected I/O device is then free to send the command pulse initiating the next readin sequence.

This process continues until the final data word has been read in from the selected I/O device. When that word has been advanced from the accumulator to the buffer, and then read into the processor by a final DATAI instruction (not followed by a DA —J→ DB transfer), the DB DA MOVE is left in the 1 state. The next readin, or series of readins, executed by the I/O device then encounters a completely cleared data control (both accumulator and buffer

empty). The first readin of the new series encounters the 1 in DB DA MOVE and therefore functions like the "first readin sequence" described above. The I/O device can then complete two readin sequences without any requirement to wait for the processor to execute a DATAI instruction.

SECTION 3

INSTALLATION

SITE SELECTION

Before installing the data control, select a suitable location near the PDP-6 arithmetic processor. The data control should be installed within 25 feet of the processor, and the I/O devices should be within 25 feet of the data control.

The data control cabinet is 69-1/2 inches high; width is 22-1/4 inches; depth is 27-1/8 inches. At least 3 feet of clearance should be allowed on all sides of the equipment for access during maintenance. A level floor is required because the equipment frames are mounted on casters. The floor should be capable of supporting 150 psf. The system is designed to operate efficiently from 50° to 100° F. The plug-in modules are cooled by blowing air out the front of the bay. Intake fans are factory-installed in the floor of the bay. No additional cooling equipment is required.

The data control runs on ordinary 115-volt, 60-cycle current. A 10-ampere line is sufficient. The data control power cable is equipped with a Hubble Twist-Loc 3-prong, 30-ampere, 250-volt plug. Although the data control draws only about 3 amperes in normal operation, turn-on surges may reach approximately 5 amperes.

UNPACKING

The data control is shipped on a skid, and may be crated or not, depending on the mode of transportation. For truck shipment it may be left uncrated. A crate is furnished for air shipment. The crate is approximately 74 inches high, 27 inches wide, and 32 inches deep.* The skid upon which it rests is about 6 inches high and 3 feet square. Interconnecting cables are specially made up for each installation, and are shipped with the equipment.

1. If the data control is crated, carefully remove all crating and strapping, and any packing material. If the unit is shipped uncrated, remove the skid and any protective padding.

* The skid upon which it rests is about 6 inches high and 3 feet square.

2. The plenum doors at the rear of the bay have spring catches. To reinforce these doors during shipment two bolts are used to hold each door shut. Remove these bolts and store them in the plastic loops provided.

3. Remove any packing material, shipping blocks, etc. from the inside of the data control.

4. The plug-in modules are taped into the logic panels to prevent damage in shipment. Remove the tape.

> NOTE: If the user plans to reship the equipment (or move it more than a short distance) in the near future, special containers and packing materials should be saved for reuse. These items are designed especially to accomodate the equipment and are the safest means of packing it for reshipment.

INSPECTION

The data control is thoroughly tested and checked before it leaves the factory. However, it should be inspected and checked again when installed to make sure that no damage has occurred during shipment. After the data control is unpacked, check the following:

1. Have all the shipping blocks, packing materials, tape, etc. been removed? If not, remove them.

2. Are all plug-in units inserted firmly in position? Secure any that are loose.

3. Are there any loose nuts or bolts? If so, tighten them.

4. Are there any loose or broken wires? If so, repair them.

CABLE CONNECTIONS

Before connecting cables, complete the inspection procedure given above.

1. Route the four cables from the PDP-6 processor in through the hole in the bottom of the bay. This hole is located directly in front of the fan.

2. Connect the four 22-pin Methode receptacles to the correct Methode plugs at the upper left of the data control logic racks. The four receptacles can be plugged in either to Methode Plugs 1A1-1A4 or to 1B1-1B4, depending on the layout of the specific installation. Each A plug is paralleled with the corresponding B plug; 1A1 is paralleled with 1B1, 1A2 with 1B2, and so on. It is essential that each of the four Methode receptacles at the ends of the processor cables should be connected to the correct pair of plugs. The receptacles should each be marked with the correct plug position. Consult CD-D-166-0-IOMB in Appendix 2 of this manual to make sure that the correct connections are made.

3. Route the 16 I/O device cables (fewer if less than six I/O devices are used) through the hole in the bottom of the bay, and connect the Methode receptacles at the cable ends to the correct plugs 1D10-1D25 at the lower right of the data control logic racks (ID-D-136-0-EDC).

4. A coiled ac power cable is taped to the fan at the bottom of the bay. Remove the tape, route the power cable out through the hole in the bottom of the bay, but do not plug it in.

PREOPERATIONAL CHECKOUT

Before using the data control, make sure that the system turns on and off properly, and that the correct voltages are present at all four logic racks. The checkout procedure makes use of the following switches:

The LOCAL/REMOTE switch at the left of the 811 Power Control. The power control is at the bottom rear of the data control.

The POWER circuit breaker slightly left of center on the 811 Power Control.

The three MCV switches (Marginal Check Voltage) at the front left of each of the four logic racks. In each of these four sets of MCV switches, the top switch is +10 vdc (A), the middle switch is +10 vdc (B), and the bottom switch is -15 vdc. All controls and indicators are described in detail in Section 4.

The following checkout procedure should be carried out after completing the cable connections described above. All voltages are measured from chassis ground with a Multimeter (Simpson Model 260A, Triplett Model 630NA, or equivalent).

1. Put the LOCAL/REMOTE switch in LOCAL position (up).

2. Turn POWER circuit breaker OFF (down).

3. Plug in power cable to 110-volt ac outlet.

4. Put all 12 MCV switches in FIXED position (down).

5. Turn POWER circuit breaker ON (up).

6. Check the fixed (internal) supply voltages at the terminals listed below.

Voltages:	+10 vdc (A)	+10 vdc (B)	-15 vdc
Terminals:	1A25A	1A25B	1A25C
	1B25A	1B25B	1B25C
	1C25A	1C25B	1C25C
	1D25A	1D25B	1D25C

The +10 vdc fixed supply voltages should be between +9.5 vdc and +11 vdc. The -15 vdc should be between -14.5 vdc and -16 vdc. If either voltage falls outside the specified range, the 728 Power Supply probably needs maintenance.

7. With the PDP-6 processor turned off, put the LOCAL/REMOTE switch in the REMOTE position (down). Check each of the fixed voltages at any terminal listed in step 6. No voltage should be present.

8. Turn on the PDP-6 processor. Again check each of the three fixed voltages. The same voltage recorded in step 6 should be present at each of the three terminals checked.

9. Put the LOCAL/REMOTE switch in the LOCAL position (up).

10. Put the four MCV switches which control the +10 vdc (A) lines (upper switch at the left of each logic rack) in the MARGINAL position (up).

11. Make the following settings at the MCV controls above the PDP-6 operator's control panel:

    a. Set polarity switch to +10 volt position (left).

    b. Adjust variac until marginal check voltage meter indicates +5 vdc.

12. Check the +10 vdc (A) voltage at the four terminals specified in step 6. It should coincide with the +5 vdc voltage shown on the marginal check voltage meter.

13. Return the MCV switches for +10 vdc (A) to the FIXED position (down) and set the MCV switches which control the +10 vdc (B) lines (middle switch at the left of each logic rack) to the MARGINAL position (up).

14. Check the +10 vdc (B) voltage at the four terminals specified in step 6. It should coincide with the +5 vdc voltage shown on the marginal check voltage meter.

15. Return the MCV switches for +10 vdc to the FIXED position (down).

16. At the PDP-6 MCV controls, set the MCV polarity switch to the -15 volt position (right) and adjust the variac until the marginal check voltage meter indicates -10 vdc.

17. Set the MCV switches which control the -15 vdc lines (bottom switch at the left of each logic rack) to the MARGINAL position (up).

18. Check the -15 vdc voltage at the four terminals specified in step 6. It should coincide with the -8 vdc voltage shown on the marginal check voltage meter.

19. Return the MCV switches for -15 vdc to the FIXED position (down).

20. Return the LOCAL/REMOTE switch to REMOTE.

21. Return the polarity switch at the PDP-6 to the center (off) position.

22. Turn off the PDP-6.

SECTION 4

OPERATING PROCEDURES AND PROGRAMMING

The purpose of this section is to provide the operator with the information needed to operate and program Data Control Type 136. A short section is devoted to descriptions of the few controls and indicators included in the data control. The remainder of the section describes the PDP-6 programming techniques used to govern the output and input cycles of the data control and its associated I/O devices. A thorough understanding of the relevant PDP-6 programming will not only facilitate operational use of the data control and its I/O devices, but also will be the best aid to efficient maintenance.

CONTROLS AND INDICATORS

Controls

811 Power Control

The 811 Power Control is mounted at the bottom of the data control cabinet. On its front panel (at the rear of the data control) are two controls, a circuit breaker, and a switch.

LOCAL/REMOVE Switch - This switch is located at the left of the 811 Power Control. In the LOCAL position (up), the processor has no effect on the data control power turn-on. The data control power can then be turned on or off by the POWER circuit breaker. The LOCAL position is used primarily for maintenance purposes. For normal operation it is usually more convenient to leave the circuit breaker on and the switch in the REMOTE position (down). The data control power is then turned on by the -15 vdc POWER ON level which is sent to all I/O units while the processor is turned on; refer to Ungated Signals in Section 2.

POWER Circuit Breaker - This circuit breaker is located to the right of the LOCAL/REMOTE switch. When the circuit breaker is in the OFF position (down) the data control power cannot be turned on. When it is in the ON position (up), and the LOCAL/REMOTE switch is

set to LOCAL, the data control power is on. When the circuit breaker is in the ON position, and the LOCAL/REMOTE switch is set to REMOTE, the data control power is on only when the processor power is on.

828 Power Distribution Panel

This convenience panel is located at the front of the data control just below the four logic racks. At the left of the panel is the DISTRIBUTION PANEL circuit breaker. When turned ON (up), this circuit breaker applies power from the hot side of the POWER circuit breaker to the four 110-volt ac 3-prong receptacles on the 828 Distribution Panel. This arrangement permits the application of power to the distribution panel whether or not the data control power is turned on. Power can be removed from all four receptacles by turning the DISTRIBUTION PANEL circuit breaker off (down).

MCV Switches

There are 12 MCV (Marginal Check Voltage) switches on the front of the data control. One set of three MCV switches is located at the left of each of the four data control logic racks. The top switch in each set governs the +10 vdc (A) power lines, the middle switch governs the +10 vdc (B) lines, and the bottom switch governs the -15 vdc lines. There are two independent +10 vdc supply lines, A and B. The two lines are applied to different logical circuits in each module. This is an aid to trouble isolation because it permits selective application of marginal check voltages. In the FIXED position (down) each MCV switch connects the associated logic rack to the fixed supply voltage from the data control internal 728 Power Supply. In the MARGINAL position (up) each MCV switch connects the associated logic rack to the variable MCV voltage from the processor. This variable MCV voltage can be adjusted at the MCV controls over the operator's control panel of the PDP-6.

MCV Controls at Processor

The processor MCV controls include a 3-position polarity switch (+10 volts, OFF, -15 volts), a variac, and a marginal check voltage meter. When the polarity switch is in the +10 volts position, the variac can apply from 0 vdc to +20 vdc to the +10 vdc (A) and (B) lines. When the polarity switch is in the -15 volts position, the variac can apply from 0 vdc to -20 vdc to the -15 vdc lines.

Indicators

There are nine indicator lights on indicator card 1B22 (on the front of the data control) at the right of the second logic rack). These indicator lamps display the contents of nine bits of the data control IC register. When the lamps are on, the corresponding bits contain ones, when off, zeros. The nine bits represented by the nine front panel indicator lamps are shown in Table 4-1.

TABLE 4-1 FRONT PANEL INDICATOR LAMPS
(Listed from Top to Bottom)

Bit Designation	Location
DA RQ	IC25
DB RQ	IC26
DBDA MOVE	IC24
IN/OUT	IC27
CH MODE 0	IC28
CH MODE 1	IC29
CCT 0	IC20
CCT 1	IC21
CCT 2	IC22

PROCESSOR INSTRUCTION CODING

Input-output instructions executed by the processor initiate all data control operations. Every IOT instruction in the PDP-6 repertoire is coded with the format shown with PDP-6 Handbook, F-65, page 22. The CONO-DATA CONTROL address coding is summarized for the operator's convenience in Table A1-7 in Appendix 1.

SAMPLE PROGRAMS

There are two ways in which the processor can be notified that the data control is ready for another data word: by an active priority interrupt request, or by passively waiting for the processor to sample IC26, the DB RQ flip-flop.

The first method of notifying the processor, the passive sampling method, is illustrated by the output and input programs presented in Use of Data Control Without Priority Interrupt, below. This method has the disadvantage of keeping the processor tied up during the entire output or input cycle even though the operations required of it take only a small fraction of the time used by the I/O device. Nevertheless, this method might be used if the programmer wished to to have the processor complete an output or input sequence before proceeding to other operations.

The second method of notifying the processor that the data control is ready, the active priority interrupt request, is illustrated by the output and input programs presented in Use of Data Control with Priority Interrupt, below. This second method is often the more efficient, in that neither the processor nor the I/O device ever need be idle. The I/O device operates asynchronously at its own speed, and the processor takes part in the output or intput sequence only during the brief intervals needed to transfer a data word out to DB or to read a data word in from DB. The processor never has to wait for the comparatively slow output transfer sequence or readin sequence of the I/O device, but instead is used only for fast, parallel, full data-word transfers to and from the data control itself.

Use of Data Control Without Priority Interrupt

The following sample programs illustrate the use of the data control without the priority interrupt system.

SAMPLE OUTPUT PROGRAM $\overline{\text{PI}}$

```
100/CONO DC, (DARQ, DBRQ, OUT, MODE, DEVICE #)
101/CONSO DC, DBRQ
102/JRST, .-1
103/BLKO DC, STORE
104/JRST, DISMISS
105/JRST, .-4
```

SAMPLE INPUT PROGRAM $\overline{PI}$

```
100/CONO DC, (DBDA MOVE, IN, MODE, DEVICE #)
101/CONSO DC, DB RQ
102/JRST, .-1
103/BLKI DC, STORE
104/JRST, DISMISS
105/JRST, .-4
```

When using the Data Control out of the PI system, the IC register is set up by the instruction in 100. At the next instruction (101) the DB RQ flag is interrogated. If it is a 1, the next instruction (102) is skipped; if it is a 0, instruction 102 is executed. Instruction 102 says: jump from this location minus one instruction (to instruction 101). Therefore the program remains in this loop until DB RQ becomes a 1. The next instruction is BLKO or BLKI (depending on direction of word transfer). This instruction executes a data instruction (DATAO or DATAI) and skips the next instruction and jumps back to 101. This process is repeated until the entire block of information has been read in or transferred out. When the block is complete, the program executes 104 and jumps to a dismiss routine or another part of the main program.

Use of Data Control With Priority Interrupt

The following sample programs illustrate the use of the data control with the priority interrupt system.

SAMPLE OUTPUT PROGRAM PI

```
100/CONO DC, (DARQ, DBRQ, OUT, MODE, DEVICE #, PI # 1)
101/JRST, .
42/BLKO DC, STORE
43/JSR, DISMISS
```

SAMPLE INPUT PROGRAM PI

```
100/CONO DC, (DBDA MOVE, IN, MODE, DEVICE #, PI # 1)
101/JRST, .
42/BLKI DC, STORE
43/JSR, DISMISS
```

When using the data control with the PI system, the IC register is set up by the instruction in 100; note also that PI channel #1 is selected. The program then executes 101, which jumps to itself until an interrupt occurs. When an interrupt has occurred, the program executes the BLKO or BLKI instruction in location 42. After each data instruction (DATAO or DATAI) the program returns to 101, and iterates that one-instruction loop until the next interrupt. The instruction in 43 will be executed only when the entire block of information has been transferred. In these two sample programs, the JRST one-instruction loop in 101 takes the place of an actual operating routine. For actual working programs, the return would be to a more useful routine.

Further discussion of input-output programming is presented in the PDP-6 Handbook, F-65, on pages 51, 52, and 65 through 68.

SECTION 5

MAINTENANCE

The data control is ordinarily used as a peripheral system of the PDP-6 processor installation. The maintenance chapter of the processor manual covers a number of topics not repeated in the data control manual, but essential to efficient maintenance of all DEC systems used with the PDP-6, including the data control. These topics include Special Tools and Test Equipment, Log Entries, Maintenance Programs, Use of Marginal Check, Troubleshooting, and Module Removal and Repair.

CAUTION

> The procedures described in the maintenance chapter of the PDP-6 processor manual should be thoroughly understood before undertaking the troubleshooting and repair of the data control.

USE OF DRAWINGS

The complete system logic is shown in the engineering logic drawings, in Appendix 2. BS-D-136-0-DC1 and DC2, BS-D-136-DAB1 and DAB2, and BS-D-136-0-DASH. Because these five drawings are the most frequently used source of troubleshooting information, it is important to be familiar with the symbols and conventions which they employ. The symbols used on these drawings are explained in Figure A2-1, DEC Logic Symbols, which precedes the engineering drawings in Appendix 2. A more detailed explanation of DEC logical design is presented in the Digital Logic Handbook.

Equipment Layout

The data control logic is contained in four DEC logic racks designated from top to bottom 1A-1D. The modules located in each rack position, and the function of each module, are shown in UML-D-136-0-DCML, Sheets 1 and 2, the logic-rack layout diagrams in Appendix 2. Power wiring is shown in PW-D-136-0-DCPW. The power supply is shown in RS-728. The power control is in RS-811. Circuit schematics of all data control modules are also presented in Appendix 2.

PREVENTIVE MAINTENANCE

Preventive maintenance procedures should be performed on a regular basis. By appropriate use of scheduled preventive maintenance techniques (described in the Arithmetic Processor portion of the PDP-6 Maintenance Manual) most potential malfunctions can be detected before they occur.

The cooling fan in the bottom of the bay should be checked daily for proper operation and free flow of air. Under normal operating conditions the air filter at the bottom of the bay should be changed and cleaned monthly. The correct procedure for changing and cleaning filters is described in the maintenance chapter of F-67(166).

RECOMMENDED SPARE PARTS

The most economical quantity of spare parts to be maintained depends on the requirements of the individual user, but the following listing is suitable for most installations.

Module Spares

Module	Quantity
1310	1
1669	1
4102	1
4112	1
4113	1
4118	1
4127	1
4129	1
4143	1
4150	1
4151	1
4215	1
4217	1
4230	2

Module	Quantity
4604	1
4606	1
4657	1
6102	1
6115	1
6117	1
6684	1

Component Spares

	Type		Quantity
Transistors:	2N456A	RCA	2
	2N485	Sprague	2
	2N489	Sprague	3
	2N599	GE	5
	2N711A	GE	3
	2N1184	RCA	3
	2N1184B	RCA	2
	2N1204	Motorola	3
	2N1304	GE	2
	2N1305	GE	5
	2N527	GE	5
	2N2451	Sprague	8
	2N1754*	General Instruments	3
	2N1754	General Instruments	5
	2N393	General Instruments	2

*BVCES 40V at 100 μa

Diodes:	1N270	Transitron	1
	1N276	Transitron	20
	1N645	Transitron	15
	1N994	Transitron	4

	Type		Quantity
	1N1217	GE	2
	1N1220	GE	1
	1N429 (Zener)	Motorola	1
	1N3316 (Zener)	Motorola	1
	GRS20SP4B4	GE	1
Pulse Transformers:	T2003	DEC	3
	T2006	DEC	1
	T2010	DEC	2
	T2012	DEC	1
	T2017	DEC	1
	T2018	DEC	1
	T2019	DEC	1
	T2020	DEC	1
	T2021	DEC	1
	T2023	DEC	1
	T2024	DEC	2
	T2029	DEC	2
	T2033	DEC	1

Mechanical Spares

Part Number and Description	Quantity
53E168, Type CFG: Rotron fan with #2R blade	1
X-1431, 10" x 10" x 2" EZ Kleen Filter	1
Type 418 Super Filter Kote, pints	1

Table A1-1 IC Register Bit Assignment

APPENDIX 1

FOLDOUT TABLES

TABLE A1-1 IC REGISTER BIT ASSIGNMENT

Field	Bit No.	Flip-flop Designation	Meaning of Designation
Character Counter	20	CCT 0	Character Counter Bits 0-2
	21	CCT 1	
	22	CCT 2	
Error Bit	23	DATA CLBD	Data lost; error condition (clobbered)
Status Bits	24	DBDA MOVE	Buffer-Accumulator Transfer Synchronization Flip-flop
	25	DA RQ	Data Accumulator Request Flip-flop
	26	DB RQ	Data Buffer Request Flip-flop
Input/Output	27	IN/OUT	Input/Output Flip-flop
Character Mode	28	CH MODE 0	Character Mode Bits 0-1
	29	CH MODE 1	
Internal Device Address	30	DEVICE 0	Internal Device Address Bits 0-2
	31	DEVICE 1	
	32	DEVICE 2	
Priority Interrupt Assignment	33	PIA 0	Priority Interrupt Assignment Bits 0-2
	34	PIA 1	
	35	PIA 2	

Table A1-2 Shift-Counter and Character Counter Done

TABLE A1-2 SHIFT-COUNTER AND CHARACTER COUNTER DONE

Char Mode IC28-29	No. of Bits in Char	No. of Shifts per Char	C(SCT) When DC SCT DONE	Asserted Output	No. of Char per Word	No. of Char per Word-1	C(CCT) When DC CCT DONE	Asserted Output
				I/O Devices 1-4				
0 0	6	1 *	0 0	1A13V	6	5	101	1A15Z
				I/O Devices 5 and 6				
0 1	n**	0	0 0	1A13V	1	0	000	1A15T
1 0	12	2	1 0	1A13R	3	2	010	1A15L
1 1	18	3	1 1	1A13K	2	1	001	1A14L

* Occurs at read in or transfer out command. Not governed by SCT.
** $1 \leq n \leq 36$

Table A1-3 Significance of DA RQ and DB RQ States

TABLE A1-3 SIGNIFICANCE OF DA RQ AND DB RQ STATES

State	In/Out	Data Accumulator is:	Data Control is:
DA RQ			
DA RQ = 0	Out	Loaded	READY to transfer out to I/O device
DA RQ = 0	In	Empty	READY to read in from I/O device
DA RQ = 1	Out	Empty	NOT READY to transfer out
DA RQ = 1	In	Loaded	NOT READY to read in
DB RQ			
DB RQ = 0	Out	Loaded	NOT READY for transfer out from processor
DB RQ = 0	In	Empty	NOT READY to read in to processor
DB RQ = 1	Out	Empty	READY for transfer out
DB RQ = 1	In	Loaded	READY to read in

Table A1-4 Configurations of Status Bits During Output Cycle

TABLE A1-4 CONFIGURATIONS OF STATUS BITS DURING OUTPUT CYCLE

Line	Preceding Event	Configuration of Status Bits: DBDA MOVE (IC 24)	DA RQ (IC 25)	DB RQ (IC 26)	Loaded Registers
CONO (Out) Instruction:					
1	CONO CLR, DC IC CLR	0	0	0	None
2	CONO SET, DC IC SET	0	1	1	None
First DATAO Instruction:					
3	DATAO CLR, DC CLR DB	0	1	0	None
4	DATAO SET, DC DB ⟵IOB	0	1	0	DB
5	1 μsec delay: DC DB ⟵IOB (t.e.)*	1	1	0	DB
6	DC DB ⟷ DA (l.e.)	0	1	1	DB-DA (same word)
7	1 μsec delay: DC DB ⟷ DA (t.e.)	0	0	1	DB-DA (same word)
Second DATAO Instruction: (can begin any time after line 6)					
8	DATAO CLR, DC CLR DB	0	0	0	DA
9	DATAO SET, DC DB ⟵IOB	0	0	0	DB,DA
10	1 μsec delay: DC DB ⟵IOB (t.e.)	1	0	0	DB,DA
Transfer-out Sequence: (can begin any time after line 7; duration depends on I/O device data rate)					
11	Transfer-out completed (CCT DONE and CP1)	1	1	0	DB
12	DC DB ⟷ DA (l.e.)	0	1	1	DA
13	1μsec delay: DC DB ⟷ DA (t.e.) (at any time hereafter a second transfer-out sequence can begin)	0	0	1	DA
Third and Succeeding DATAO Instructions: (Same as second DATAO. Can begin any time after line 12.)					

* Note: l.e. = leading edge of pulse; t.e. = trailing edge.

Table A1-5 Configurations of Status Bits During Input Cycle

TABLE A1-5 CONFIGURATIONS OF STATUS BITS DURING INPUT CYCLE

Line	Preceding Event	Configuration of Status Bits: DBDA MOVE (IC 24)	DA RQ (IC 25)	DB RQ (IC 26)	Loaded Registers
CONO (In) Instruction:					
1	CONO CLR, DC IC CLR	0	0	0	None
2	CONO SET, DC IC SET	1	0	0	None
First Read-in Sequence: (Note that read-in duration depends on I/O device data rate.)					
3	Read-in completed. (CCT DONE and CP1)	1	1	0	DA
4	DC DB ⟷ DA (l.e.)*	0	1	1	DB
5	1 μsec delay: DC DB ⟷ DA (t.e.)	0	0	1	DB
Second Read-in Sequence: (can begin any time after line 5)					
6	Read-in completed (CCT DONE and CP1)	0	1	1	DA, DB
First DATAI Instruction: (can occur any time after line 4)					
7	IOB DATAI 0 ⟶ 1, DB ⟶ IOB	0	1	0	DA
8	2.5 μsec delay: IOB DATAI 1 ⟶ 0	1	1	0	DA
9	DC DB ⟷ DA (l.e.) (at any time hereafter a second DATAI can begin.)	0	1	1	DB
10	1 μsec delay: DC DB ⟷ DA (t.e.) (at any time hereafter another read-in can begin.)	0	0	1	DB

* Note: l.e. = leading edge of pulse; t.e. = trailing edge.

Table A1-6 CONO Bit Assignments - Output Cycle

TABLE A1-6 CONO BIT ASSIGNMENTS - OUTPUT CYCLE

Bit No.	Contents	Meaning
23	0	No DATA CLBD error condition present.
24	0	Initial state of DADB MOVE synchronizing flip-flop.
25	1	DA RQ = 1; indicates that accumulator is initially empty.
26	1	DB RQ = 1; indicates that data buffer is initially empty.
27	1	An output cycle follows.
28-29	00	Data to be sent out in 6-bit character mode.
30-32	001	Data to be sent to I/O device No. 1.
33-35	011	Priority Interrupt Channel No. 3 assigned to data control.

Table A1-7 Effective Address Coding for CONO Data Control Instructions

TABLE A1-7 EFFECTIVE ADDRESS CODING FOR CONO DATA CONTROL INSTRUCTIONS

Bit Nos.	Field	Configuration	When Applicable
23	Error Bit	0	All Cycles
24-26	Status Bits	X 1 1	Output Cycles
		1 0 0	Input Cycles
27	In/Out	1	Output Cycles
		0	Input Cycles
28-29	Character Mode	0 0	All 6-bit Cycles
		0 1	All n-bit Cycles ($1 \leq n \leq 36$)
		1 0	All 12-bit Cycles
		1 1	All 18-bit Cycles
30-32	Internal Device Address	0 0 1	I/O Device No. 1
		0 1 0	I/O Device No. 2
		. . .	. . . . .
		1 1 0	I/O Device No. 6
33-33-35	Priority Interrupt Assignment	0 0 0	No PI Assignment
		0 0 1	PI Request Channel 1 Assigned
		. . .	. . . . .
		1 1 1	PI Request Channel 7 Assigned

APPENDIX 2

ENGINEERING DRAWINGS

Reduced engineering drawings are produced in this appendix as an aid to understanding and maintaining the system. A complete set of formal engineering drawings is supplied separately with each system. Should any discrepancy exist between the drawings in this manual and those supplied with the equipment, assume the formal drawings to be correct.

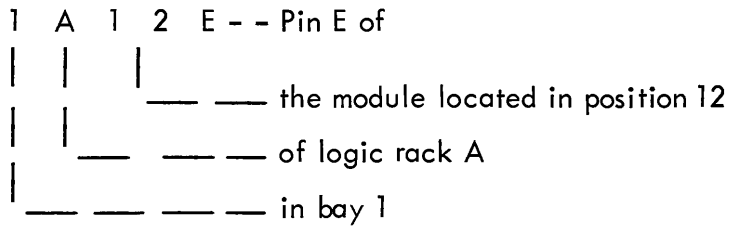
Refer to the Table of Contents for a complete list of engineering drawings for this system and the pages on which the drawings appear.

For all modules illustrated, circuit type is always shown as a 4-digit number. This number is the same type number used to identify the circuit in the DEC System Modules Catalog.

Examples: 1310 - - - Delay Line (5-mc series)
4112 - - - Six 2-Input Negative Diode Gates (500-kc series)

On all circuit modules the circuit location code is lettered directly below the circuit type number. Circuit location code is shown as a single letter preceded by one digit and followed by one or two digits.

Example:



As indicated on certain block schematics, the normal 4-digit module identification number is followed by the letter "R." This letter indicates that these modules contain internal jumpers which connect output terminals to clamped load resistors located on the module. Such modules are shipped from Digital with all jumpers connected. Before these modules can be installed in a particular Digital component, the module user must consult the block schematic for that component to determine which jumpers are to be removed and which jumpers are to remain connected.

Each engineering logic drawing is divided into 32 zones (4 horizontal, and 8 vertical) by marginal map coordinates. Figure references in the text are usually followed by a letter and a digit specifying the zone in which the referenced circuit is located.

SYMBOLS AND TERMINOLOGY

Engineering drawing numbers for this equipment contain five pieces of information, separated by hyphens. Read from left to right these bits of information are a 2-letter code specifying the type of drawing, a 1-letter code specifying the size of the drawing, the type number of the equipment, the manufacturing series of the equipment, and a 2-digit number specifying the number of a drawing within a particular series. The drawing type codes are:

a. BS, block schematic or logic diagram
b. CD, cable diagram
c. CS, circuit schematic
d. FD, flow diagram
e. ID, interconnection drawing
f. PW, power wiring
g. RS, replacement schematic
h. SD, system diagram
i. TD, timing diagram
j. TFD, timing and flow diagram
k. UML, utilization module list
l. WD, wiring diagram

Logic Signals

All logic signals are either standard Digital logic levels or standard Digital pulses. A standard Digital logic level is either a ground (0 to -0.3 volts) or -3 volts (-2.5 to -3.5 volts). Logic signals are generally given mnemonic names which indicate the condition represented by assertion of the signal. An open diamond (——◇) indicates that the signal is a level and that ground represents assertion; a solid diamond (——◆) indicates that the signal is a level and that -3 volts represents assertion. All logic levels applied to the conditioning-level inputs of capacitor-diode gates must be present either 1 or 3 microseconds (depending on the module used) before an input triggering pulse is applied to the gate.

The standard Digital negative pulse is indicated by a solid triangle (——►) and goes from ground to -2.5 or -3 volts (-2.3 to -3.5 volt tolerances). The standard Digital positive pulse, indicated by an open triangle (——▷), goes either from -3 volts to ground or from ground to +2.5 volts (+2.3 to +3.0 volts). The width of the standard pulses used in this equipment is either 1.0, 0.4, or 0.07 microseconds, depending on the module and application.

Occasionally, the transition of a level is used at an input where a standard pulse is otherwise expected and a composite symbol (——◇▷) is drawn to indicate this fact. The triangle is drawn open or solid depending respectively on whether the positive (-3 volt to ground) or the negative (ground to -3 volt) transition triggers circuit action. The shading of the diamond either is the same as that of the triangle to indicate triggering on the leading edge of a level, or is opposite that of the triangle to indicate triggering on the trailing edge.

Any other signal is non-standard and is indicated by an arrowhead (——→) pointing in the direction of signal flow. Figure A2-1 shows the standard symbols used in all Digital logic drawings.

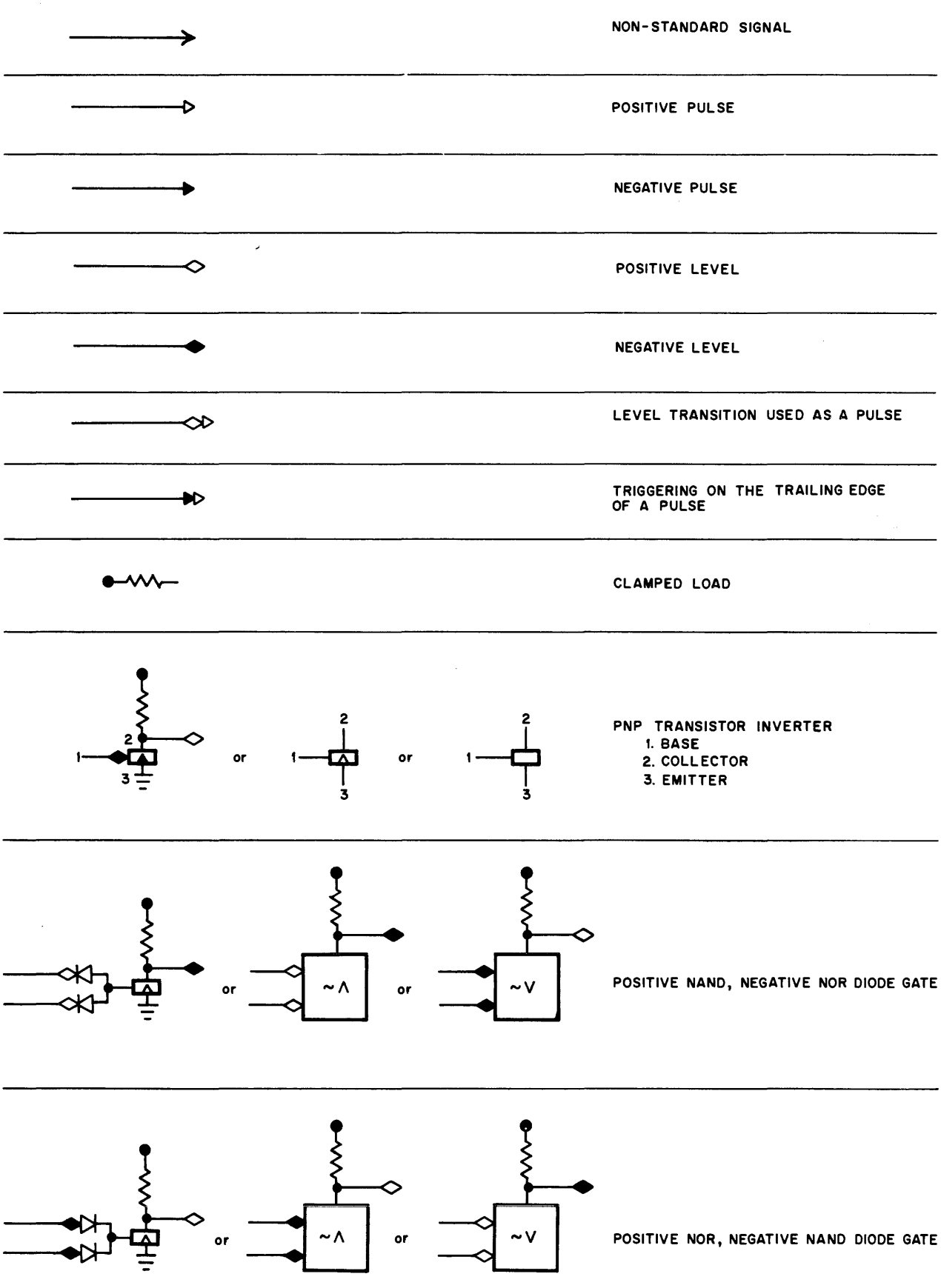
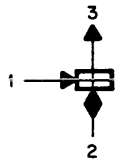
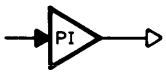


Figure A2-1 DEC Logic Symbols

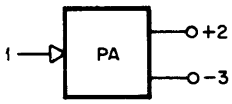


CAPACITOR-DIODE GATE, POSITIVE OR NEGATIVE INDICATED BY POLARITY OF THE INPUTS.

1. PULSE INPUT
2. CONDITIONING LEVEL INPUT
3. PULSE OUTPUT



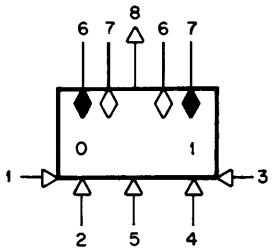
PULSE INVERTER



PULSE AMPLIFIER

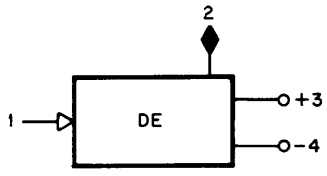
1. PULSE INPUT, POLARITY INDICATED BY INPUT SIGNAL

2,3. TRANSFORMER-COUPLED PULSE OUTPUT



FLIP-FLOP (MOST FLIP-FLOPS HAVE ONLY SOME OF THE FOLLOWING):

1. DIRECT-CLEAR INPUT
2. GATED-CLEAR INPUT
3. DIRECT-SET INPUT
4. GATED-SET INPUT
5. COMPLEMENT INPUT
6. OUTPUT LEVEL, -3 V IF 0, 0 V IF 1
7. OUTPUT LEVEL, 0 V IF 0, -3 V IF 1
8. CARRY PULSE OUTPUT

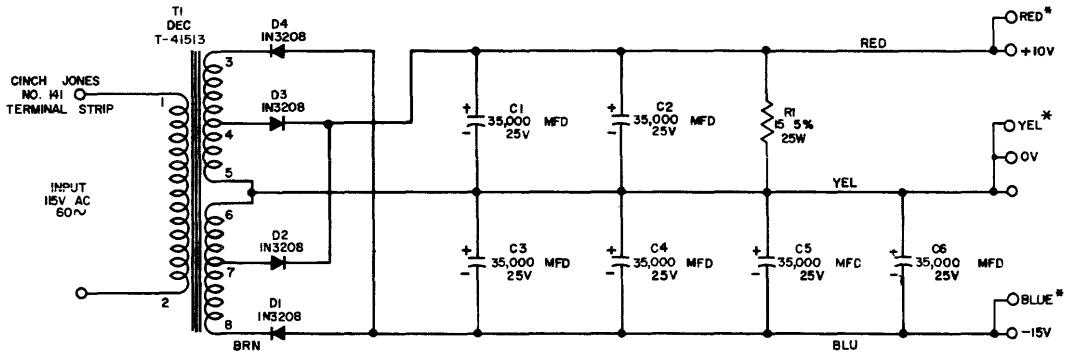


DELAY (ONE-SHOT MULTIVIBRATOR)

1. INPUT PULSE
2. OUTPUT LEVEL, -3V DURING DELAY

3,4. TRANSFORMER-COUPLED PULSE OUTPUT

Figure A2-1 DEC Logic Symbols (continued)



* HEYMAN MFG. CO.
TAB TERMINALS

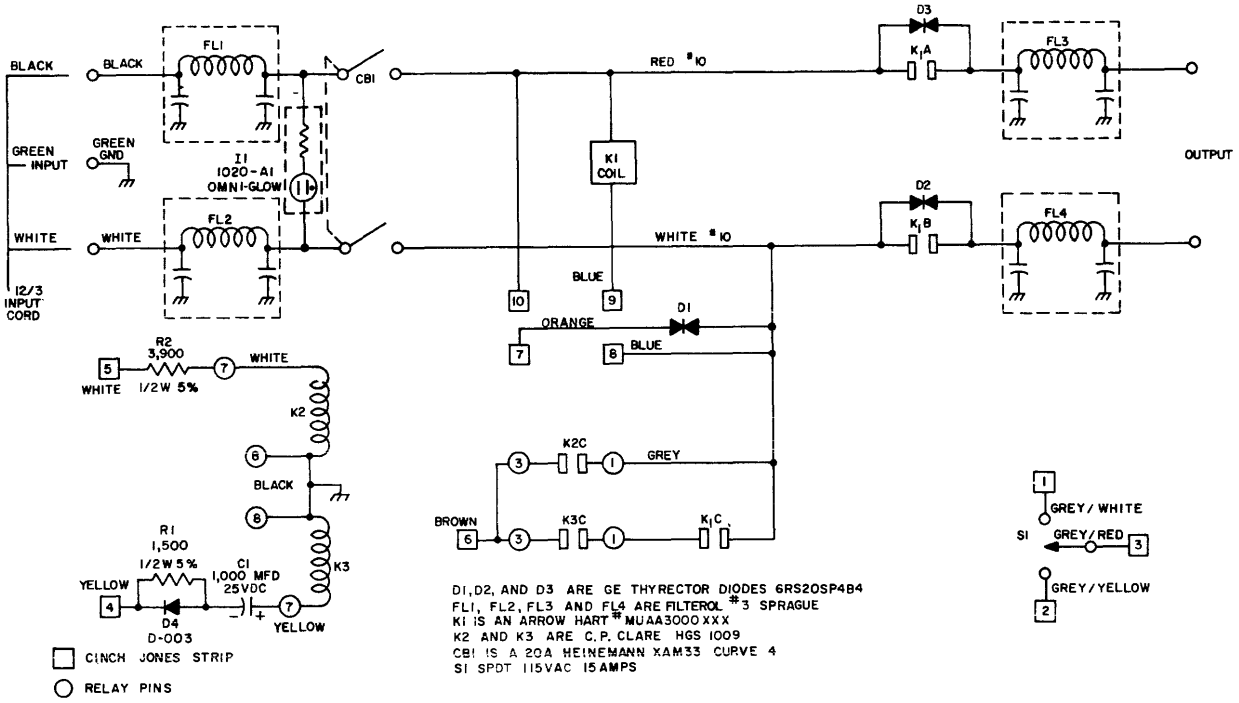
NOTE:
IN ORDER TO KEEP OUTPUT VOLTAGE WITHIN THE FOLLOWING LIMITS:
+10V: +9.5 TO +11V
-15V: -14.5 TO -16V
THE LOADING SHOULD BE WITHIN THE FOLLOWING LIMITS:

BOTH SIDES LOADED	+10 V	0	TO	7.0	AMPS
	-15 V	1.0	TO	8.0	AMPS
ONE SIDE LOADED	+10 V	0	TO	7.5	AMPS
	-15 V	1.0	TO	8.5	AMPS

SUM OF THE OUTPUT CURRENTS ARE LIMITED BY THE FOLLOWING EQUATION: $5I_{10} + 6I_{15} \leq 53$

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
IN3208	IN3208		

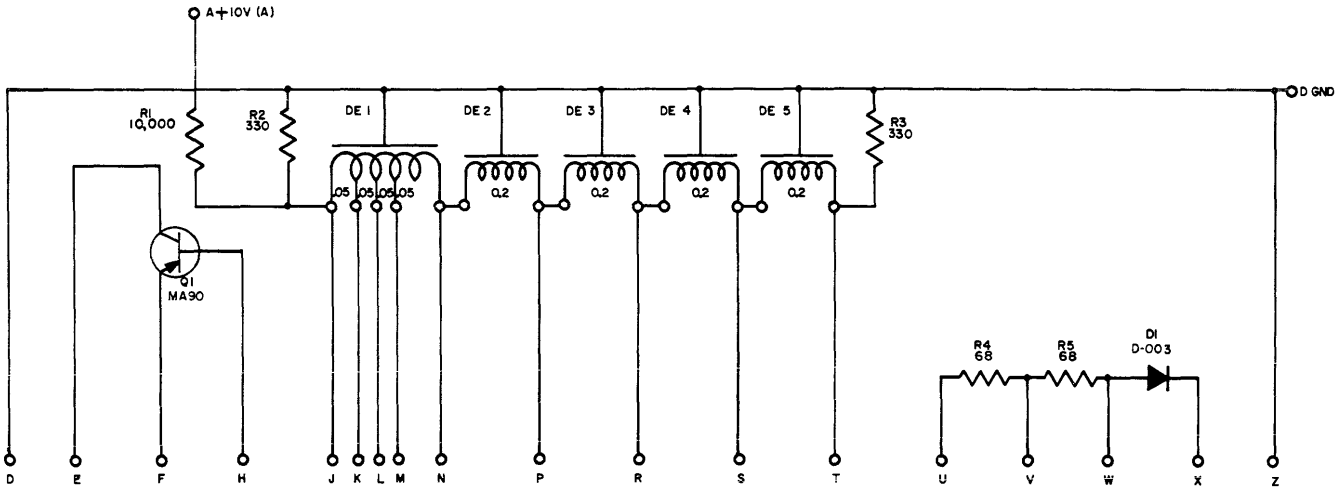
Power Supply RS-728



D1, D2, AND D3 ARE GE THYRECTOR DIODES 6RS20SP4B4
FL1, FL2, FL3 AND FL4 ARE FILTEROL #3 SPRAGUE
K1 IS AN ARROW HART # MUAA3000XXX
K2 AND K3 ARE C.P. CLARE HGS 1009
CB1 IS A 20A HEINEMANN XAM33 CURVE 4
SI SPDT 115VAC 15AMPS

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
D-003	IN994		

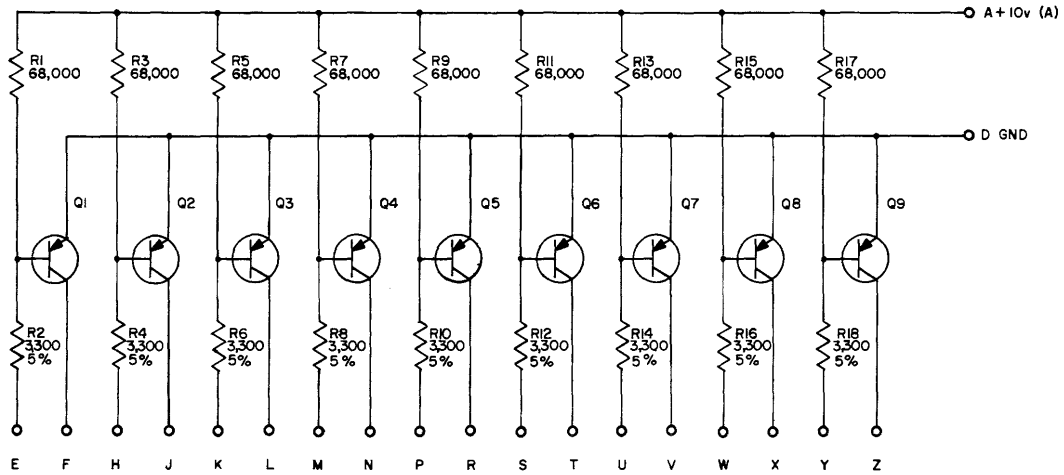
Power Control RS-811



UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/2W; 10%
DE1= TECHNITROL 0.2 μ sec DELAY LINE 330 OHMS TAPPED AT 0.05 μ sec. INTERVAL
DE2= DE5 TECHNTROL 0.2 μ sec. DELAY LINE

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
MA90	2N2481		
D-003	IN994		

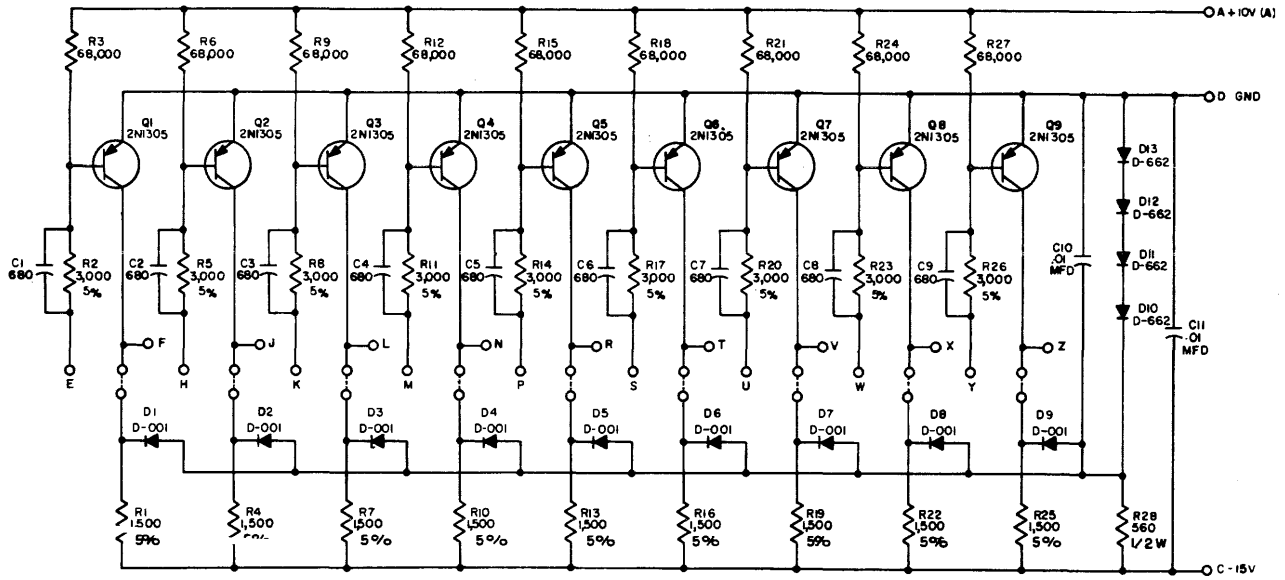
Delay Line RS-1310



UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%
TRANSISTORS ARE 4JX1C741

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
4JX1C741	2N527		

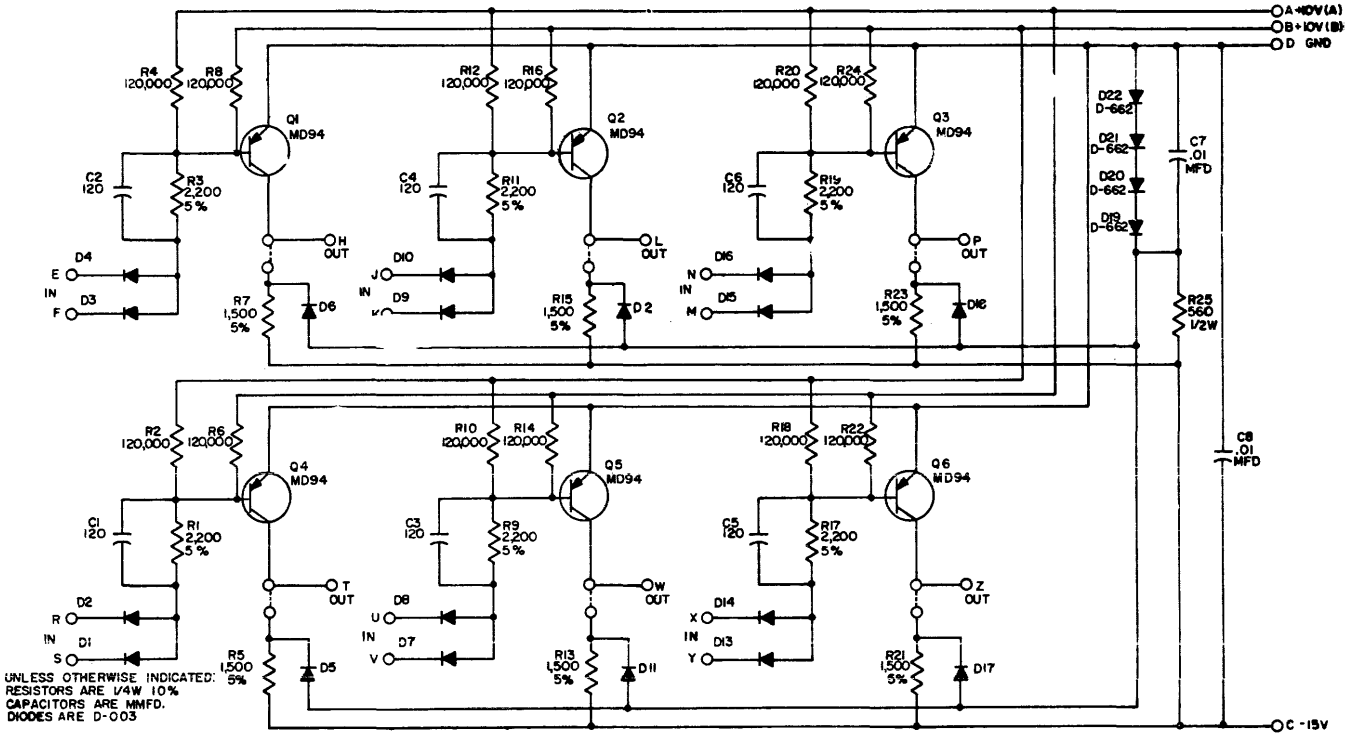
Indicator Driver RS-1669



UNLESS OTHERWISE INDICATED
RESISTORS ARE 1/4W 10%
CAPACITORS ARE MMFD

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
2N1305	2N1305		
D-001	1N276		
D-662	1N645		

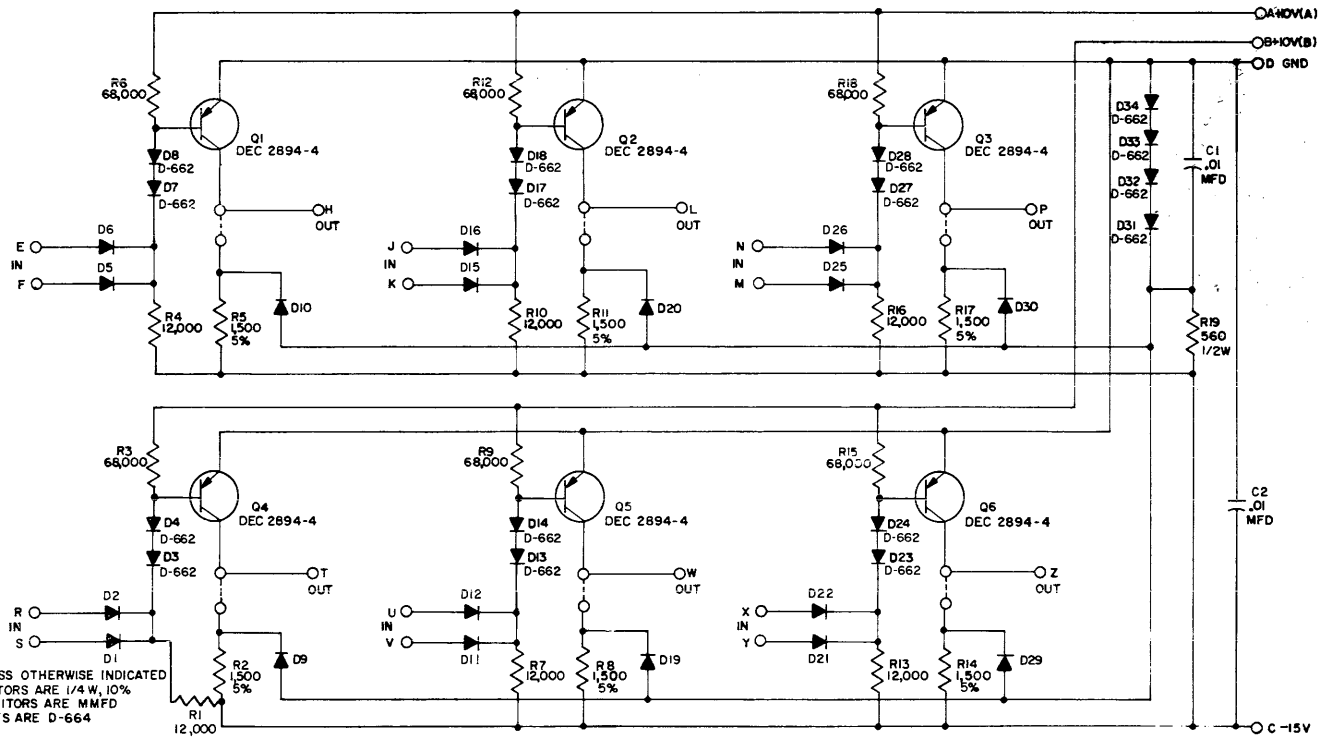
Inverter RS-4102



UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W 10%
CAPACITORS ARE MMFD.
DIODES ARE D-003

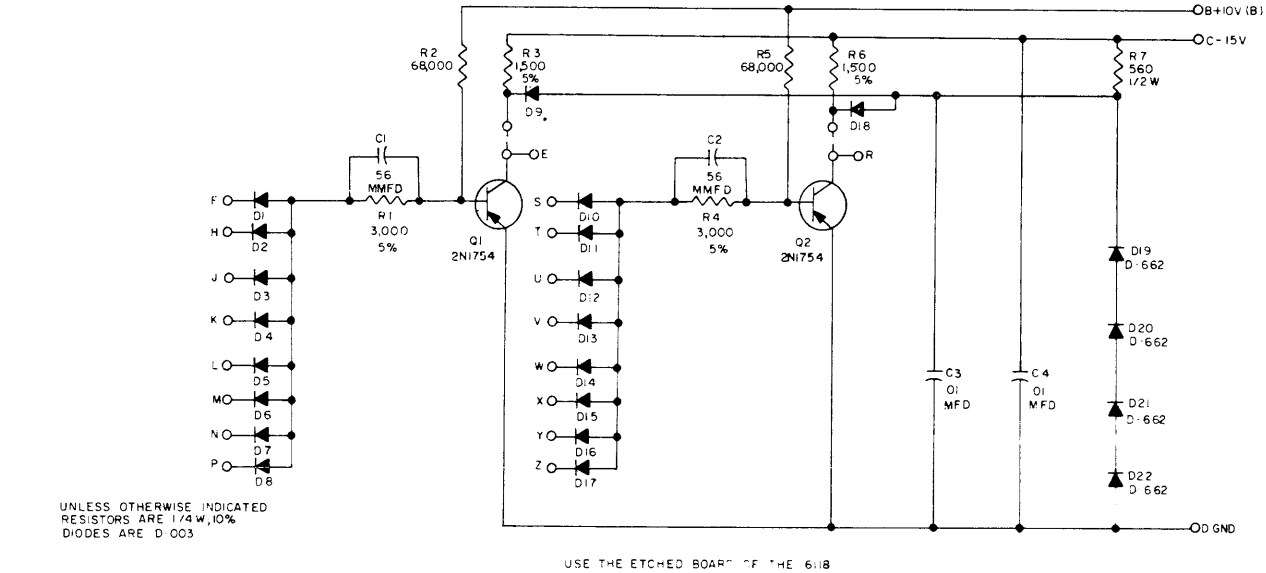
TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
MD94	2N2486		
D-003	1N994		
D-662	1N645		

Diode CS-4112



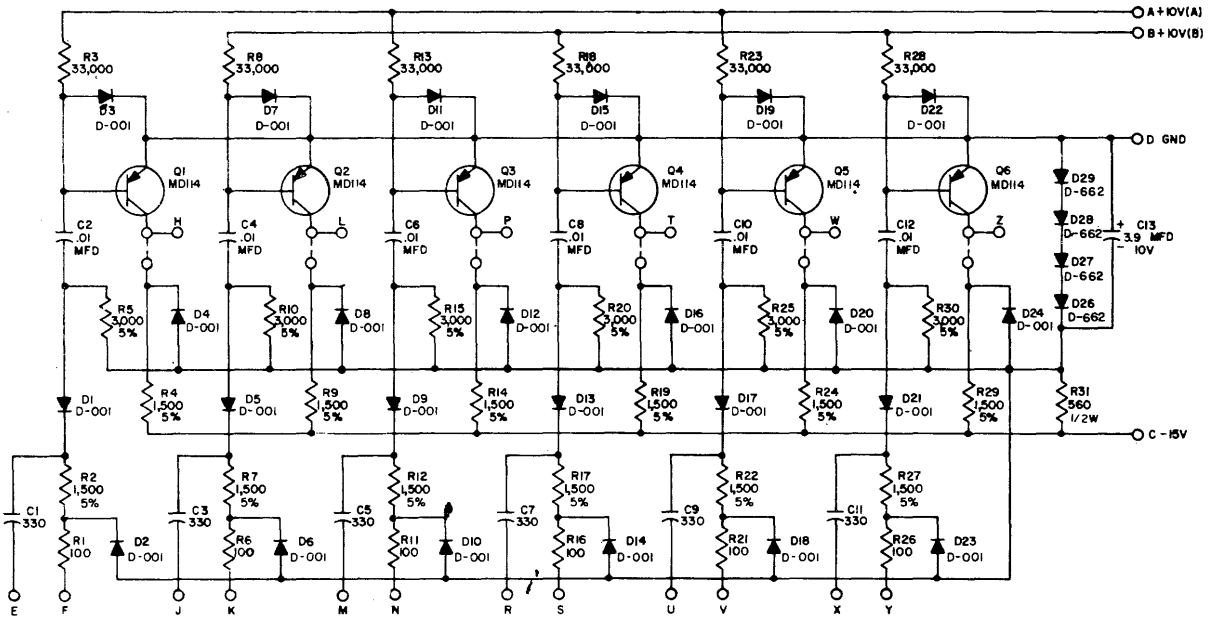
TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-4	DEC 2894		
D-662	IN645		
D-664	IN3606		

Diode RS-4113



TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
D-003	IN994		
D-662	IN645		
2N1754	2N1754		

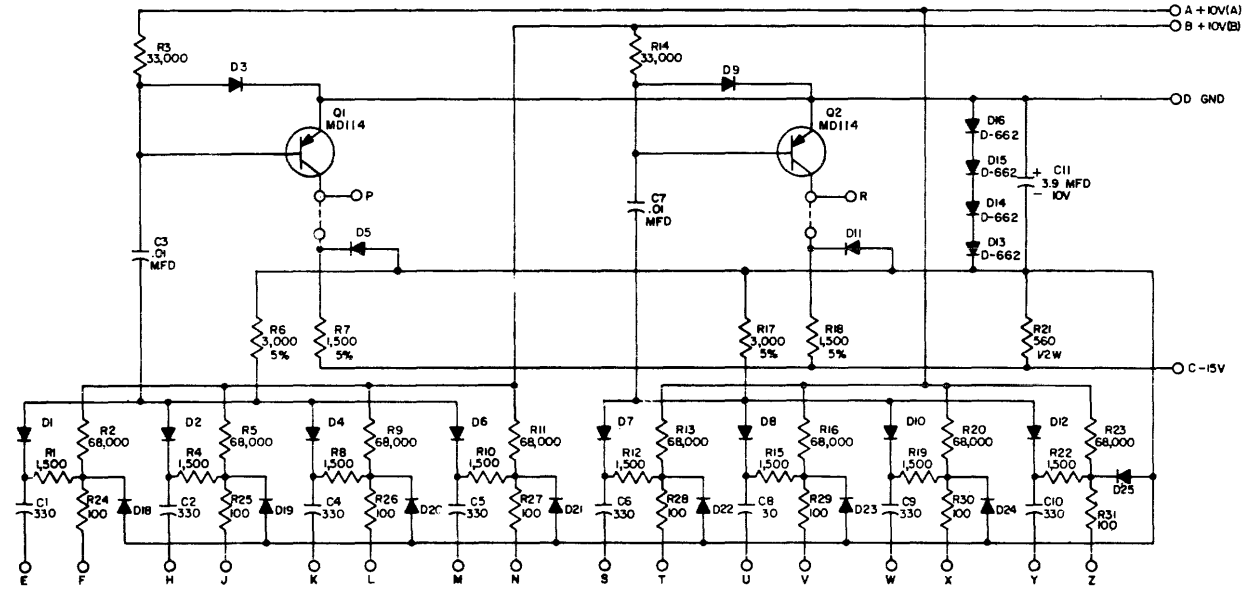
Diode Gate RS-4118



UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W, 10%
CAPACITORS ARE MMFD

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
MD114	2N1499A		
D-001	IN276		
D-662	IN645		

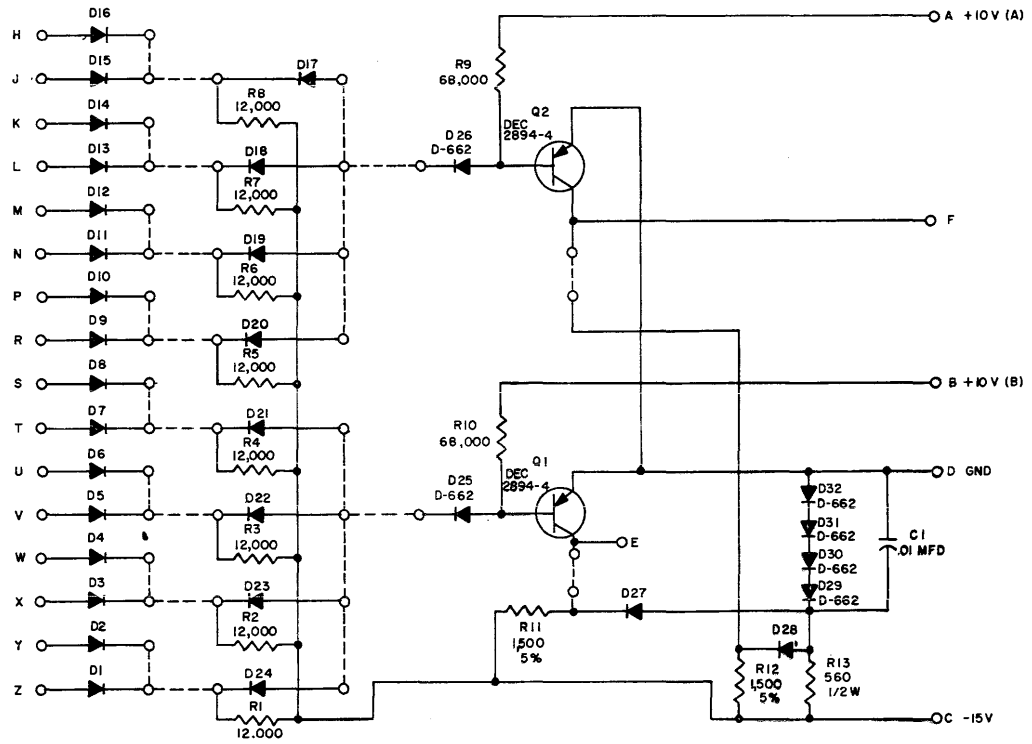
Capacitor Diode Inverter RS-4127



UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W ; 10%
CAPACITORS ARE MMFD
DIODES ARE D-001

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
MD114	2N1499A		
D-001	IN276		
D-662	IN645		

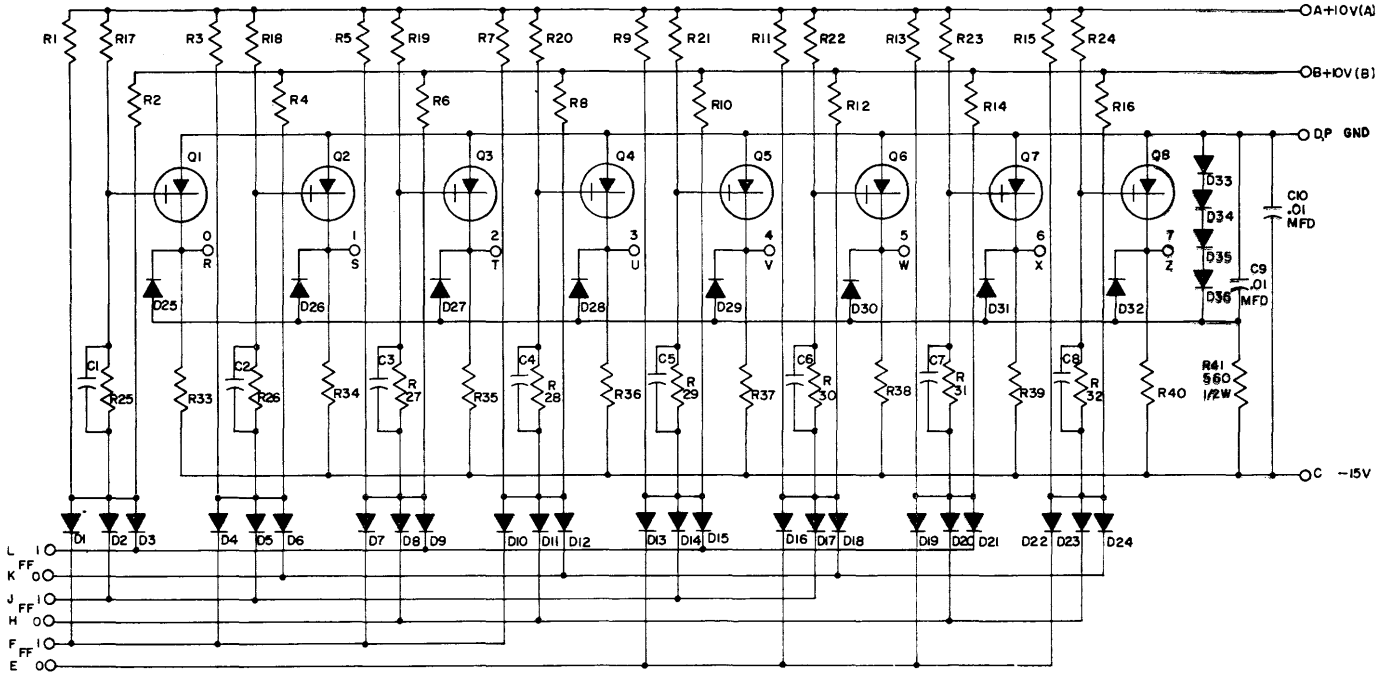
Capacitor Diode Inverter RS-4129



UNLESS OTHERWISE INDICATED
DIODES ARE D-664
RESISTORS ARE 1/4 W, 10%

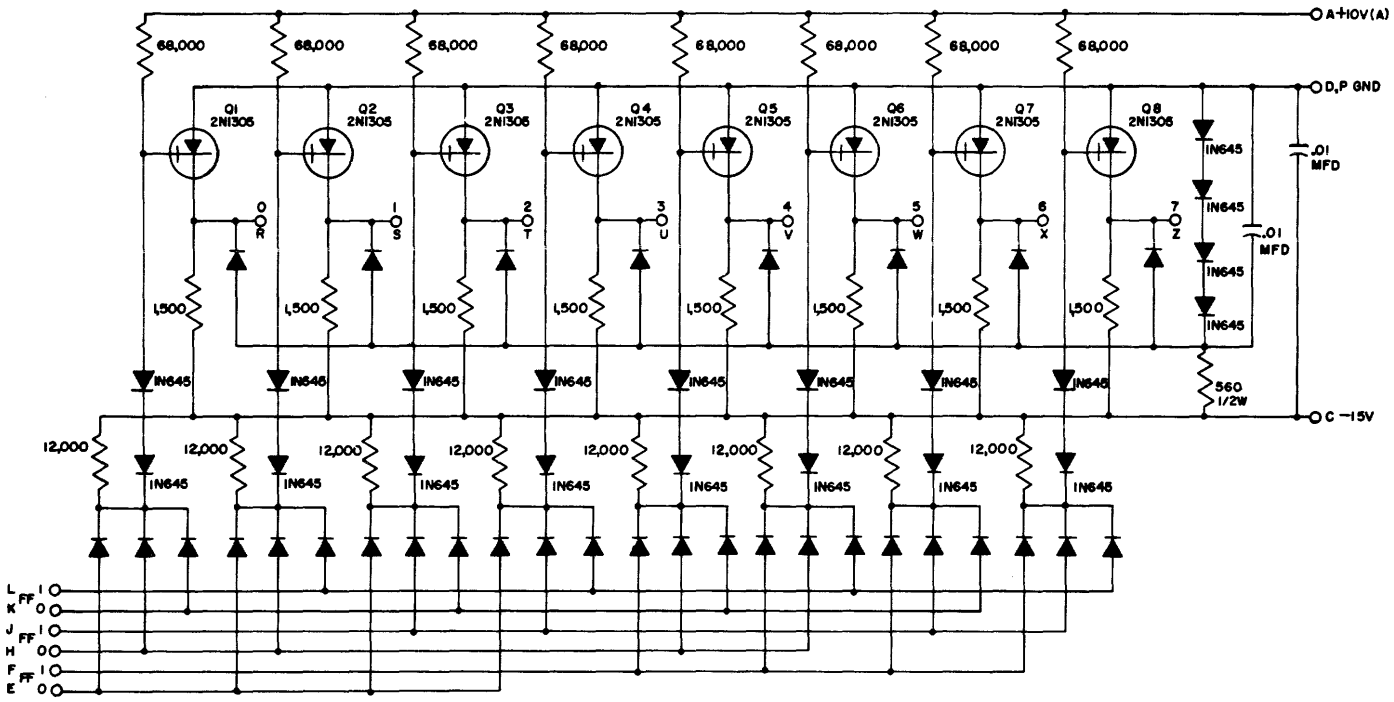
TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-4	DEC 2894		
D-664	IN3606		
D-662	IN645		

Diode Unit RS-4143



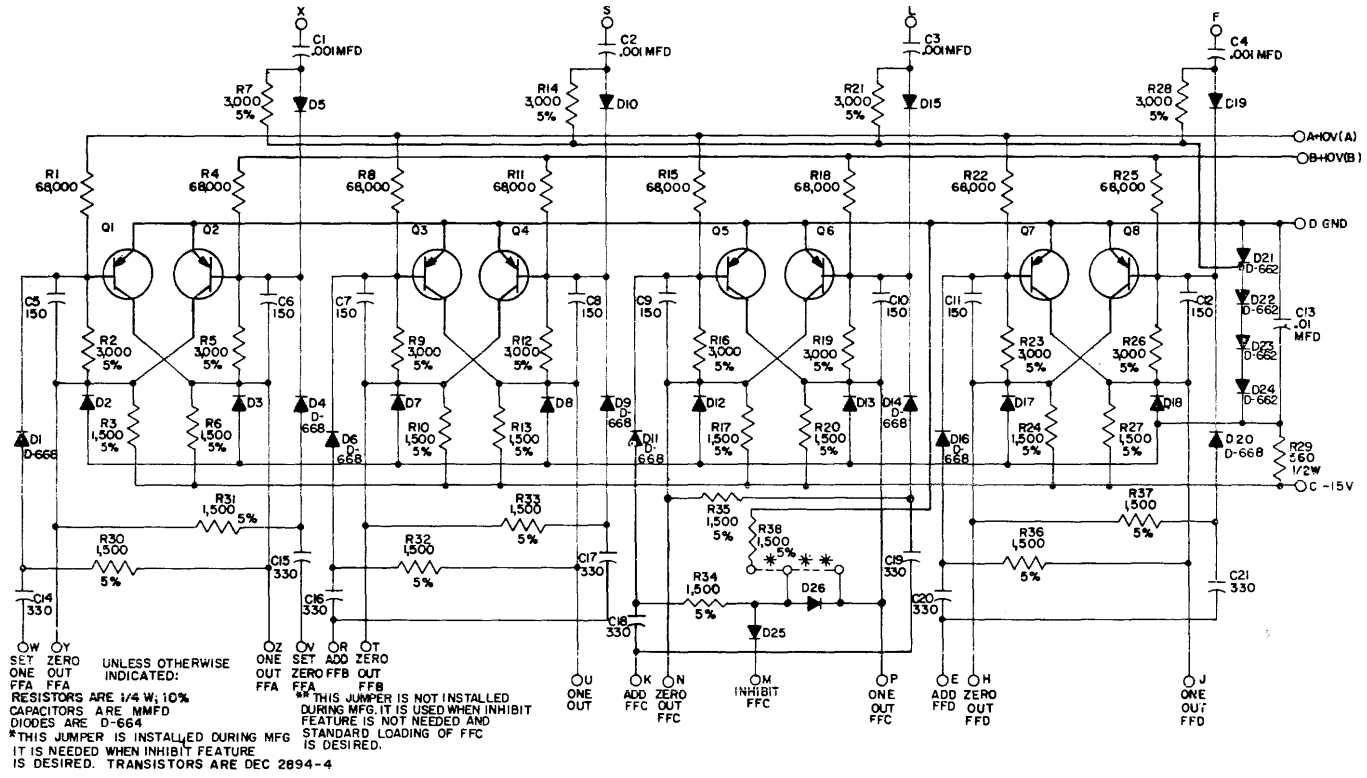
Q1-Q8 ARE 2N1305: D1-D32 ARE IN276
D33-D36 ARE IN645: R1-R16 ARE 27000, 1/4W, 10%
R17-R24 ARE 56,000, 1/4W, 10%: R25-R32 ARE 2,200, 1/4W, 5%
R33-R40 ARE 1,500, 1/4W, 5%: C1-C8 ARE .001 MFD

Binary-to-Octal Decoder RS-4150



UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4 W, 10%
DIODES ARE IN276

Binary-to-Octal Decoder RS-4151



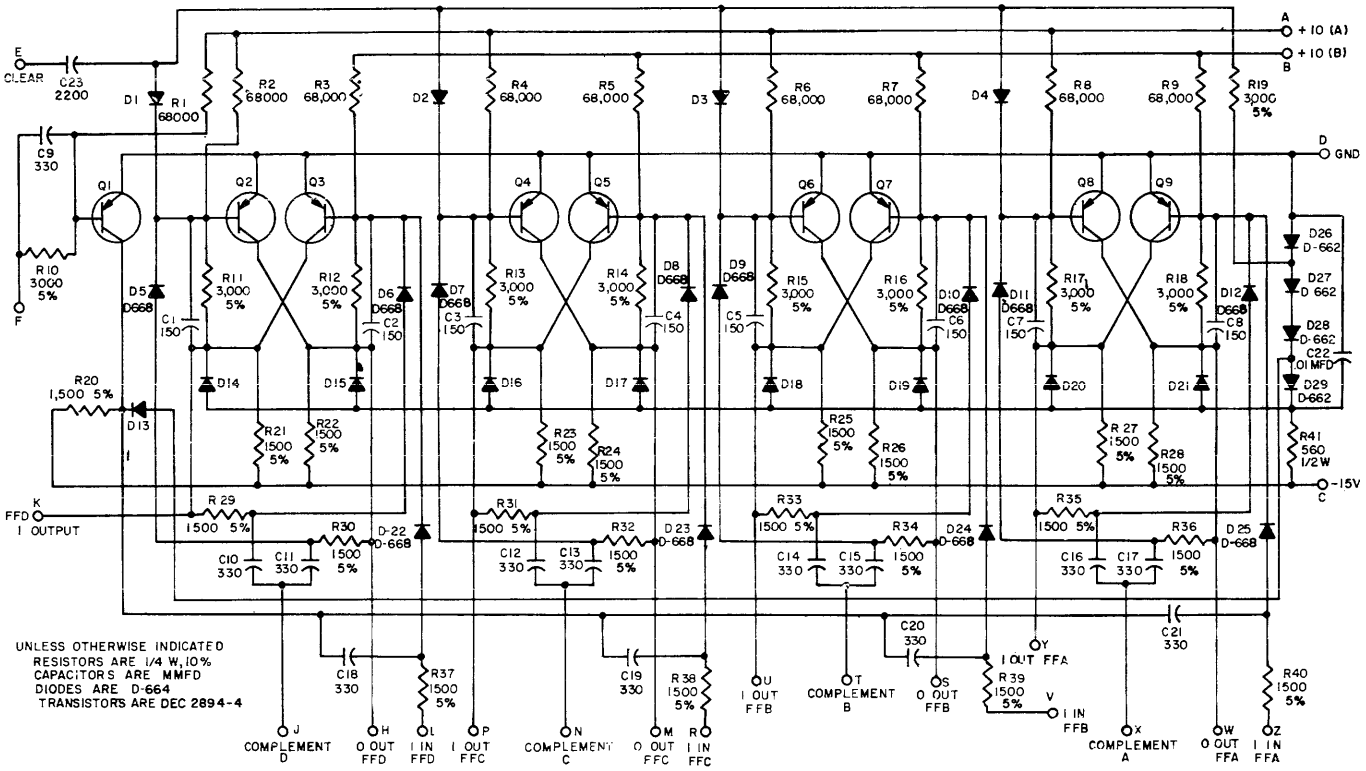
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4 W; 10%
CAPACITORS ARE MMFD
DIODES ARE D-664
*THIS JUMPER IS INSTALLED DURING MFG IT IS NEEDED WHEN INHIBIT FEATURE IS DESIRED. TRANSISTORS ARE DEC 2894-4

**THIS JUMPER IS NOT INSTALLED DURING MFG. IT IS USED WHEN INHIBIT FEATURE IS NOT NEEDED AND STANDARD LOADING OF FFC IS DESIRED.

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-4	DEC 2894		
D-668	D-668*		
D-662	IN645		
D-664	IN3606		

NOTES
* (TWO) IN3606 IN SERIES

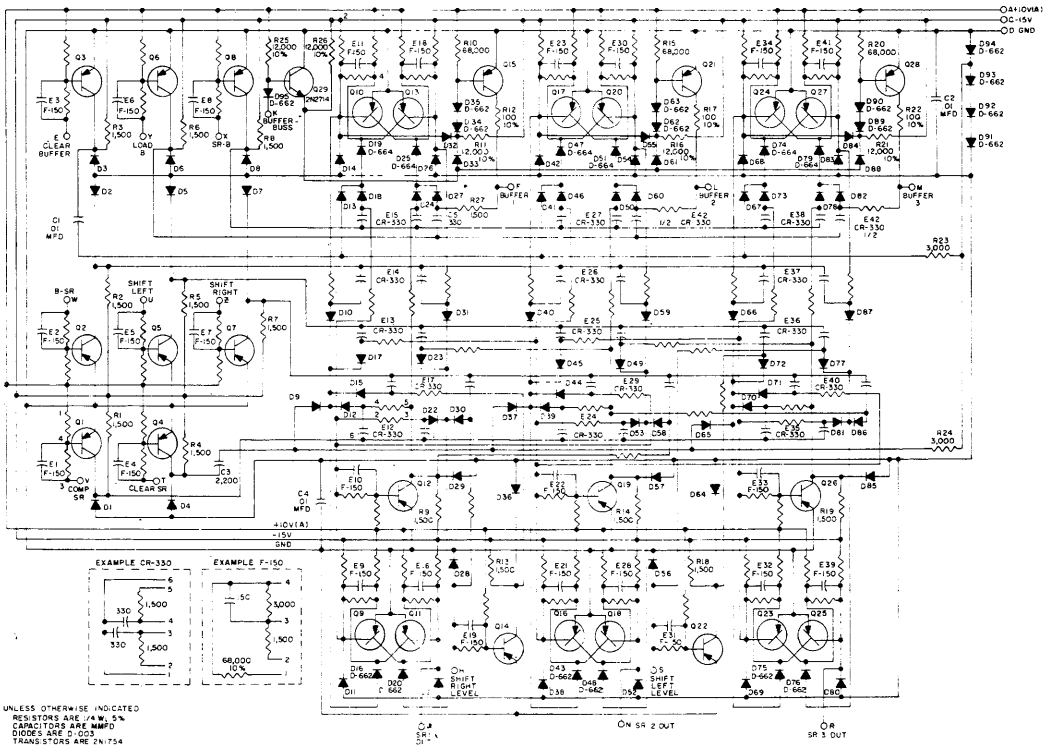
Four-Bit Counter RS-4215



TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-4	DEC 2894		
D-668	D-668*		
D-662	IN645		
D-664	IN3606		

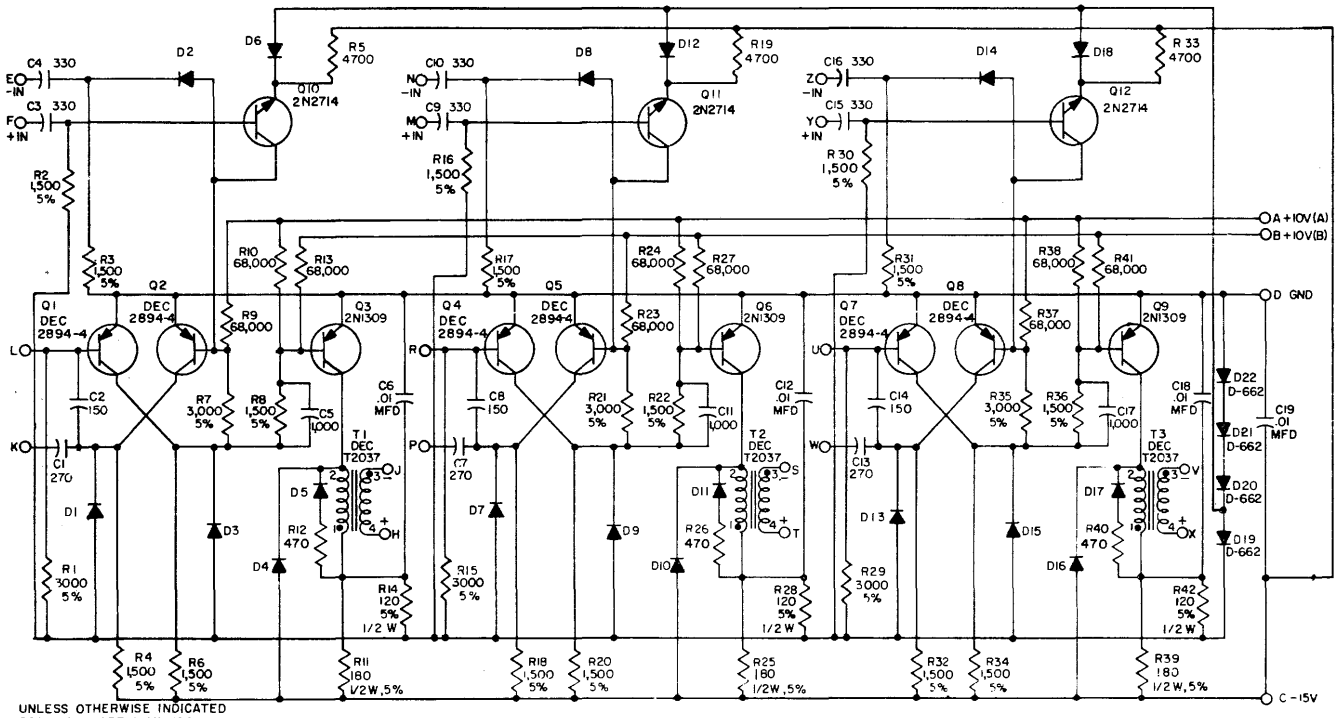
NOTES
*(TWO) IN3606 IN SERIES

Four-Bit Counter RS-4217



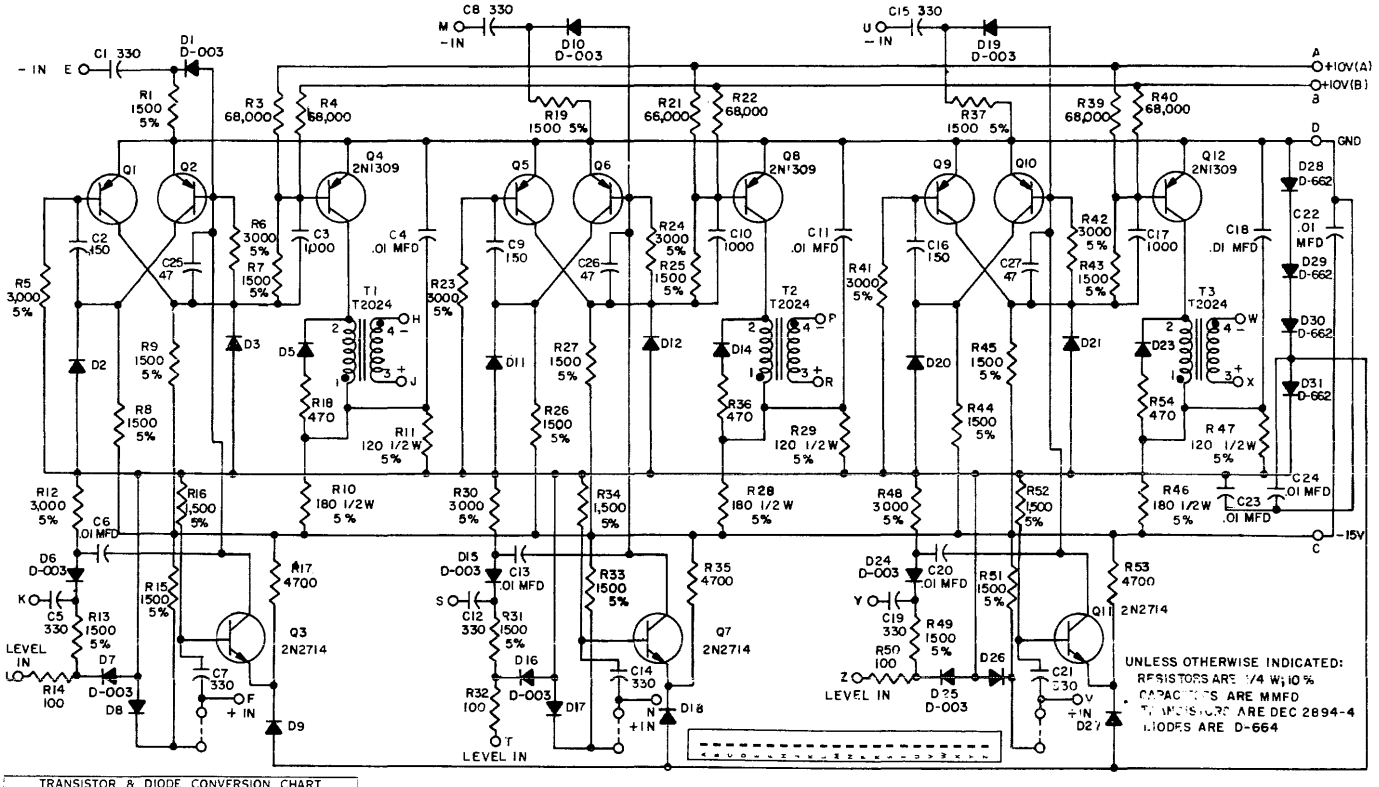
TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
2N1754	2N1754		
D-003	1N994		
D-664	IN3606		
D-662	IN645		
2N2714	2N2714		

Three-Bit Shift Register with Buffer Register RS-4230



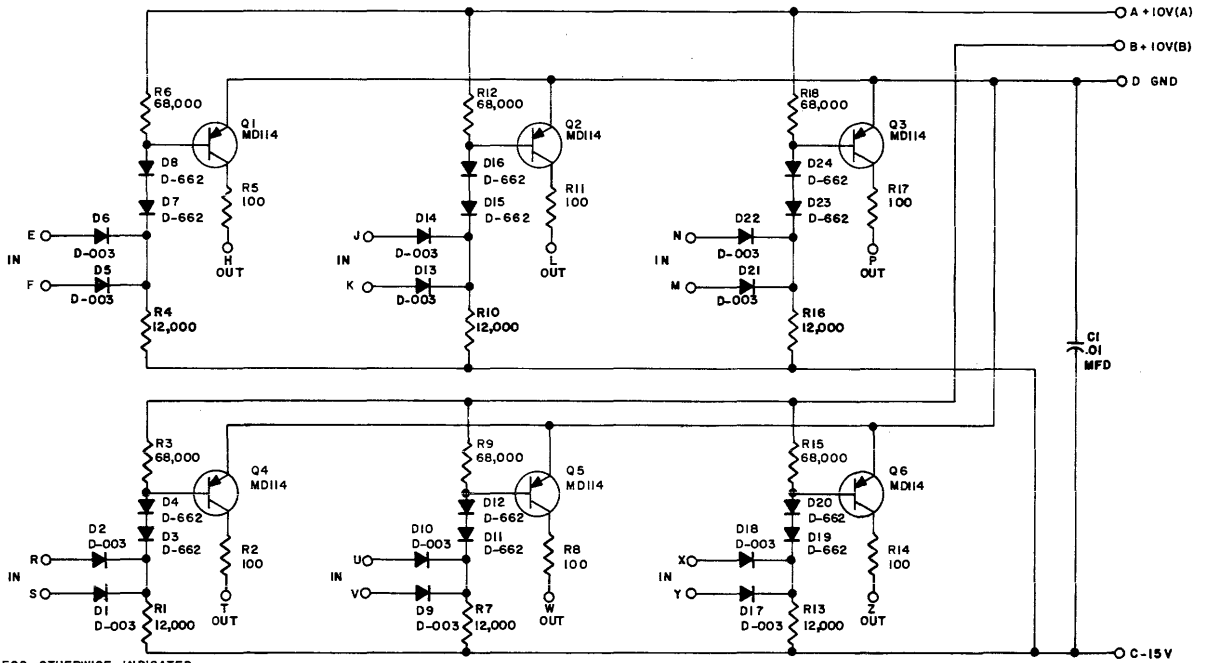
TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
2N2714	2N2714		
DEC 2894-4	DEC 2894		
2N1309	2N1309		
D-664	IN3606		
D-662	IN645		

Pulse Amplifier RS-4604



TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-4	DEC 2894	D-003	IN994
2N2714	2N2714		
2N1309	2N1309		
D-664	IN3606		
D-662	IN645		

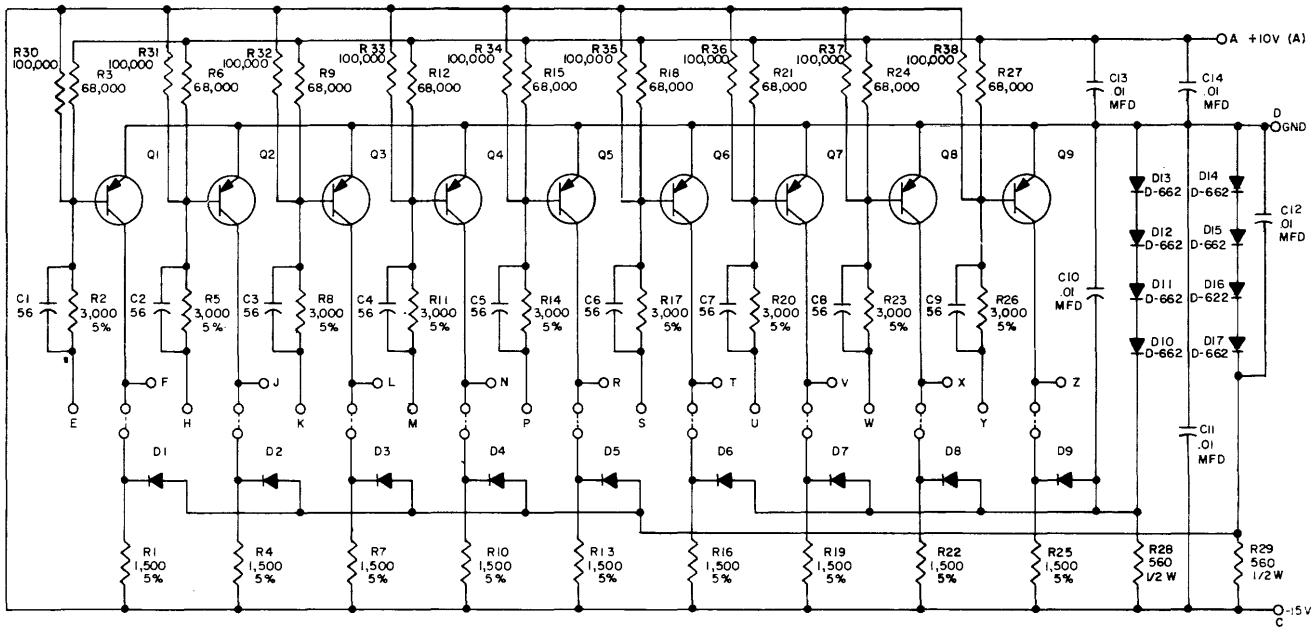
Pulse Amplifier RS-4606



UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W, 10%
CAPACITORS ARE MMFD

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
MD114	2N1499A		
D-003	IN994		
D-662	IN642		

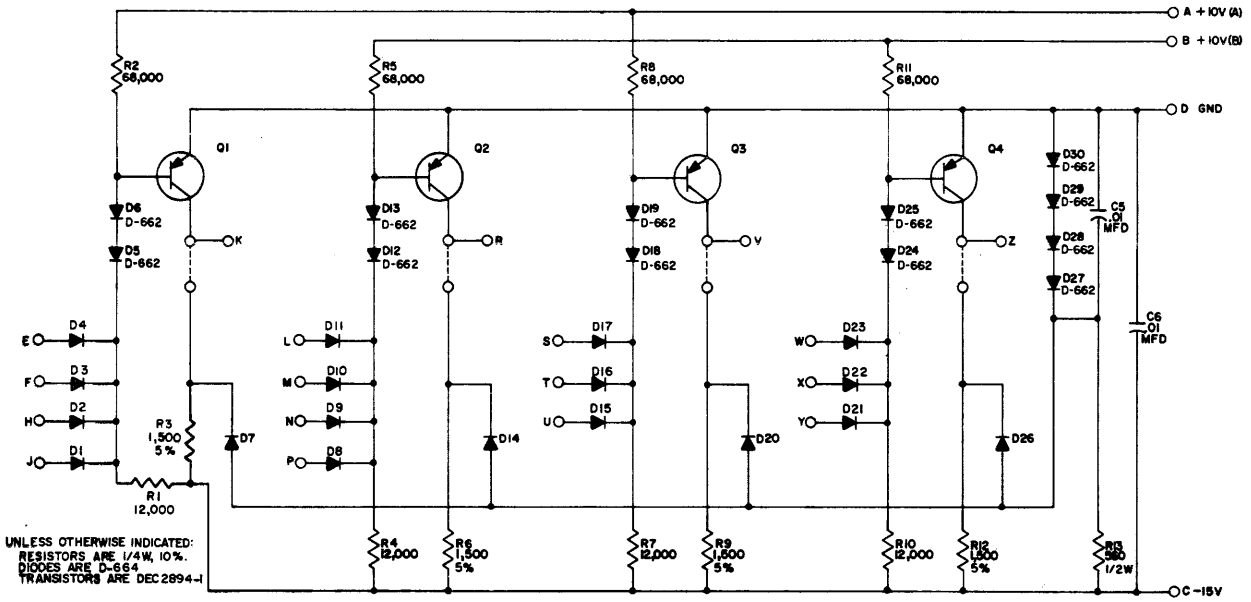
I/O Bus Driver RS-4657



UNLESS OTHERWISE INDICATED
RESISTORS ARE 1/4W, 10%
CAPACITORS ARE MMFD
TRANSISTORS ARE DEC 2894-1
DIODES ARE D-664

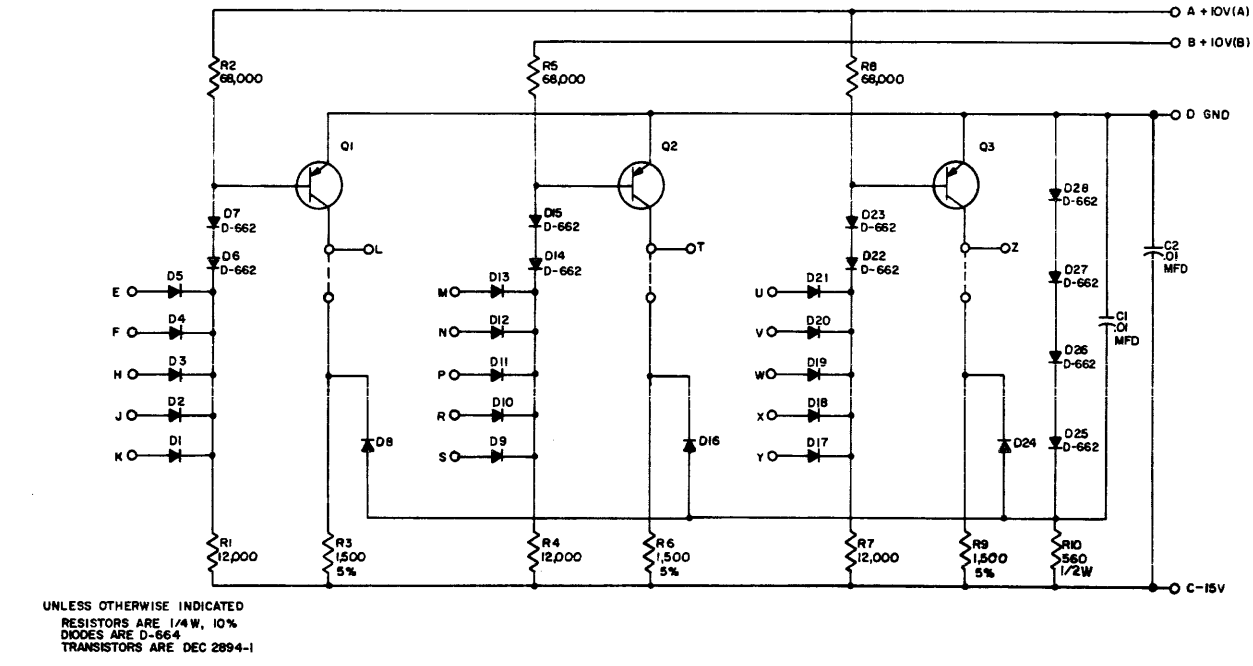
TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-1	2N2894		
D-664	IN914		
D-662	IN645		

Inverter RS-6102



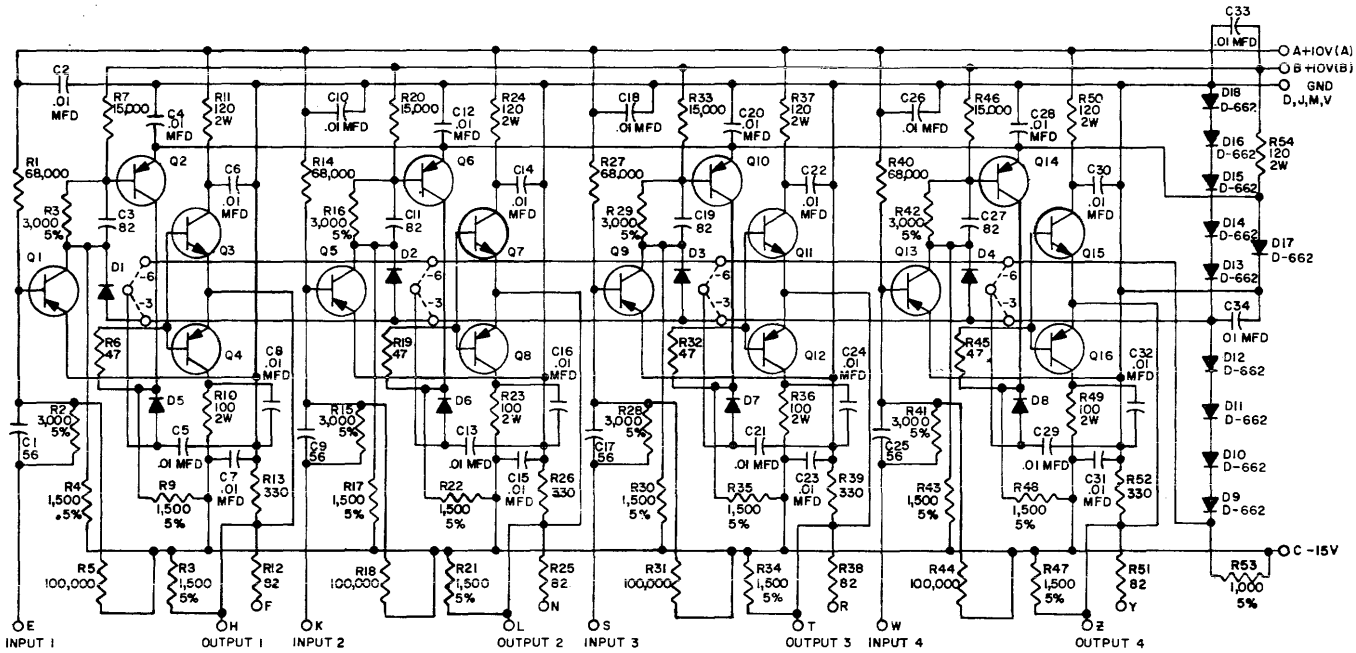
TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-1	DEC 2894		
D-664	IN914		
D-662	IN645		

Diode RS-6115



TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-1	DEC 2894		
D-664	IN914		
D-662	IN645		

Diode RS-6117

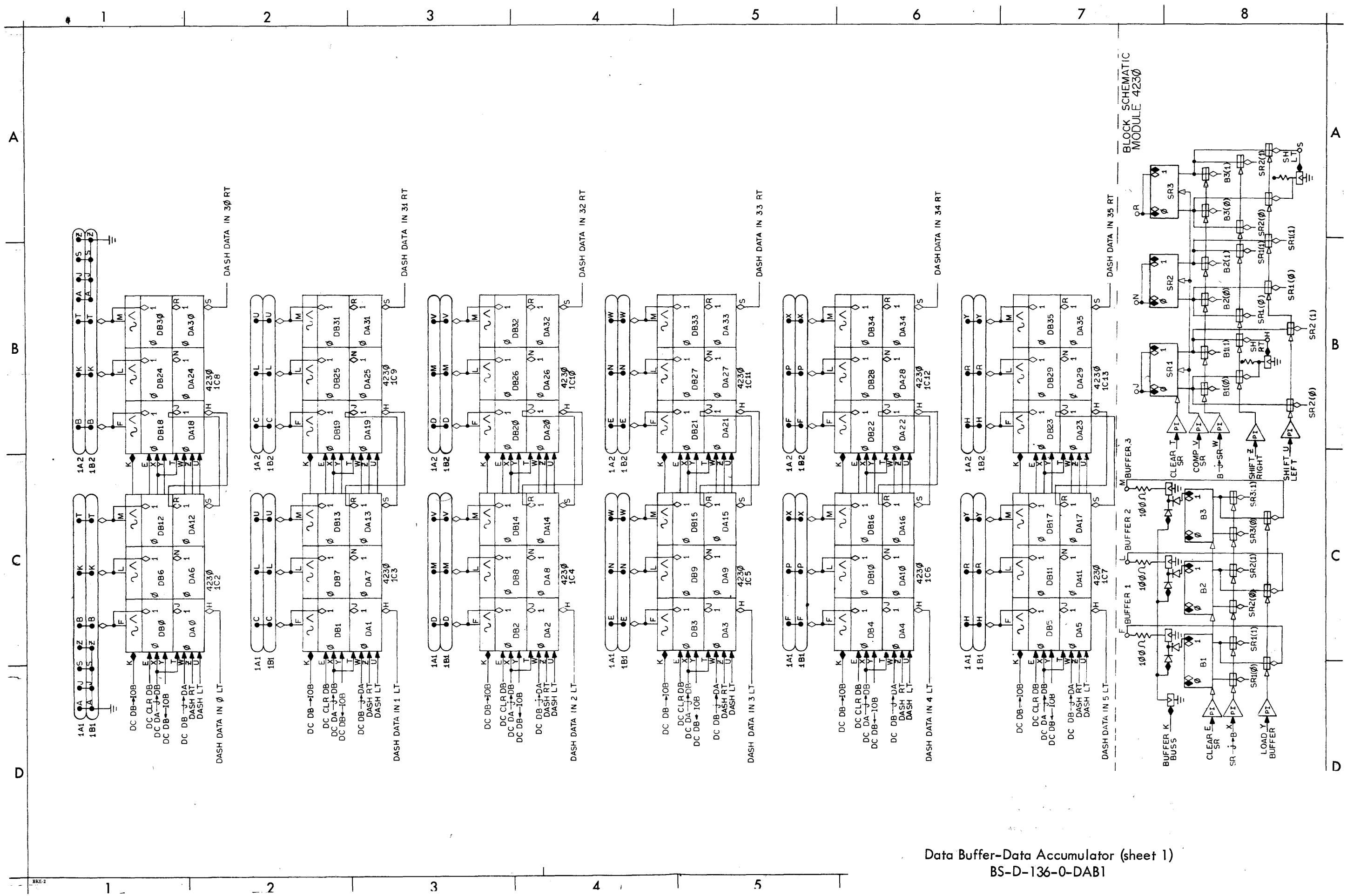


UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%
CAPACITORS ARE MMFD
DIODES ARE D-664
Q3,Q7,Q11,& Q15 ARE 2N3009
Q4,Q8,Q12,& Q16 ARE 2N2904
ALL OTHER TRANSISTORS ARE DEC 2894-1

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-1	DEC 2894		
2N2904	2N2904		
2N3009	2N3C09		
D-664	IN3606		
D-662	IN645		

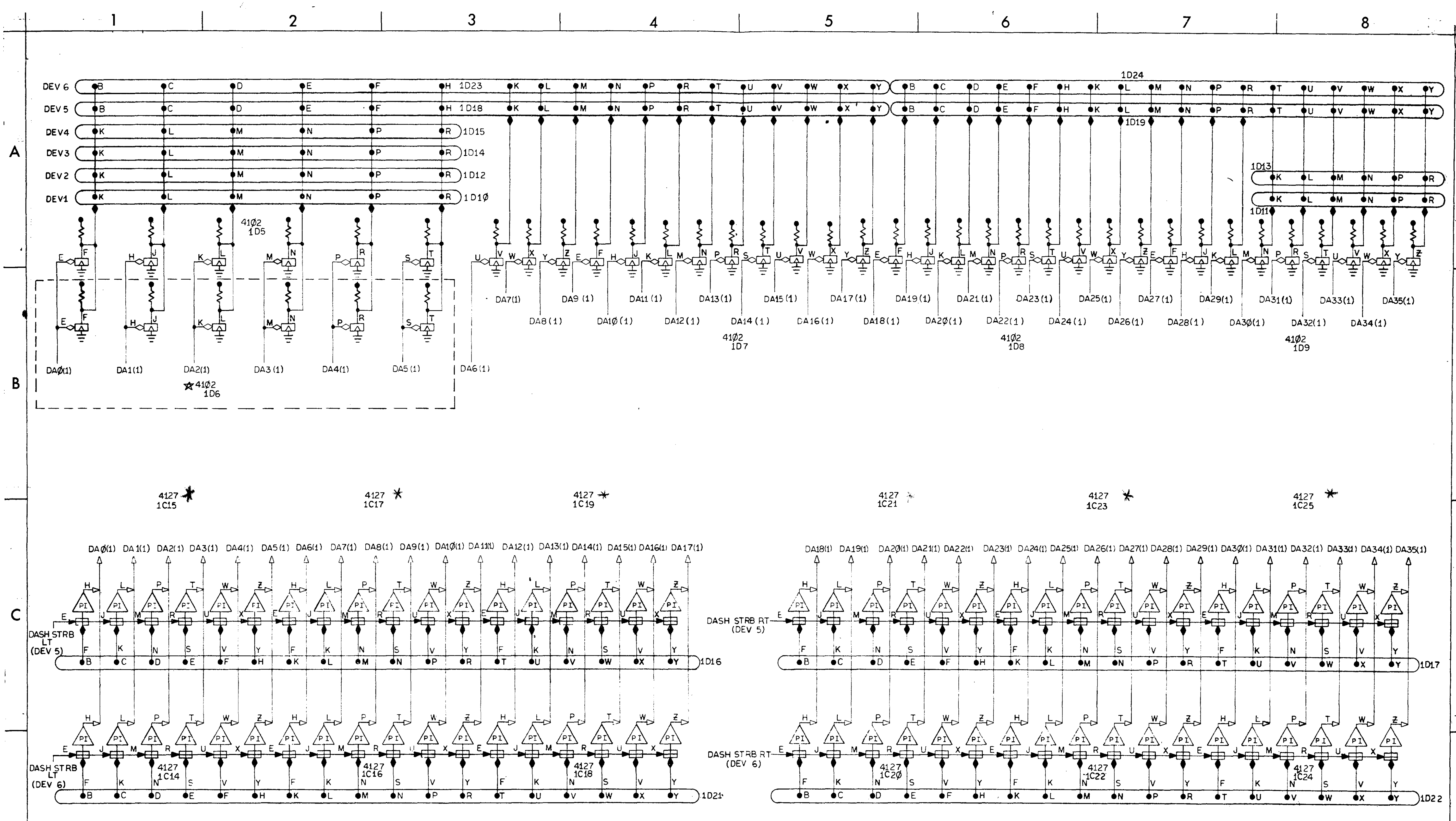
Bus Driver RS-6684

Data Buffer-Data Accumulator (sheet 1)
BS-D-136-0-DAB1



Data Buffer-Data Accumulator (sheet 1)
BS-D-136-0-DAB1

Data Buffer-Data Accumulator (sheet 2)
BS-D-136-0-DAB2

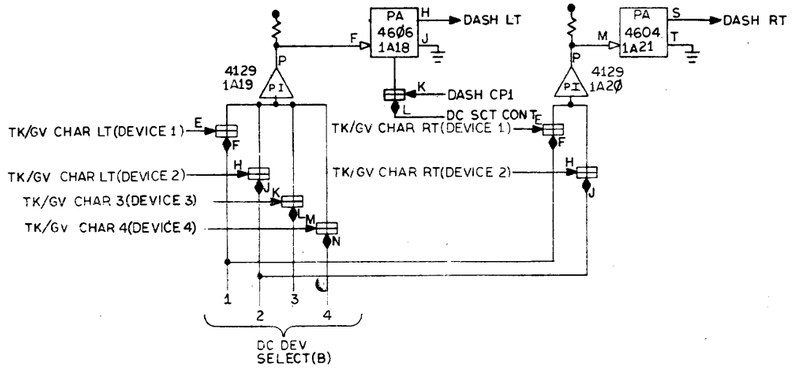


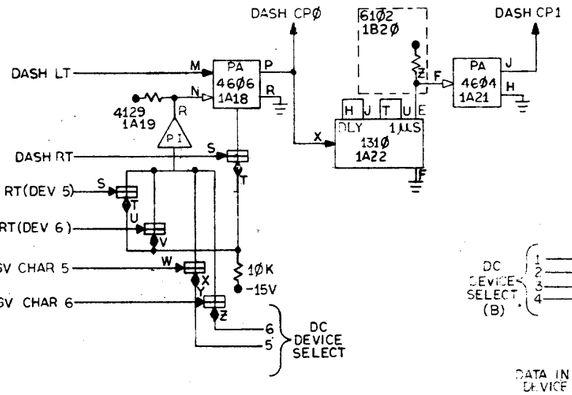
NOTES

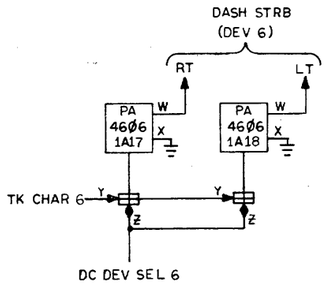
1. ☆ PKG USED WITH FOUR OR MORE DEVICES CONNECTED
2. 1D10 THRU 1D25 GROUNDS ARE A,J,S, AND Z.
3. ✱ Pkg must be changed To [illegible] when [illegible] To [illegible] Disc File

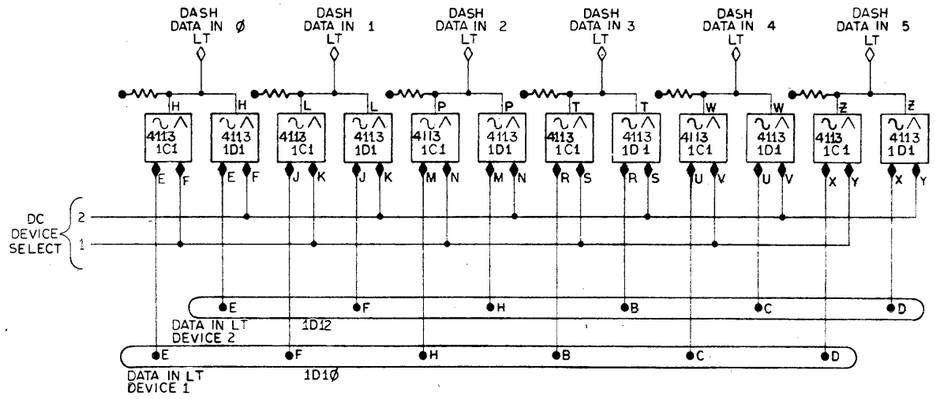
Data Buffer-Data Accumulator (sheet 2)
BS-D-136-0-DAB2

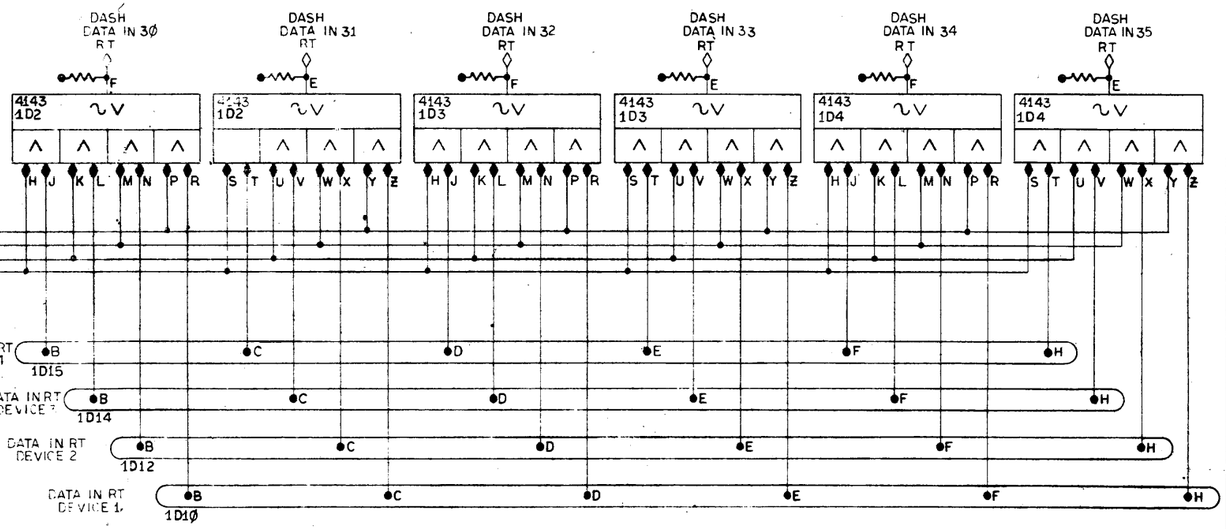
Data Accumulator Shift BS-D-136-0-DASH

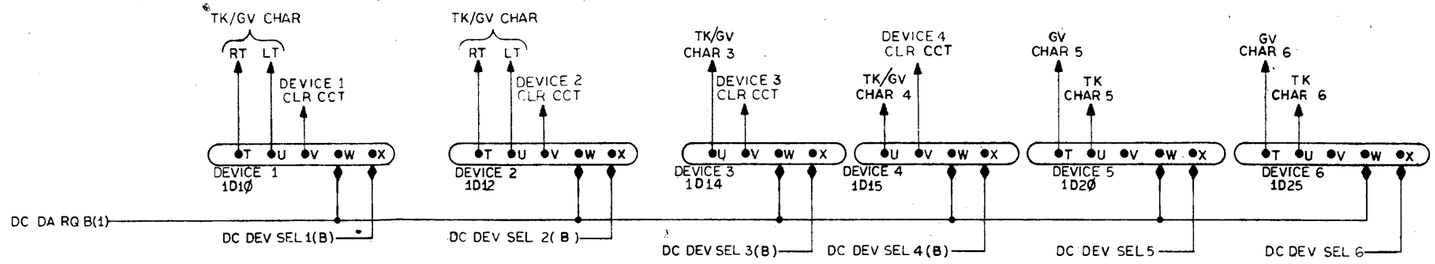






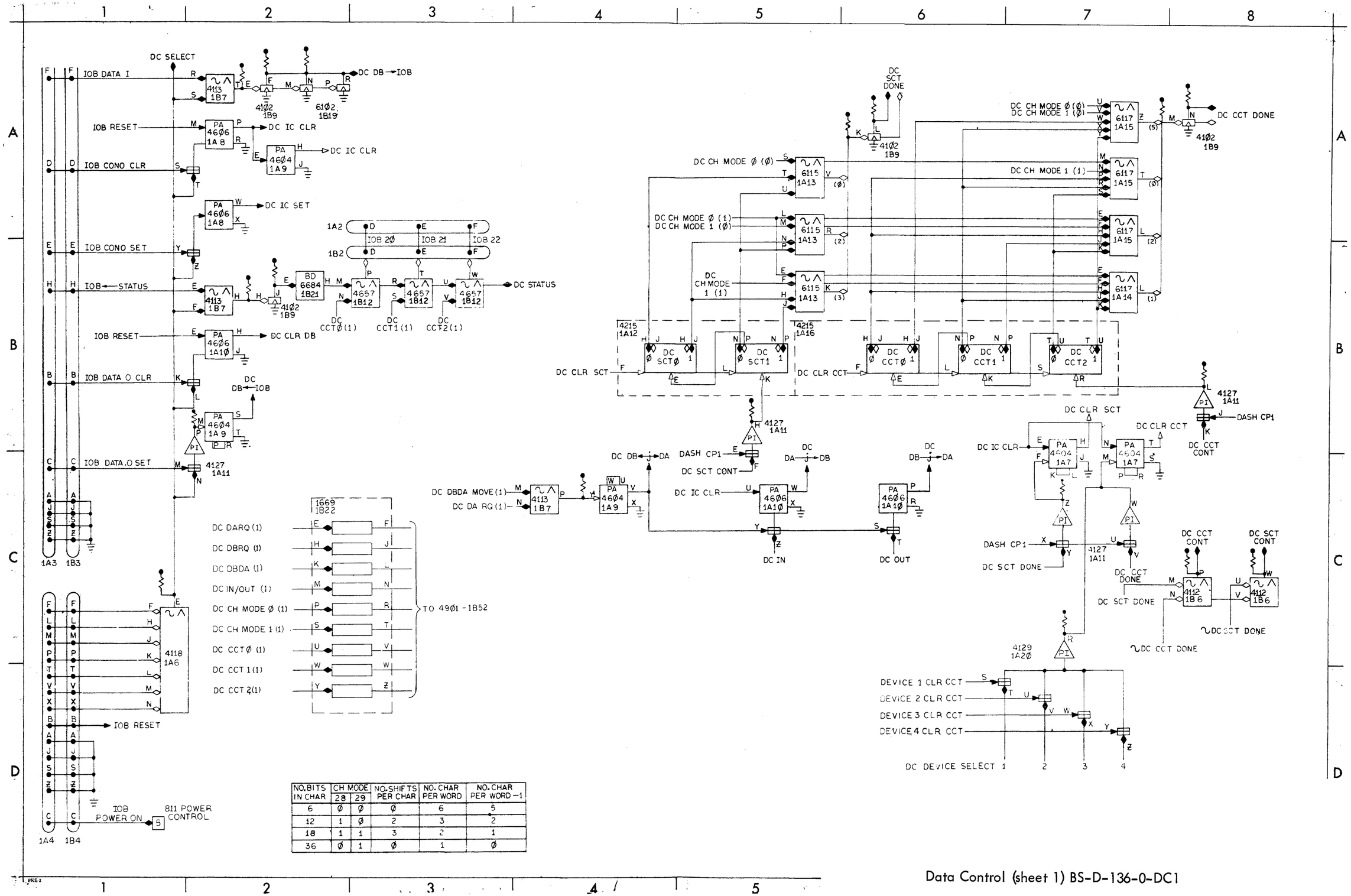






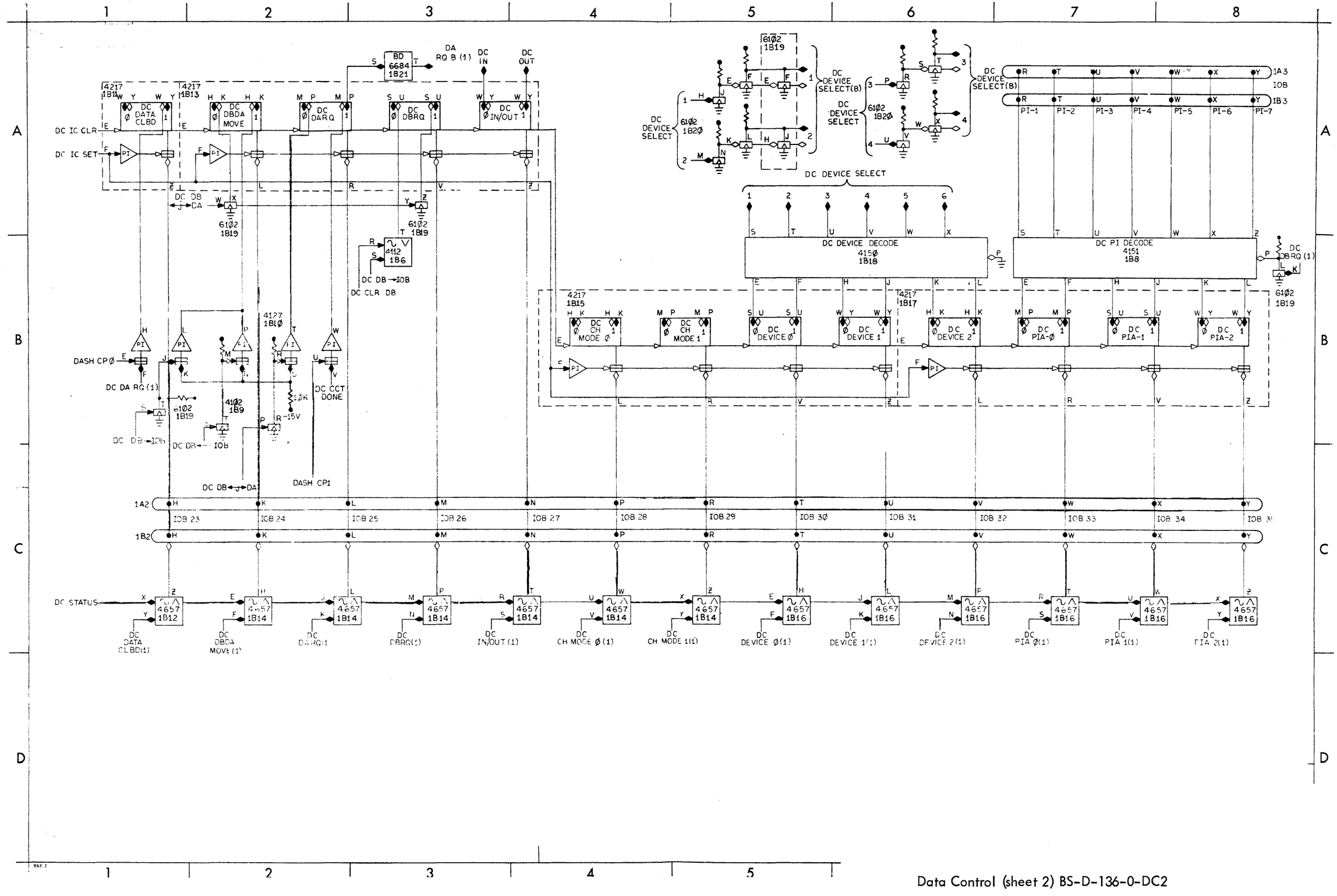
Data Accumulator Shift BS-D-136-0-DASH

Data Control (sheet 1) BS-D-136-0-DC1



NO. BITS IN CHAR	CH MODE 28	CH MODE 29	NO. SHIFTS PER CHAR	NO. CHAR PER WORD	NO. CHAR PER WORD −1
6	Ø	Ø	Ø	6	5
12	1	Ø	2	3	2
18	1	1	3	2	1
36	Ø	1	Ø	1	Ø

Data Control (sheet 1) BS-D-136-0-DC1

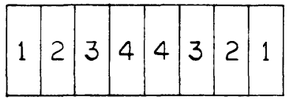


Data Control (sheet 2) BS-D-136-0-DC2

Methode Plug Connections at
Processor Interface CD-D-166-0-IOMB

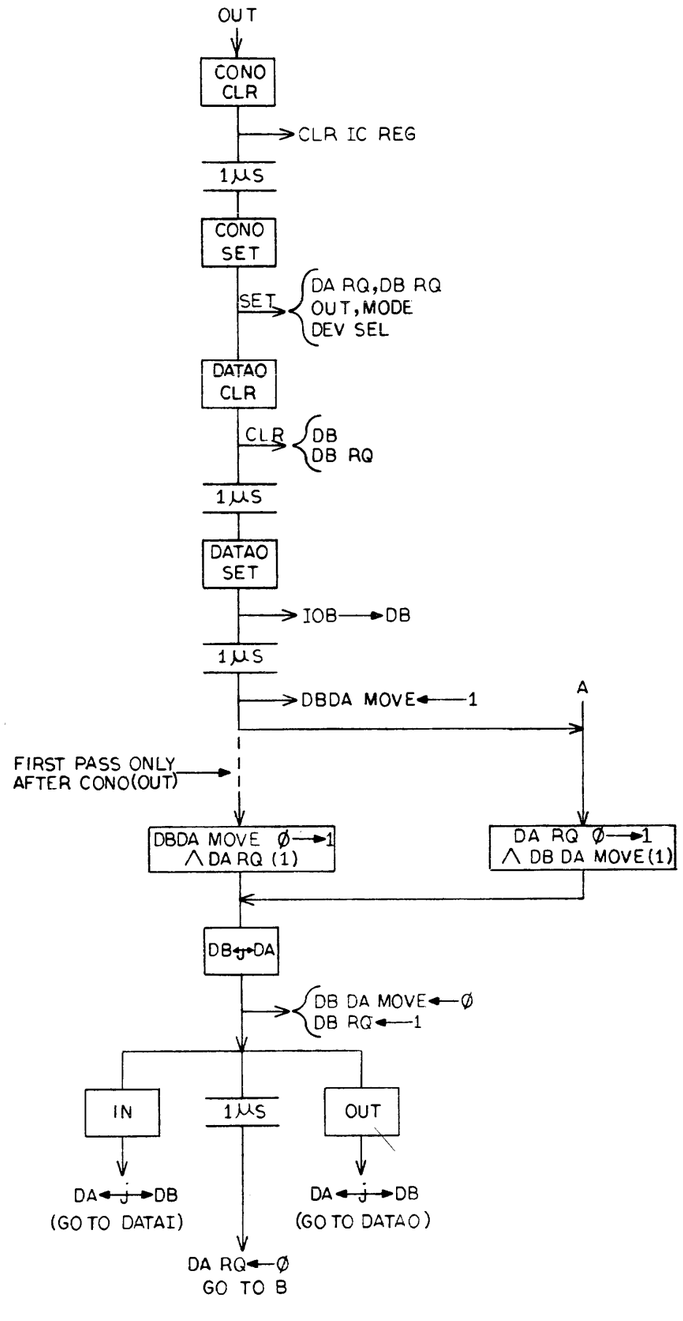
I.O. BUS

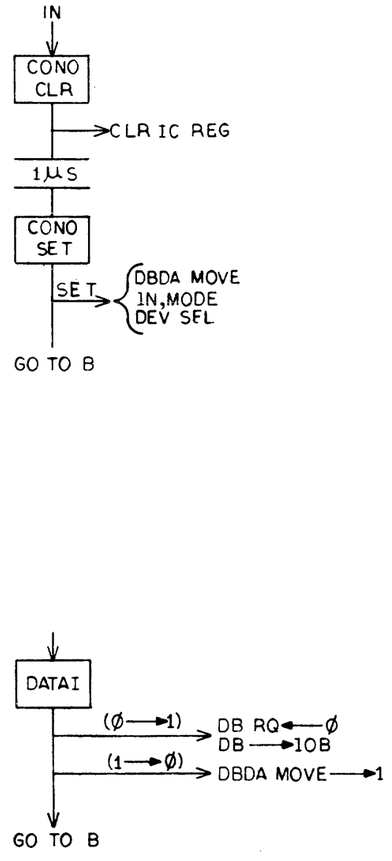
PIN	I.O. CABLE #1	I.O. CABLE #2	I.O. CABLE #3	I.O. CABLE #4
A	GND	GND	GND	GND
B	IOB Ø (1) ⟷	IOB 18 (1) ⟷	POWER RESET →	DATAO CLEAR →
C	IOB 1 (1) ⟷	IOB 19 (1) ⟷	POWER ON -15V	DATAO SET →
D	IOB 2 (1) ⟷	IOB 2Ø (1) ⟷		CONO CLEAR →
E	IOB 3 (1) ⟷	IOB 21 (1) ⟷		CONO SET →
F	IOB 4 (1) ⟷	IOB 22 (1) ⟷	IOS 3 (Ø) ⟷	IOB ← DATAI →
H	IOB 5 (1) ⟷	IOB 23 (1) ⟷	IOS 3 (1) ⟷	IOB ← STATUS →
J	GND	GND	GND	GND
K	IOB 6 (1) ⟷	IOB 24 (1) ⟷	IOS 4 (Ø) ⟷	
L	IOB 7 (1) ⟷	IOB 25 (1) ⟷	IOS 4 (1) ⟷	
M	IOB 8 (1) ⟷	IOB 26 (1) ⟷	IOS 5 (Ø) ⟷	
N	IOB 9 (1) ⟷	IOB 27 (1) ⟷	IOS 5 (1) ⟷	
P	IOB 1Ø (1) ⟷	IOB 28 (1) ⟷	IOS 6 (Ø) ⟷	
R	IOB 11 (1) ⟷	IOB 29 (1) ⟷	IOS 6 (1) ⟷	PI REQ 1 ⟷
S	GND	GND	GND	GND
T	IOB 12 (1) ⟷	IOB 3Ø (1) ⟷	IOS 7 (Ø) ⟷	PI REQ 2 ⟷
U	IOB 13 (1) ⟷	IOB 31 (1) ⟷	IOS 7 (1) ⟷	PI REQ 3 ⟷
V	IOB 14 (1) ⟷	IOB 32 (1) ⟷	IOS 8 (Ø) ⟷	PI REQ 4 ⟷
W	IOB 15 (1) ⟷	IOB 33 (1) ⟷	IOS 8 (1) ⟷	PI REQ 5 ⟷
X	IOB 16 (1) ⟷	IOB 34 (1) ⟷	IOS 9 (Ø) ⟷	PI REQ 6 ⟷
Y	IOB 17 (1) ⟷	IOB 35 (1) ⟷	IOS 9 (1) ⟷	PI REQ 7 ⟷
Z	GND	GND	GND	GND
	(SOURCE 3L6)	(SOURCE 3L7)	(SOURCE 2L13)	(SOURCE 2N25)
PLUG NO.	1A1 — 1B1	1A2 — 1B2	1A4 — 1B4	1A3 — 1B3

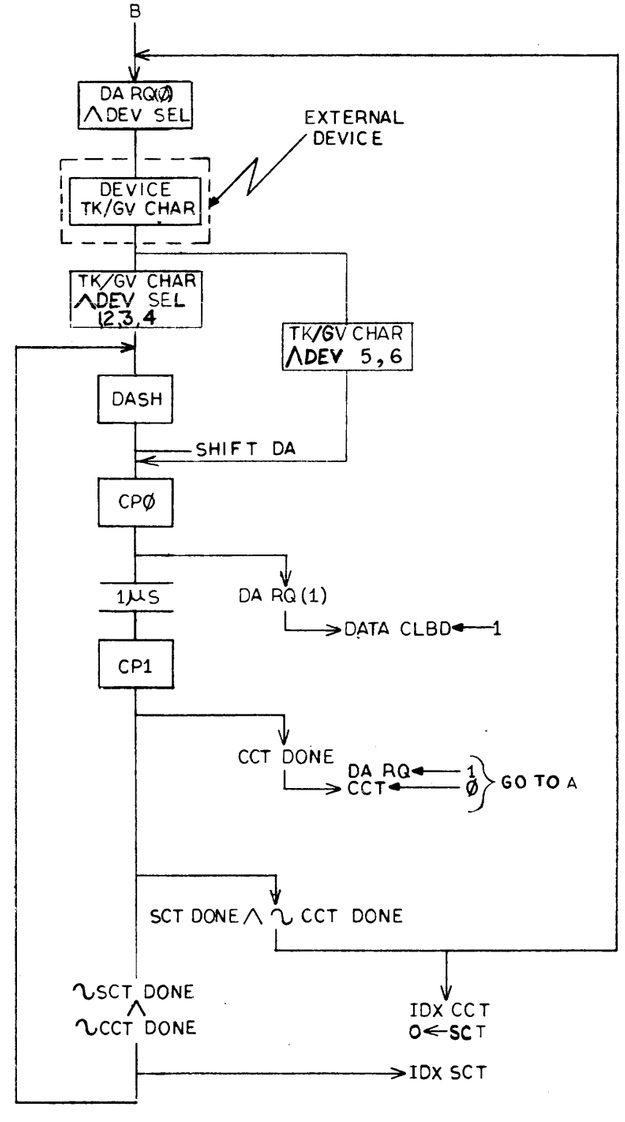


Methode Plug Connections at
Processor Interface CD-D-166-0-IOMB

Flow Diagram FD-D-136-0-FD1







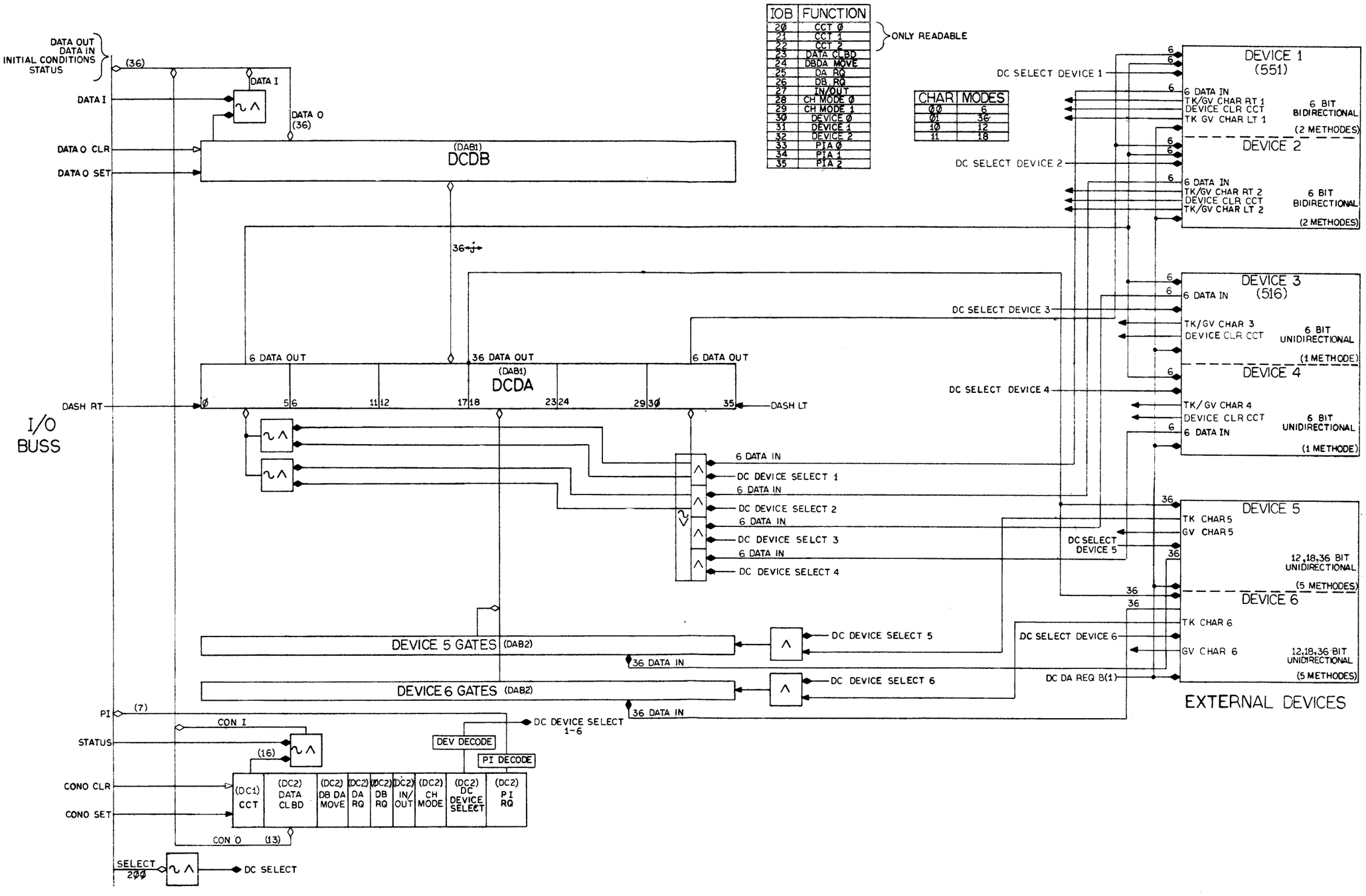
Flow Diagram FD-D-136-0-FD1

* PIN	DEVICE 1 MT 551		DEVICE 2		DEVICE 3 MT 516	DEVICE 4	DEVICE 5 36 OR LESS BIT DEVICE					DEVICE 6 36 OR LESS BIT DEVICE				
	1E10	1E11	1E12	1E13	1E14	1E15	1E16	1E17	1E18	1E19	1E20	1E21	1E22	1E23	1E24	1E25
A	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND
B	30 IN		30 IN		30 IN	30 IN	0 IN	18 IN	0 OUT	18 OUT		0 IN	18 IN	0 OUT	18 OUT	
C	31 IN		31 IN		31 IN	31 IN	1 IN	19 IN	1 OUT	19 OUT		1 IN	19 IN	1 OUT	19 OUT	
D	32 IN	SPARE	32 IN	SPARE	32 IN	32 IN	2 IN	20 IN	2 OUT	20 OUT	SPARE	2 IN	20 IN	2 OUT	20 OUT	SPARE
E	33 IN		33 IN		33 IN	33 IN	3 IN	21 IN	3 OUT	21 OUT		3 IN	21 IN	3 OUT	21 OUT	
F	34 IN		34 IN		34 IN	34 IN	4 IN	22 IN	4 OUT	22 OUT		4 IN	22 IN	4 OUT	22 OUT	
H	35 IN		35 IN		35 IN	35 IN	5 IN	23 IN	5 OUT	23 OUT		5 IN	23 IN	5 OUT	23 OUT	
J	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND
K	0 OUT	30 OUT	0 OUT	30 OUT	0 OUT	0 OUT	6 IN	24 IN	6 OUT	24 OUT		6 IN	24 IN	6 OUT	24 OUT	
L	1 OUT	31 OUT	1 OUT	31 OUT	1 OUT	1 OUT	7 IN	25 IN	7 OUT	25 OUT		7 IN	25 IN	7 OUT	25 OUT	
M	2 OUT	32 OUT	2 OUT	32 OUT	2 OUT	2 OUT	8 IN	26 IN	8 OUT	26 OUT	SPARE	8 IN	26 IN	8 OUT	26 OUT	SPARE
N	3 OUT	33 OUT	3 OUT	33 OUT	3 OUT	3 OUT	9 IN	27 IN	9 OUT	27 OUT		9 IN	27 IN	9 OUT	27 OUT	
P	4 OUT	34 OUT	4 OUT	34 OUT	4 OUT	4 OUT	10 IN	28 IN	10 OUT	28 OUT		10 IN	28 IN	10 OUT	28 OUT	
R	5 OUT	35 OUT	5 OUT	35 OUT	5 OUT	5 OUT	11 IN	29 IN	11 OUT	29 OUT		11 IN	29 IN	11 OUT	29 OUT	
S	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND
T	TK/GV RT		TK/GV RT		SPARE	SPARE	12 IN	30 IN	12 OUT	30 OUT	GV/CHAR	12 IN	30 IN	12 OUT	30 OUT	GV/CHAR
U	TK/GV LT		TK/GV LT		TK/GY 3	TK/GV 4	13 IN	31 IN	13 OUT	31 OUT	TK/CHAR	13 IN	31 IN	13 OUT	31 OUT	TK/CHAR
V	CLR CCT 1		CLR CCT 2		CLR CCT 3	CLR CCT 4	14 IN	32 IN	14 OUT	32 OUT		14 IN	32 IN	14 OUT	32 OUT	
W	DARQ(1)-	SPARE	DARQ(1)-	SPARE	DARQ(1)-	DARQ(1)-	15 IN	33 IN	15 OUT	33 OUT	DARQ(1)-	15 IN	33 IN	15 OUT	33 OUT	DARQ(1)-
X	SEL 1 -		SEL 2 -		SEL 3 -	SEL 4-	16 IN	34 IN	16 OUT	34 OUT	SEL 5-	16 IN	34 IN	16 OUT	34 OUT	SEL 6
Y	SPARE		SPARE		SPARE	SPARE	17 IN	35 IN	17 OUT	35 OUT	SPARE	17 IN	35 IN	17 OUT	35 OUT	SPARE
Z	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND	GND

* NOTE
ALL CONNECTORS ARE 22 PIN METHODE PLUGS (MALE)
AND ALL MATING PLUGS SHOULD BE SAME.

External Device Connections
ID-D-136-0-EDC

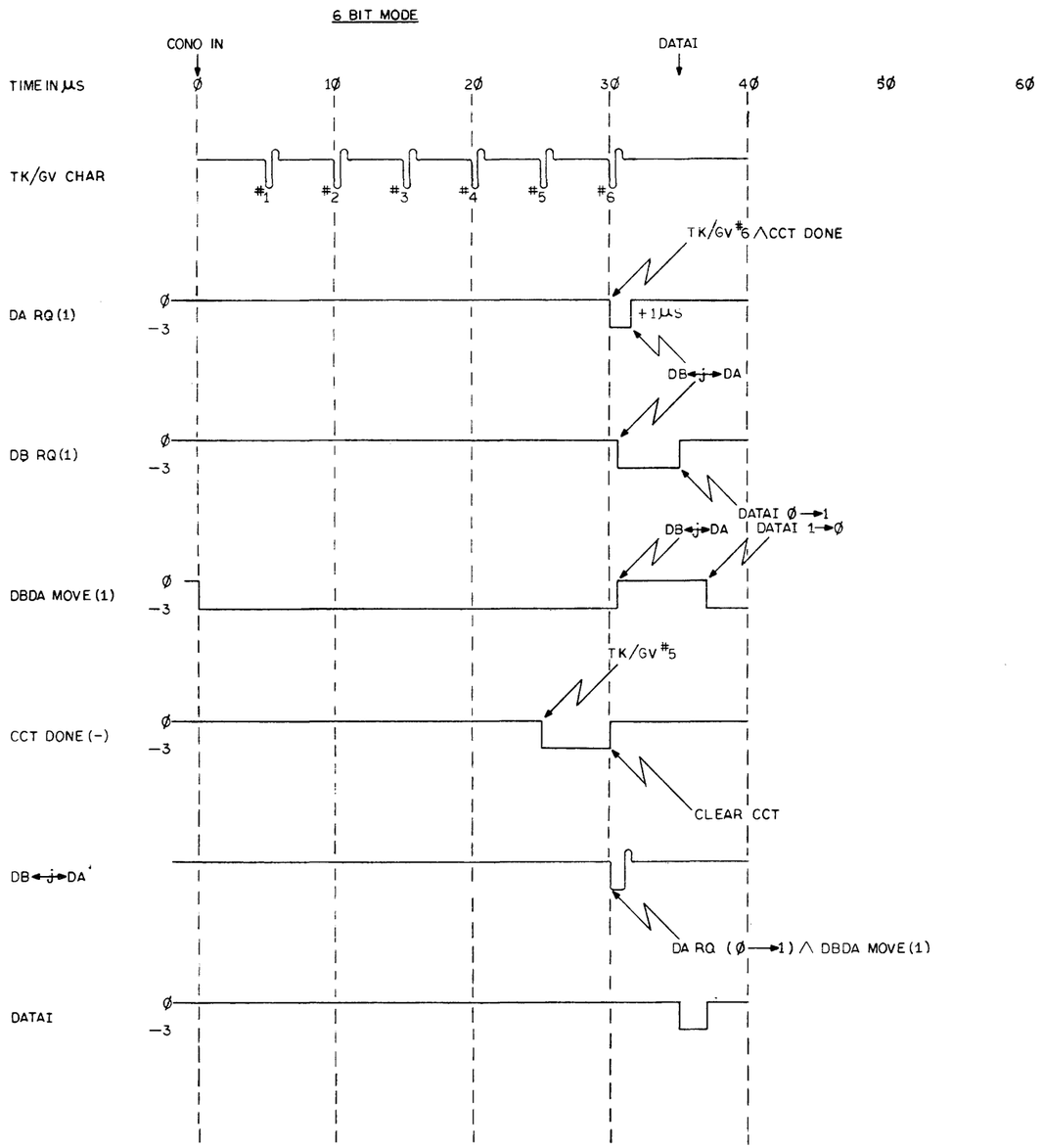
Data Control System Diagram
SD-D-136-0-DCSD



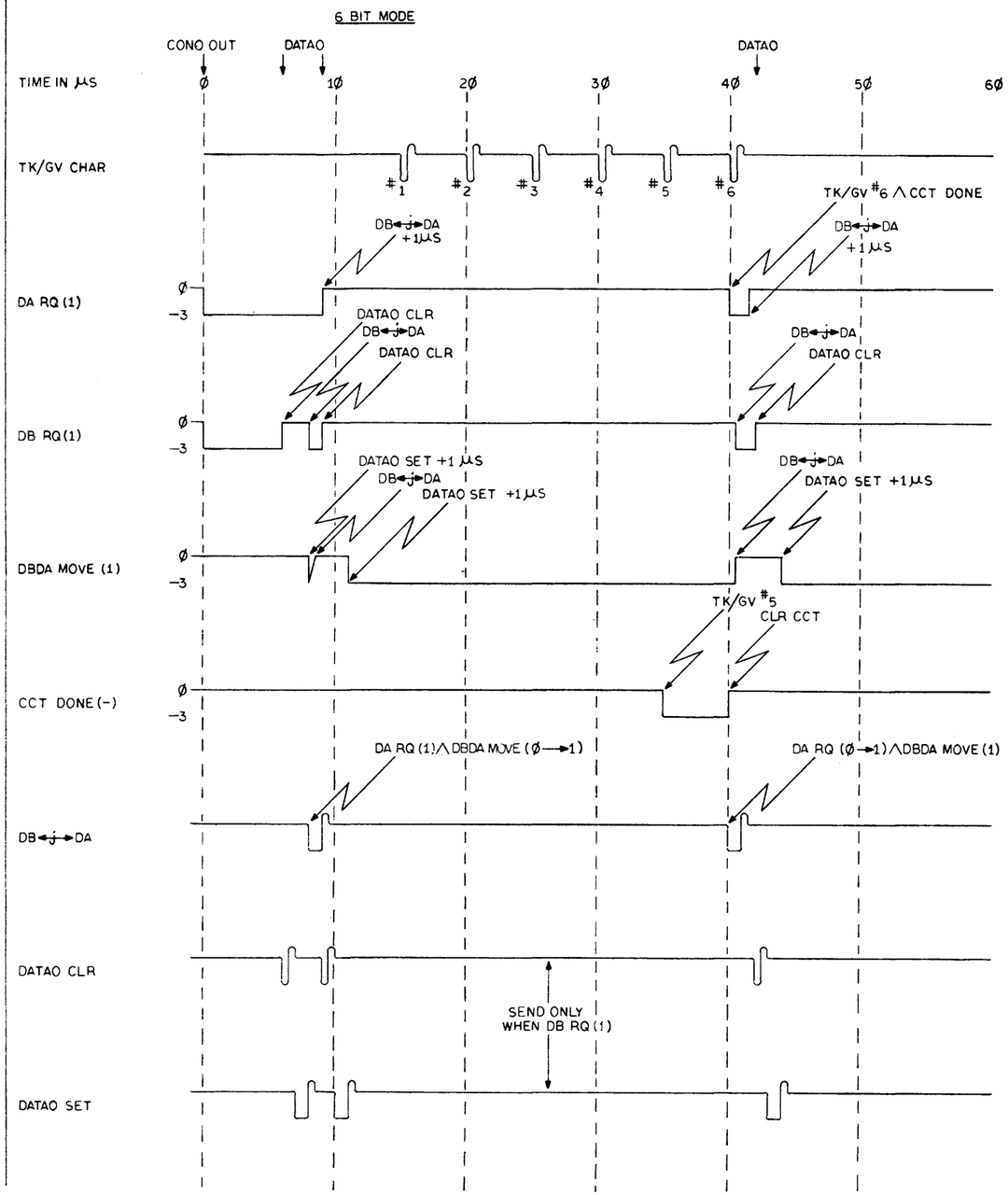
Data Control System Diagram
SD-D-136-0-DCSD

Timing Diagram TD-D-136-0-TD1

INPUT

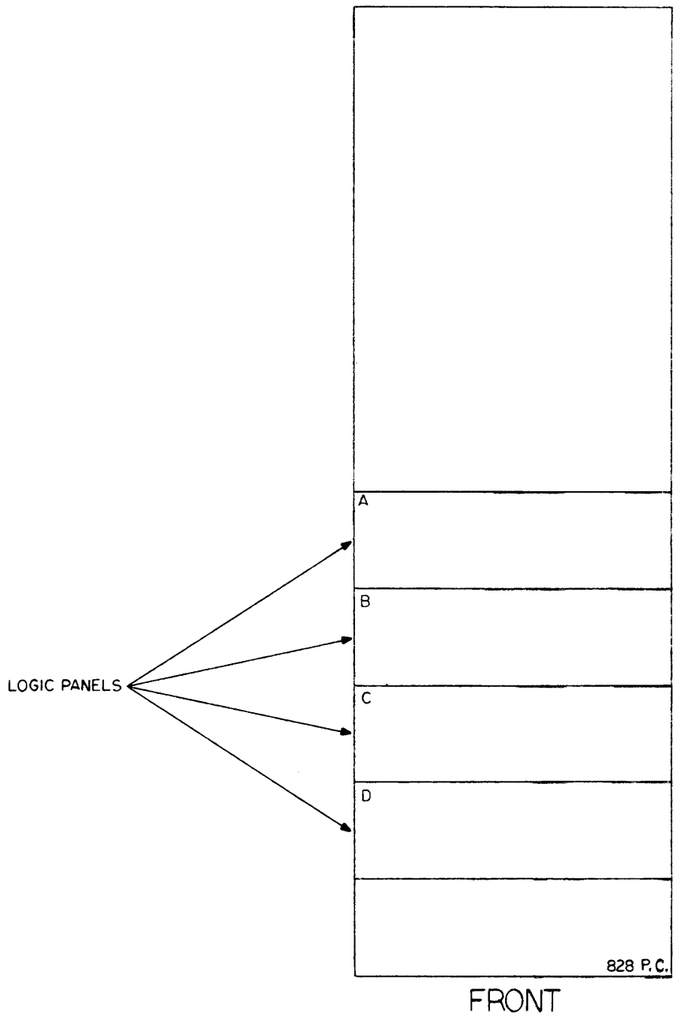


OUTPUT

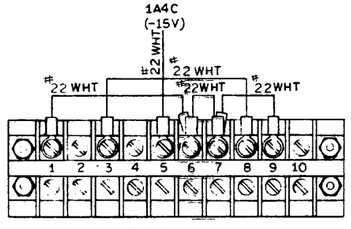


Timing Diagram TD-D-136-0-TD1

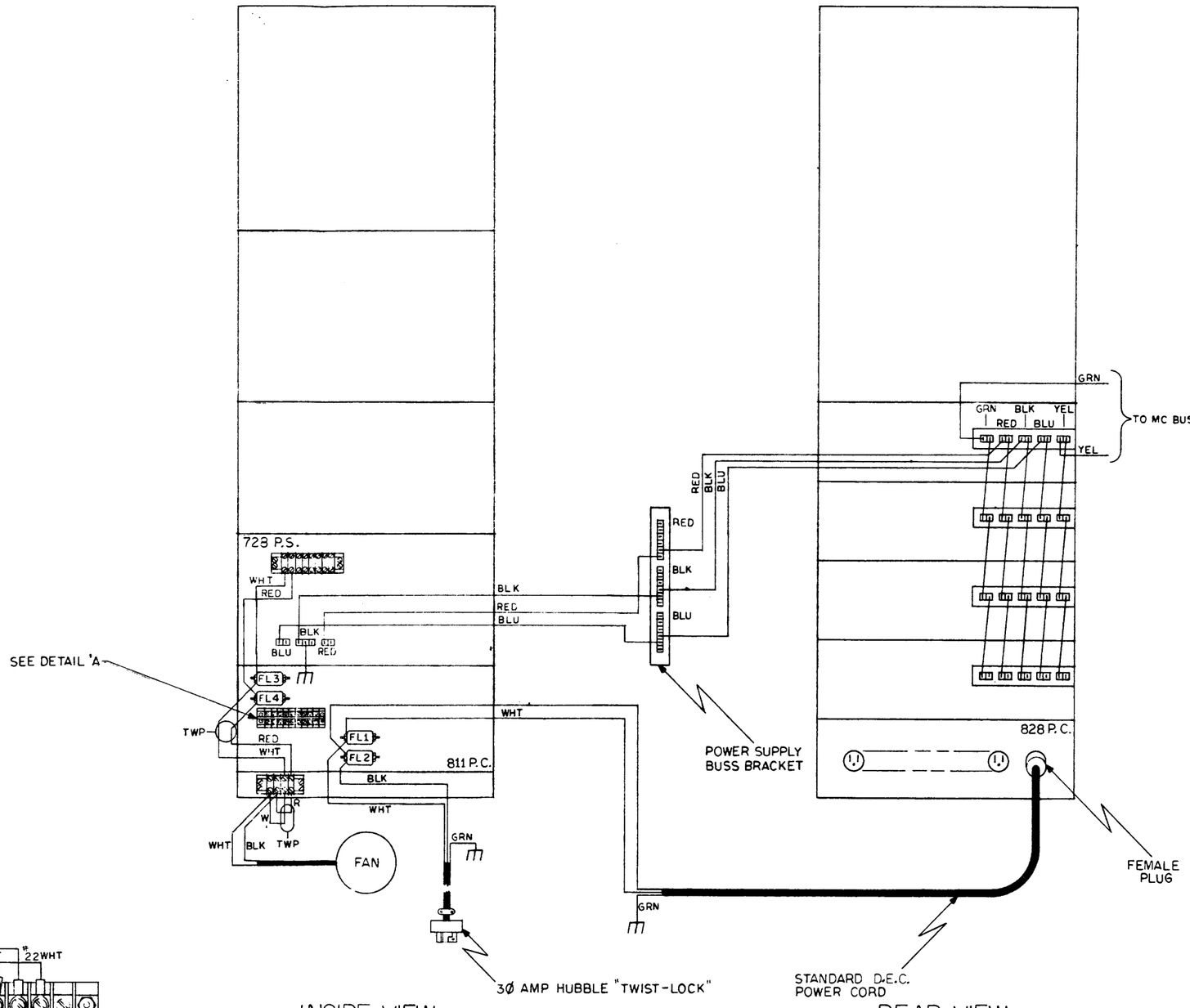
Data Control Power Wiring
PW-D-136-0-DCPW



FRONT



DETAIL A



INSIDE VIEW
OF PLENUM DOOR

REAR VIEW
OF LOGIC PANELS

Data Control Power Wiring
PW-D-136-0-DCPW

Utilization Module List (sheet 1)
UML-D-136-0-DCML

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
	1Ø32	1Ø32	1Ø32	1Ø32		4118-R	46Ø4	4606	46Ø4	46Ø6	4127-75	4215	6115-02	6117-R	6117	4215-R	4606	4666	4129-R	4129-R	46Ø4	1310			
1A		METHODE		PLUGS		DC SELECT	DC CLR SCT DC CLR CCT	DC IC CLR DC IC SET	DC IC CLR DC DB←IOB DC DB RQ←1	DC CLR DB DC DB→J→DA DC DA→J→DB	DC SCT 1 DC CCT 2 DC DB←IOB DC CLR CCT DC CLR SCT	DC SCT Ø DC SCT 1	DC SCT DONE	DC CCT DONE	DC CCT DONE	DC CCT Ø DC CCT 1 DC CCT 2	DASH STRB PT DEV 5 DASH STRB LT DEV 5 DASH STRB RT DEV 6	DASH LT DASH CP Ø DASH STRB LT DEV 6	DASH LT R DASH CP Ø	DASH RT R DC CLR CCT	DASH CP 1 DASH RT	DASH CP 1			
	1Ø32	1Ø32	1Ø32	1Ø32		4112-76	4113-R	4151	4102-R	4127	4217	4657	4217	4657	4217	4657	4217	4150	6102-15Ø	6102-R	6684	1669			
						R DC CCT CONT R DC SCT CONT DC DB RQ	R DC STATUS R DC DB RQ←1 R DC DB→IOB	DC PI DECODE	DC DB IOB DC STATUS DC SCT DONE DC CCT DONE DC DA RQ DC DB DA MOVE	DC DATA CLBD DC DBDA MOVE DC DA RQ	DC DATA CLBD	IOB 2Ø IOB 21 IOB 22 DC DATA CLBD	DC DB DA MOVE DC DA RQ DC DB RQ DC IN/OUT	DC DB DA MOVE DC DA RQ DC DB RQ DC IN/OUT DC CH MODE Ø DC CH MODE 1	DC CH MODE Ø DC CH MODE 1 DC DEV Ø DC DEV 1	DC DEVICE Ø DC DEVICE 1 DC DEVICE 2 DC PIA Ø DC PIA 1 DC PIA 2	DC DEV 2 DC PIA Ø DC PIA 1 DC PIA 2	DC DEVICE DECODE	DC DEVICE SELECT (B) R DC PI DECODE R DC DB→IOB R DC DB DA MOVE DC DBDA MOVE DC DB RQ	DC DEVICE SELECT (b) DASH CP 1	DC STATUS DC DA RQ B (1)	DC DARQ(1) DC DBRQ(1) DC DBDA(1) DC IN/OUT(1) DC CH MODE Ø(1) DC CH MODE 1(1) DC CCT Ø(1) DC CCT 1(1) DC CCT 2(1)			
1B		METHODE		PLUGS																					

Utilization Module List (sheet 1)
UML-D-136-0-DCML

Utilization Module List (sheet 2)
UML-D-136-0-DCML

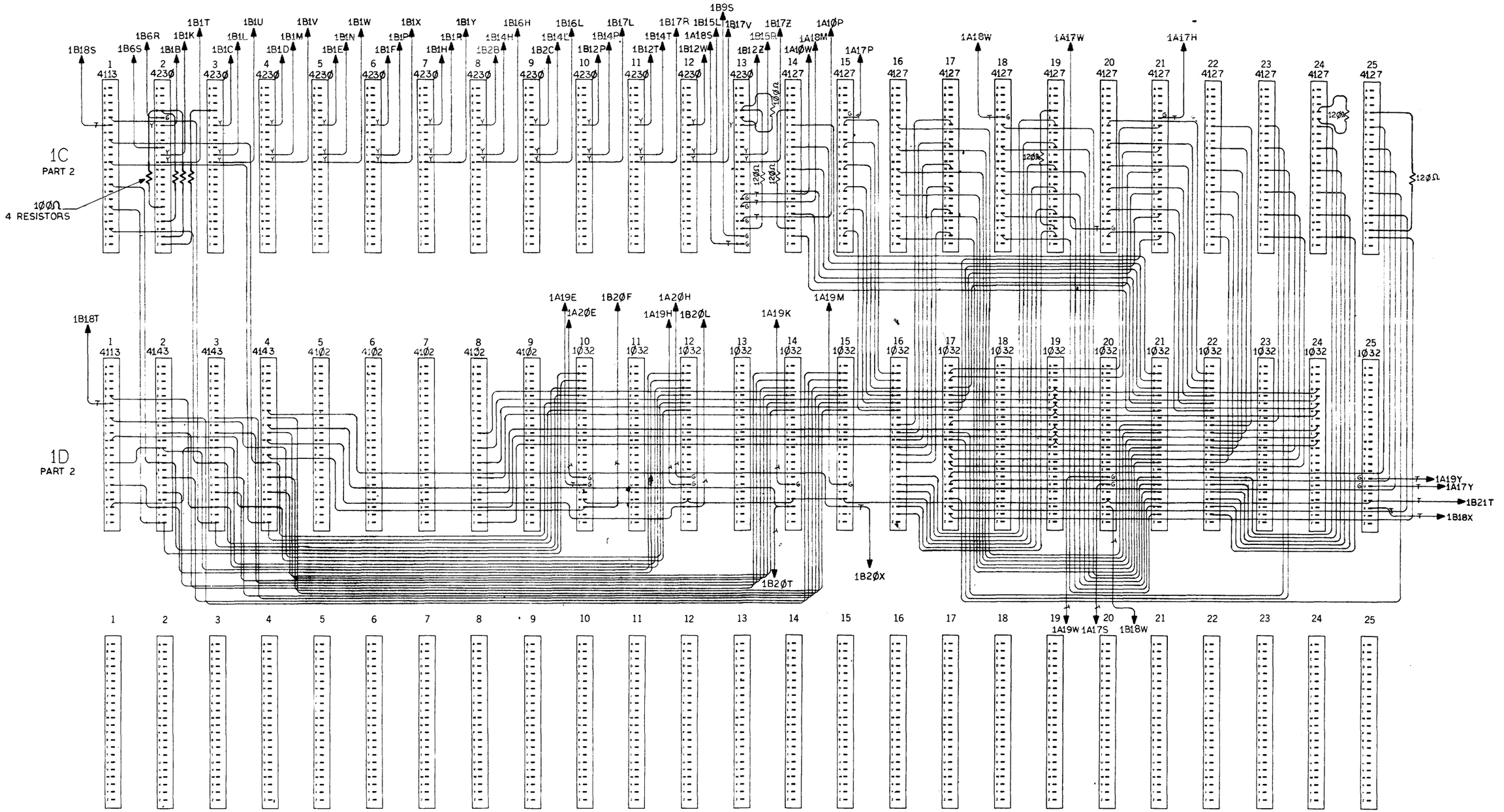
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
.4113-R	4230	4230	4230	4230	4230	4230	4230	4230	4230	4230	4230	4230	4127	4127	4127	4127	4127	4127	4127	4127	4127	4127	4127	4127
DASH DATA IN Ø LT																								
DASH DATA IN 1 LT													DAØ(1)	DAØ(1)	DA6(1)	DA6(1)	DA12(1)	DA12(1)	DA18(1)	DA18(1)	DA24(1)	DA24(1)	DA3Ø(1)	DA3Ø(1)
DASH DATA IN 2 LT	DBØ,6,12 & DAØ,6,12	DB1,7,13 & DA1,7,13	DB2,8,14 & DA2,8,14	DB3,9,15 & DA3,9,15	DB4,1Ø,16 & DA4,1Ø,16	DB5,11,17 & DA5,11,17	DB18,24,3Ø & DA18,24,3Ø	DB19,25,31 & DA19,25,31	DB2Ø,26,32 & DA2Ø,26,32	DB21,27,33 & DA21,27,33	DB22,28,34 & DA22,28,34	DB23,29,35 & DA23,29,35	THRU DA5(1)	THRU DA5(1)	THRU DA11(1)	THRU DA11(1)	THRU DA17(1)	THRU DA17(1)	THRU DA23(1)	THRU DA23(1)	THRU DA29(1)	THRU DA29(1)	THRU DA35(1)	THRU DA35(1)
DASH DATA IN 3 LT																								
DASH DATA IN 4 LT																								
DASH DATA IN 4 LT																								
4113	4143-R	4143-R	4143-R	41Ø2-R	41Ø2-R	41Ø2-R	41Ø2-R	41Ø2-R	1Ø32	1Ø32	1Ø32	1Ø32	1Ø32	1Ø32	1Ø32	1Ø32	1Ø32	1Ø32	1Ø32	1Ø32	1Ø32	1Ø32	1Ø32	1Ø32
DASH DATA IN Ø LT																								
DASH DATA IN 1 LT	DASH DATA IN 3Ø RT	DASH DATA IN 32 RT	DASH DATA IN 34 RT	DA Ø B(1) THRU DA 8 B(1)	DA Ø B(1) THRU DA 5 B(1)	DA 9 B(1) THRU DA 17 B(1)	DA 18 B(1) THRU DA 26 B(1)	DA 27 B(1) THRU DA 35 B(1)																
DASH DATA IN 2 LT																								
DASH DATA IN 3 LT	DASH DATA IN 31 RT	DASH DATA IN 33 RT	DASH DATA IN 35 RT																					
DASH DATA IN 4 LT																								
DASH DATA IN 5 LT																								
										M	E	T	H	O	D	E			P	L	U	G	S	

1C

1D

Utilization Module List (sheet 2)
UML-D-136-0-DCML

Data Buffer and Data Accumulator Racks C and D
WD-D-136-1-DAB1



Data Buffer and Data Accumulator Racks C and D
WD-D-136-1-DAB1

H-270

DISCFILE CONTROL

TYPE 270

INSTRUCTION MANUAL

DIGITAL EQUIPMENT CORPORATION · MAYNARD, MASSACHUSETTS

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

CONTENTS (continued)

ILLUSTRATIONS

ENGINEERING DRAWINGS

ENGINEERING DRAWINGS (continued)

TABLES

CHANGES TO DISCFILE CONTROL 270 INSTRUCTION MANUAL

(Notation: "Line 5-B" means 5th line from bottom of page.)

No.	Page	Line	Correction
1)	viii	5-B	change page 6-9 to page 6-4
2)	3-4	3-B	~SEL 5 (-SC:B7)
3)	3-5	4	~SEL 5 level
4)	3-5	---	(The following entry should be inserted in Table 3-2 between the SELECT entry and the ALERT entry:) ADDRESS TERMINATION (-TR:A1) — Notifies the discfile that the address transmission has been completed.
5)	3-11	10	The ATP pulse steps the octoflop to its third state ADT (Address Terminated) and sends an ADDRESS TERMINATION pulse to the discfile. During the address ... etc.
			(Items Below All in Chapter 8 Signal Glossary)
6)	1d		Insert between 3) and 4) the following: 3A) ADDRESS TERMINATION
	1d		Change 96 to read as follows: 96) WCT0-7
7)	5		Renumber page 5, "5a" and insert page 5b (attached) after page 5a.
8)	12	7-B	the discfile. It is amplified, redesignated ADDRESS TERMINATION, and sent to the discfile approximately 15 μsec after the transmission of the parity bit, the final bit of the 21-bit address.

No.	Page	Line	Correction
9)	34	1-B	ALS state to CMS state, and triggers a 200-μsec delay that ends with the generation of the NDP-1 pulse.
10)	90	1-B	The discfile control sends a TAKE A CHARACTER pulse to the Data Control 136.
11)			In general wherever "Take a Character" or "Give a Character" is encountered change to "TAKE A CHARACTER" or "GIVE A CHARACTER", respectively. (The reason for this is that all output pulses from the Discfile Control are for consistency capitalized throughout the manual.)
12)	106	1	WCT0-7 - Word Counter (-WC:A3-7)

(3A) ADDRESS TERMINATION (-TR:A1)

The ADDRESS TERMINATION pulse is sent to the discfile through the odd data 1's twisted-pair transmission line. It notifies the discfile that the address transmission has been completed.

Generated by-

ATP

The discfile control produces an ATP pulse to advance the file-control octoflop from SNA state to ADT state (see ATP below.) After being amplified by pulse amplifier D30, the ATP pulse is redesignated ADDRESS TERMINATION and sent to the discfile.

CHAPTER 1

INTRODUCTION

1.1 PURPOSE AND SCOPE

The purpose of this instruction manual is to aid personnel in the installation, operation, and maintenance of the DEC Discfile Control 270 (Figure 1-1).

1.2 MANUAL ORGANIZATION

Chapter 2, General Description, comprises a broad description of system application and specifications. Chapter 3 contains the theory of operation of the discfile control logic; this chapter is organized horizontally by areas of the system logic and presents a brief overall description of all major signals and circuits used in the discfile control. Chapter 3 is fully cross-referenced to the Chapter 8 signal glossary. Chapter 4, System Operating Sequences, adds a vertical time dimension to the theory of operation by presenting a step-by-step analysis of system function in the form of highly detailed micro-flow-charts that illustrate each of the four major operating sequences executed by the discfile control. Chapter 5, Maintenance, includes information useful for inspection, troubleshooting, and repair. Chapter 6, Installation, explains the procedures to be followed when first unpacking and installing the discfile control. Chapter 7, Programming, tabulates the coding of the various I/O instructions used with the discfile control, and gives a listing and brief explanation of several typical subroutines. Chapter 8 presents a complete signal

glossary for the discfile control, its inputs, and its outputs. Chapter 9 contains introductory information on DEC drawing conventions and all pertinent engineering drawings for this manual.

a Figures

This manual includes four general classes of figures: engineering logic diagrams; flow diagrams; circuit schematics; and miscellaneous figures such as photographs, block diagrams, and module location diagrams. The complete system logic is shown in the seven engineering logic diagrams listed below. These seven drawings are included in Chapter 9.

ENGINEERING LOGIC DIAGRAMS

Drawing Number	Title
BS-D-270-0-AC	Accumulator, Shift Counter, and Address Clock Logic
BS-D-270-0-AD	Accumulator and Bit-Control Logic
BS-D-270-0-CM	Command Logic
BS-D-270-0-FC	File-Control Logic
BS-D-270-0-SC	Status Logic
BS-D-270-0-TR	Transmitters, Receivers, and Data Clock Logic
BS-D-270-0-WC	Word Counter Logic

All system operations are shown in the four flow diagrams listed below; these diagrams are located in Chapter 9.

FLOW DIAGRAMS

Drawing Number	Title
FD-D-270-0-8	File-Control State Diagram
FD-D-270-0-FD (Sheet 1)	Address Transmission and Data-Output Flow Diagram
FD-D-270-0-FD (Sheet 2)	Data Clock Flow Diagram
FD-D-270-0-FD (Sheet 3)	Processor Instructions, Discfile Command Signals, and Error Indications Flow Diagram

Module location diagrams and circuit schematics for the 27 types of FLIP CHIP Modules that are used in the discfile control are also in Chapter 9.

At the date of issue of this manual, up-to-date engineering drawings are included in Chapter 9. If significant engineering changes occur, DEC customarily issues revised drawing sets to reflect the changes. If any discrepancy exists between the engineering drawings of this manual and a current set of engineering drawings, the current engineering drawing set is correct.

For quick reference, all engineering drawings within a given category (Logic Diagrams, Flow Diagrams, etc.) are arranged numerically. Furthermore, all circuit schematics of Chapter 9 are arranged in order of circuit type designation (e. g. Dual Flip-flop R202 precedes Triple Flip-flop R203).

1.3 USE OF MANUAL

This manual is intended to serve two purposes: instruction and reference. Chapters 3 and 4 on theory of operation are intended primarily for instruction. These chapters cover the major logical networks and operating sequences of the system in a systematic and detailed manner, but they are not organized for rapid reference. After being read, they rarely need be referred to again.

To avoid burdening the reader with unnecessary detail, the theory of operation chapters usually concentrate on the single circuit or operation being described. Peripheral description of detailed signal generation conditions and additional uses of the signal mentioned are not included in the instruction text. However, all such detailed reference data is readily available for quick look-up in Chapter 8.

The Chapter 8 signal glossary includes alphabetically-ordered entries for all input and output signals of the discfile control, for all logic levels and pulses used within the system, and for all control and status flip-flops. Each glossary entry comprises a detailed description of both source and application of the signal. The signal glossary as a whole thus provides a random-access description of the entire system logic. Because the glossary is completely cross-referenced, the user can extract the specific information he requires with minimum time and effort.

Once they have been thoroughly understood, the engineering logic drawings and the flow diagrams provide the fastest and most extensive source of reference data. After a reasonable learning period, a competent

maintenance engineer can orient logic function with particular drawings and even with the approximate area of the drawing which shows given sections of the logic. For reference purposes, the Chapter 8 signal glossary can be used as a highly detailed index to the engineering drawings. The glossary includes the drawing coordinates of each signal as well as a description of its source and use.

1.4 REFERENCE CONVENTIONS

The Digital Equipment Corporation engineering drawing conventions and instruction manual referencing should be understood at this point. A study of the material contained in Chapter 9 and the following paragraphs before proceeding with detailed descriptions will save considerable reference time and preserve continuity of thought when reading the text that follows.

Any reference to an illustration by a chapter-oriented figure number indicates that the figure is to be found in the text following the reference.

Example: Figure 2-1 (The first in-text figure of Chapter 2.)

Any reference to an engineering drawing number indicates that the drawing is to be found in a special drawing section or chapter. All engineering drawings are referenced first by the full drawing number.

Example: BS-D-270-516-0-CM

To locate a specific signal or function on a drawing, a system of coordinates is used. As shown on the drawings of Chapter 9, coordinates are designated by a letter and number. Thus, in any drawing reference,

coordinate location appears immediately after the number separated by a colon.

Example: BS-D-270-516-0-CM:A3

One last text reference convention should be noted. Occasionally it is desirable to indicate the state of a circuit within a logic description. This is done by following the mnemonic designation of the circuit with the circuit state in parentheses. For example, the parity error flip-flop is designated PER on the Status Logic drawing. If the flip-flop is in the 1 state, assertion of either of its output levels is designated PER(1), but if the flip-flop contains 0, then the asserted output levels are designated PER(0).

Its output levels are designated as follows:

When PER contains 1: PER(1) is asserted

When PER contains 0: PER(0) is asserted

Note that the assertion level can be negative or ground for either state of the flip-flop.

The signal glossary of Chapter 8 is also an important adjunct to both the text and the drawings. This glossary can be used in a cross-indexing manner so that any signal and the conditions that generate it can be easily and completely referenced. By using a chain look-up procedure, the user can proceed from any input to its final consequences, or from any output to its original sources.

1.5 REFERENCE DOCUMENTS

By Digital Equipment Corporation:

1) FLIP CHIP Modules Catalog, C-105

2) MAINDEC 670 - Magnetic Disc Test Maintenance Program

By Data Products Corp. DISCfILE* Division:

1) General Information Manual dp/f-5022, DISCfILE Storage System (Oct. 1964)

2) DISCfILE Instruction Manual Model dp/f-5022 with Addendum 0012 for Series 1-17 Systems (March 15, 1965)

3) Specification for Model 5022 DISCfILE Storage System No. 102253 with Addendum A (Dec. 23, 1964)

4) Pre-Delivery Acceptance Test for Model 5022 DISCfILE Storage System No. 102254 (Dec. 4, 1964)

5) Customer Information Bulletins for Model 5022 DISCfILE Storage Systems

* Trademark of Data Products Corp. DISCfILE Division, P.O. Box 751, Gateway Station, Culver City, California

CHAPTER 2

GENERAL DESCRIPTION

2.1 SYSTEM APPLICATION

The Discfile Control 270 and Discfile 5022 constitute a fast, random-access mass storage system capable of rapidly transferring large blocks of data to or from PDP-6 core memory. Each discfile unit stores 5.76 million 37-bit data words, slightly over twice the amount of data that can be stored on a 2400-foot magnetic tape at 556 frames per inch density. The rate of data transfer to or from the discfile system is about ten times that of magnetic tape (assuming that the tape is operated at 556 density and 112.5 inches per second tape speed). For inner-zone operation, the discfile transfers data at about 400 kc (2.4 μsec per bit); the outer-zone data transfer rate is about 700 kc (1.4 μsec per bit). Over 63% of the data is stored in outer zone sectors, so the mean data-transfer rate is about 590 kc. Random access is available to any given sector of data but not to individual data words within the sectors; each sector contains 128 data words. Following a single access, up to 44 sequentially-addressed sectors can be processed without further addressing.

Figure 2-1 is a block diagram showing a typical system configuration. Every discfile system contains a Discfile Control 270 and a Discfile 5022. The Discfile 5022 consists of a logic unit and one to four discfile units. During both output and input operations, 19-bit addresses are sent directly from the processor to the discfile control. The actual data sent to or

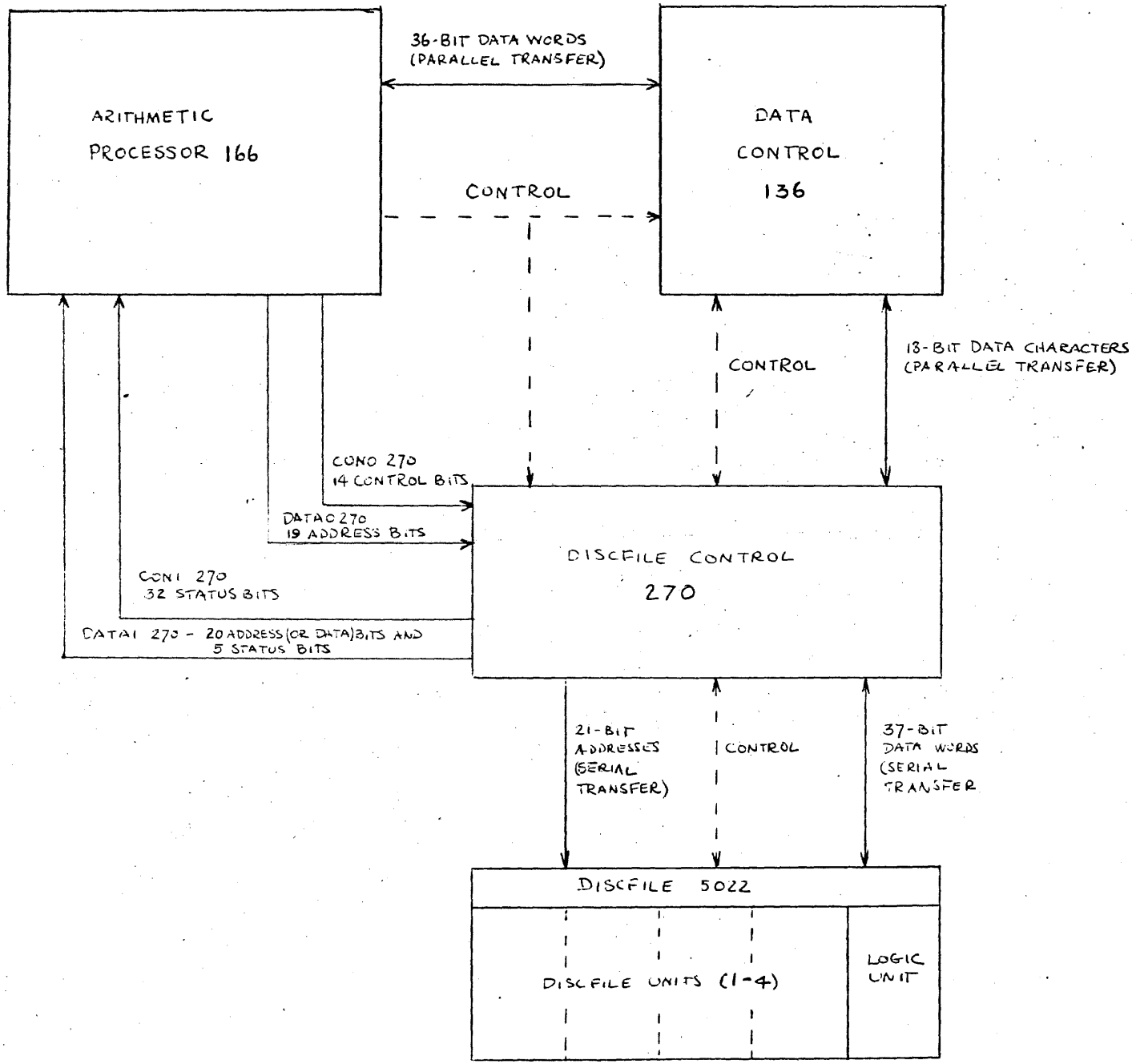


FIGURE 2-1 DISCFILE SYSTEM CONFIGURATION

received from the addressed locations follows a different route between the processor and the discfile control. Data flow from the processor to the discfile control and from the discfile control to the processor is always through the Data Control 136.

Output data from the processor is transferred out to the Data Control 136 in 36-bit data words. After receiving each 36-bit data word from the processor, the Data Control 136 divides it into two 18-bit characters and transfers these characters to the Discfile Control 270 one at a time. If the data is to be written at the discfile (rather than to be used within the discfile control for read comparison), the discfile control transmits the data to the Discfile 5022 serially one bit at a time. After transmitting each pair of characters to the discfile, the discfile control generates a parity bit and transmits it also to the discfile. At the discfile, all 37 bits (36 data bits and the parity bit) are written onto the addressed sector.

For input operations, the process described above is essentially reversed. At the Discfile 5022, data is read from the addressed sector. This data is then serially transferred into the Discfile Control 270, every 36 data bits being followed by a parity bit. At the discfile control, the parity of the incoming data is checked. If the data is to be read into the processor (rather than to be used within the discfile control for read comparison) the discfile control assembles 18-bit data characters for transfer into the Data Control 136. The data control in turn assembles pairs of 18-bit characters into 36-bit data words, and transfers these data words into the processor.

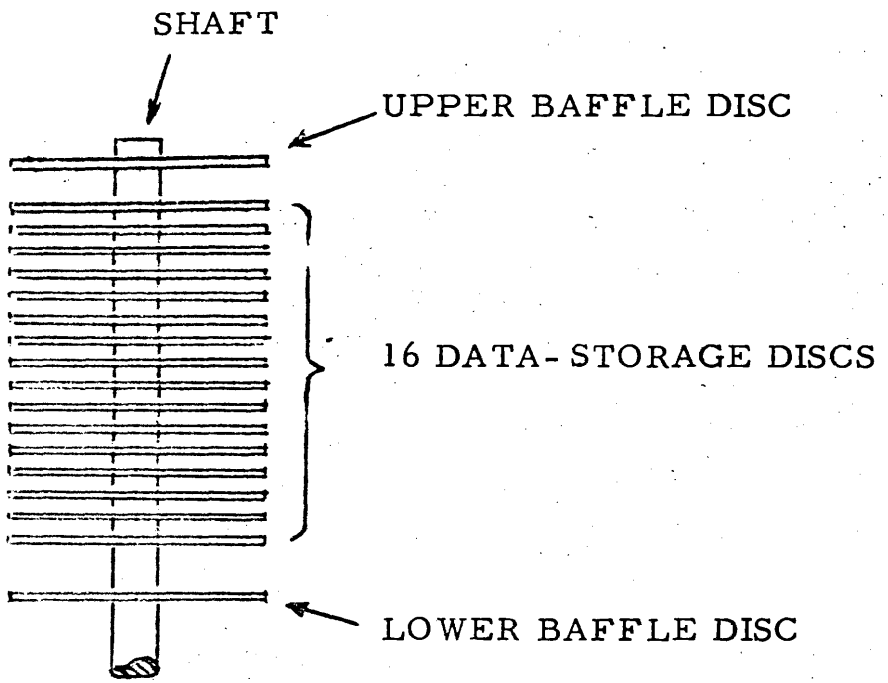
2.2 DESCRIPTION OF DISCFILE 5022

A basic understanding of the structure and operation of the Discfile 5022 is of considerable help in learning the function of the Discfile Control 270. The present paragraph is included only as a brief summary description of the discfile; for more detailed information refer to the reference documents listed at the end of Chapter 1.

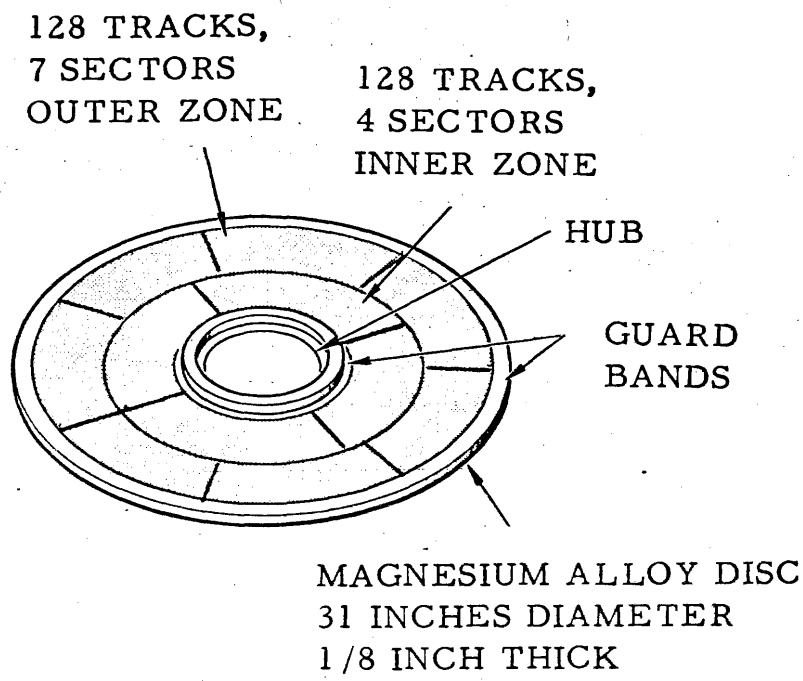
a Discfile Data-Storage

A Discfile 5022 contains a logic unit and one to four discfile units. Each discfile unit stores data on 16 ferrous-oxide coated magnesium alloy discs 31 inches in diameter, all mounted on a common axial shaft which rotates at about 1200 rpm. Top and bottom baffle discs are mounted on the same shaft above and below the stack of 16 data-storage discs (Figure 2-2). The bottom surface of the upper baffle disc contains the control tracks that provide inner and outer zone write clock pulses, sector characters and index characters. The lower baffle disc contains control tracks or data.

A typical data-storage disc is shown at the right of Figure 2-2. The data is stored on both the top and bottom surfaces of the disc. Each surface of the disc is divided into two zones, each containing 128 concentric data tracks. The tracks are spaced 0.037 inches apart center to center. The inner zone is divided into four sectors; the outer zone is divided into seven sectors. On a single sector of a single track the discfile can store 128 37-bit data words. Because seven outer-zone sectors may be written or read in the same time that is required to write or read four inner-zone



DISC STACK



SINGLE DATA-STORAGE DISC

FIGURE 2-2 THE DATA-STORAGE ASSEMBLY OF A DISCFILE UNIT

sectors, the outer zone uses a faster clock rate than the inner zone (1.4 μsec per bit, outer zone; 2.4 μsec per bit, inner zone).

Associated with each disc is a forked access arm and a digital positioner. The digital positioner is a linear dc motor which includes a permanent magnet armature that is attached to one end of the access arm. The digital positioner can move the access arm rapidly and precisely to any of 64 evenly spaced radial positions. Each of these 64 positions is permanently fixed by a magnetic "notch" in the stator of the digital positioner.

One branch of each forked access arm passes over the associated disc, the other branch passes under the disc (see Figure 2-3). Each of the two branches of the access arm supports an inner-zone flying-head pad and an outer-zone flying-head pad.

TABLE 2-1 FLYING-HEAD PAD ASSIGNMENT

Branch of Access Arm	Flying-head Pad	Provides Access to the 128 Data Tracks Located In:
Upper	Inner-zone	Inner zone of disc's top surface
Upper	Outer-zone	Outer zone of disc's top surface
Lower	Inner-zone	Inner zone of disc's bottom surface
Lower	Outer-zone	Outer zone of disc's bottom surface

Each of the four flying-head pads that are mounted on a given access arm contain two separate read-write heads accurately mounted at a fixed radial separation equal to an odd number of track widths. (Each of these

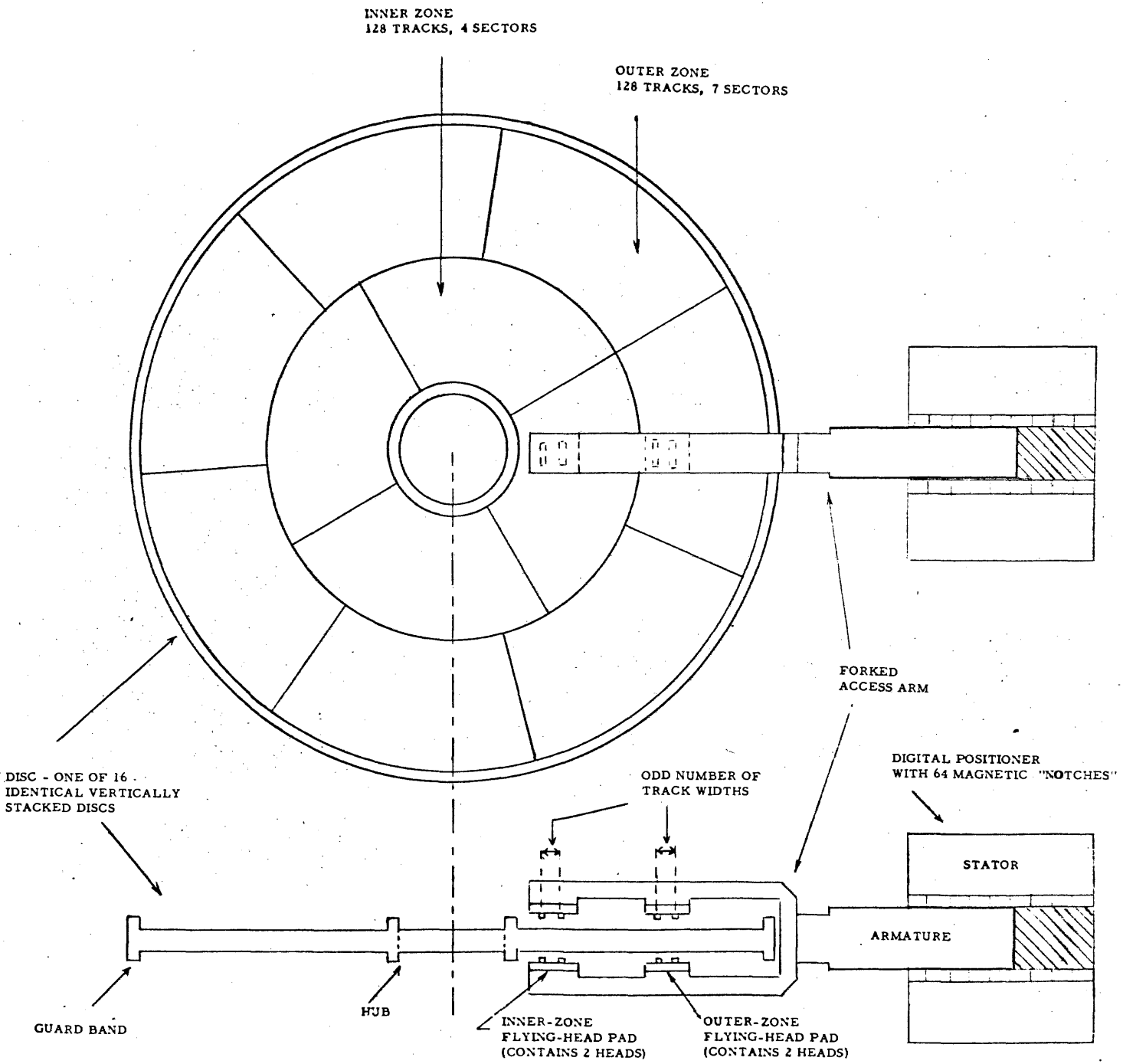


FIGURE 2-3 SIMPLIFIED SCHEMATIC DIAGRAM
OF SINGLE DISC AND ACCESS ARM

read-write heads contains a read-write gap and an erase gap.) As a result of this arrangement, eight separate tracks (one track corresponding to each of the eight read-write heads) are available for reading or writing at any given position of the access arm.

If the digital positioner were to move the access arm to each of the 64 discrete radial positions (one after another), the eight read-write heads would index to all 512 data tracks on the disc (64 x 8 = 512). These 512 data tracks would include every one of the 256 outer-zone tracks (128 outer-zone tracks on the top surface of the disc and 128 outer-zone tracks on the bottom surface of the disc) and every one of the 256 inner-zone tracks. When the access arm is at rest in any one of the 64 positions to which the digital positioner may be indexed, 44 sectors of data are available to the read-write heads without further mechanical motion (28 outer-zone sectors, and 16 inner-zone sectors).

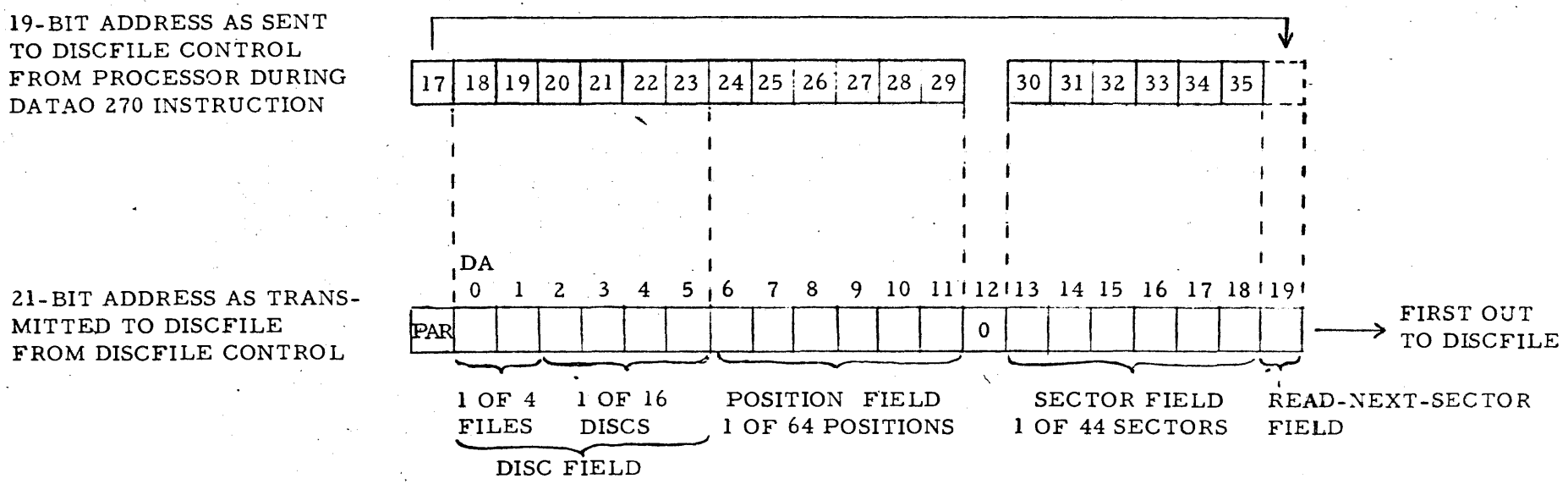
The elements that make up a Discfile are summarized in Table 2-2 below.

TABLE 2-2 ELEMENTS OF DISCFILE 5022

Elements	Number of Elements in Each: Position	Disc	Discfile Unit	Discfile 5022
Discfile Units	-	-	-	1-4
Discs, Access Arms, Positioners	-	-	16	16-64
Flying-head pads	-	4	64	64-256
Read-write Heads	-	8	128	128-512
Positions	-	64	1024	1024-4096
Tracks	8	512	8192	8192-32,768
Sectors	44	2816	45,056	45,056-180,224

b Address Structure

Data is written into or read from the Discfile 5022 in sectors of 128 data words each (sometimes referred to as records). Any given sector is uniquely addressed by a 21-bit address which specifies its physical location within the discfile. The structure of this address is shown in the lower portion of Figure 2-4. The upper portion of the figure shows the source of the address fields in the DATAO 270 instruction coding. Table 2-3 tabulates the read-write heads addressed by the 44 presently used sector-field configurations.



DISCFILE ADDRESS FIELDS

FIGURE 2-4 ADDRESS STRUCTURE

TABLE 2-3 HEAD ADDRESS TABULATION

Decimal Address	Record Field Binary Equivalent							Addressed Head	Zone
	64	32	16	8	4	2	1		
0	0	0	0	0	0	0	0		
through				through				0	IZ
3	0	0	0	0	0	1	1		
4	0	0	0	0	1	0	0		
through				through				1	IZ
7	0	0	0	0	1	1	1		
8	0	0	0	1	0	0	0		
through				through				2	IZ
11	0	0	0	1	0	1	1		
12	0	0	0	1	1	0	0		
through				through				3	IZ
15	0	0	0	1	1	1	1		
16	0	0	1	0	0	0	0		
20	0	0	1	0	1	0	0		
24	0	0	1	1	0	0	0		
28	0	0	1	1	1	0	0	4	OZ
32	0	1	0	0	0	0	0		
36	0	1	0	0	1	0	0		
40	0	1	0	1	0	0	0		
17	0	0	1	0	0	0	1		
21	0	0	1	0	1	0	1		
25	0	0	1	1	0	0	1		
29	0	0	1	1	1	0	1	5	OZ
33	0	1	0	0	0	0	1		
37	0	1	0	0	1	0	1		
41	0	1	0	1	0	0	1		
18	0	0	1	0	0	1	0		
22	0	0	1	0	1	1	0		
26	0	0	1	1	0	1	0		
30	0	0	1	1	1	1	0	6	OZ
34	0	1	0	0	0	1	0		
38	0	1	0	0	1	1	0		
42	0	1	0	1	0	1	0		

TABLE 2-3 HEAD ADDRESS TABULATION (continued)

Decimal Address	Record Field Binary Equivalent							Addressed Head	Zone
	64	32	16	8	4	2	1		
19	0	0	1	0	0	1	1		
23	0	0	1	0	1	1	1		
27	0	0	1	1	0	1	1		
31	0	0	1	1	1	1	1	7	OZ
35	0	1	0	0	0	1	1		
39	0	1	0	0	1	1	1		
43	0	1	0	1	0	1	1		

The first address bit to be transmitted from the discfile control to the discfile is the read-next-sector bit. When this bit is coded 1 (as it normally is), it causes the discfile to process the single sector of data that is uniquely specified by the next 19 bits of the address. If, however, the read-next-sector bit is coded 0, the discfile ignores the first three bits of the sector field (the three bits that select a single sector to be processed from among the four or seven sectors constituting the addressed track). The discfile then processes whichever sector of the addressed track first passes the heads. (Ordinarily, this mode of operation is useful only for maintenance procedures; it is rarely employed during normal operations.)

There are several differences worth noting between the 19-bit address sent from the processor to the Discfile Control 270, and the 21-bit address transmitted to the Discfile 5022 from the discfile control. A 0 bit, generated within the discfile control, is always transmitted to the

discfile between the sector field and the position field of the address. At the discfile this bit is presently unused, but if the system were to be modified to include smaller but more numerous sectors, the 0 bit would be available for use as a seventh bit of the sector-field. After transmitting 20 address bits to the discfile, the discfile control generates and transmits an address parity bit. This 21st and final bit of the address enables the discfile to determine whether any errors have occurred during the address transmission. The format of the discfile logic requires that the read-next-sector bit be located at the sector-field end of the address (rather than at the disc-field end of the address as it is coded in the DATAO 270 instruction. The requisite transposition is accomplished at the input gating of the discfile control data accumulator.

c Minimum-Access Programming

When a new sector is selected for processing, the required access time depends on the amount of switching time and physical positioning time that is needed to index from the last sector processed to the new sector address. Table 2-4, System Operating Specifications, illustrates the method of computing the access time that is expended for any given change of address. For most applications, a typical file of data occupies only a fraction of the total storage capacity of the discfile system. By choosing the shape of the file in an appropriate manner, the average access time can be substantially reduced, and the operating rate of the system can be increased proportionately.

2.3 PROGRAMMED OPERATIONS

The DATAI 270 and CONI 270 instructions permit the processor to sample the state of the discfile system. The DATAO 270 instruction is used to send a discfile address to the discfile control. The CONO 270 instruction sends control information to the discfile control.

a DATAI 270 Instruction (Figure FD-D-270-0-FD-Sheet 3-A2, 3)

The DATAI 270 instruction permits the processor to sample the contents of the data accumulator DA0-19, and the contents of the bit-shift counter BSC0-4. The bit assignments pertaining to this instruction are listed in Chapter 7, Table 7-1. This instruction is used primarily for maintenance purposes. During actual operating sequences there is usually no requirement to sample the accumulator or the bit-shift counter at the processor.

b CONI 270 Instruction (Figure FD-D-270-0-FD-Sheet 3-D5-7)

The CONI 270 instruction permits the processor to sample the contents of the command buffer, the file-control octoflop, the nine error-status flags, and the relay-meter test switches. The bit assignments pertaining to the instruction are listed in Chapter 7, Table 7-2. The CONI 270 instruction is perhaps most frequently used for sampling the file-control octoflop and the error-status flags, but like the DATAI 270 instruction described above, the CONI 270 is also used extensively for maintenance purposes.

c DATAO 270 Instruction (Figure FD-D-270-0-FD-Sheet 1)

The Discfile Control 270 can accept an address only during certain portions of its operating cycle. These portions of the operating cycle correspond to the IDS and ADT states of the file-control octoflop (described in Chapter 3 below). The IDS state of the octoflop is frequently used to initiate a priority interrupt request, thus notifying the programmer that the discfile control has completed the preceding operation and is ready to receive a new data address. The CONI 270 instruction provides an alternate means of testing the state of the octoflop and the readiness of the system to receive a new address.

The DATAO 270 instruction transmits 19 discfile address bits to the discfile control. Eighteen of these nineteen bits specify the location of the next data sector to be processed; the nineteenth bit is the read-next-sector bit. This bit determines whether the full address is to be utilized, or only that portion of the address that specifies the appropriate data track (of four or seven sectors, as the case may be). Normally, the read-next-sector bit is coded 0, the full address is used, and the discfile operates on a uniquely-addressed data sector. If, however, the read-next-sector bit is coded 1, the discfile processes the first sector of the addressed track to arrive under the read-write heads.

The DATAO 270 instruction first causes the discfile control to send a SELECT pulse to the discfile; the SELECT prepares the discfile to receive the address. The discfile control then transmits 21 address bits to the discfile (the 19 bits received from the processor, one additional bit

which is always 0, and a parity bit). After completing the address transmission, the discfile control sends the discfile an ADDRESS TRANSMITTED pulse; this pulse disconnects the discfile from the discfile control and initiates the address seek operation. The bit assignments pertaining to the DATAO 270 instruction are listed in Chapter 7, Table 7-3 .

d CONO 270 Instruction (Figure FD-D-270-0-FD-Sheet 3-A-D1, 2)

The processor sends 15 bits of control information to the discfile control at each CONO 270 instruction. Twelve of these 15 control bits are loaded into the command buffer. The remaining three control bits set no flip-flops in the discfile control, but are instead used directly as control inputs.

Functions Controlled by CONO 270 Coding

The functions of the Discfile Control 270 that are controlled by the coding of the CONO 270 instruction include:

1. The choice of the command to be executed (read, write or read compare).
2. Choice of whether or not the command is to be continued beyond the end of the first sector processed. (In the case of the read command only, whether or not the command is to be terminated at the time of the CONO 270 instruction.)
3. Choice of whether or not to remove power from the addressed discfile positioner at the end of the command.
4. Specification of the particular conditions which are to be allowed to initiate a priority interrupt request.

5. Selection of the priority interrupt channel to be assigned to the discfile control.

6. Choice of whether or not to clear the nine error-status flags.

7. Two maintenance functions. These functions provide program control of the GIVE A CHARACTER and TAKE A CHARACTER pulses sent to the Data Control 136.

The bit assignments pertaining to the CONO 270 instruction are listed in Chapter 7, Table 7-4 .

Command Repertoire

By suitable coding of the two command-code bits of the CONO 270 instruction (bits 25 and 26), the processor can cause the Discfile Control to execute any of the following three commands.

(1) Read Command - Input data is read from the discfile starting at the beginning of the 128-word sector addressed by the preceding DATAO 270 instruction. The data is loaded into DA2 as it is read in serial fashion, one bit at a time, and is then shifted right until a full 18-bit data character is stored in DA2-19. At the conclusion of this loading process, the discfile control sends a TAKE A CHARACTER pulse to the Data Control 136, causing the data control to strobe the character into its low order accumulator bits, and if necessary, to shift it left to make room for the second character of the data word. A similar process is followed for the second 18-bit data character and all succeeding data characters. Every 36 data bits from the discfile are followed by a parity bit, and the parity of each data word is checked at the discfile control.

The Data Control 136 assembles pairs of data-characters into 36-bit data words and transfers these data words into the processor. Normally this process continues until a full 128-word sector of data has been read into the processor via the data control. Unlike the write and the read compare commands, the read command can, however, be terminated at any point within a sector, or for that matter within a data word, by giving a CONO 270 command with a 1 in bit 23. Such a CONO 270 command sets the END flip-flop in the discfile control, thereby terminating the read operations almost immediately.

(2) Write Command - Output data is written at the discfile starting at the beginning of the 128-word sector addressed by the preceding DATAO 270 instruction. The Data Control 136 loads 18-bit data characters into DA2-19 of the discfile control data accumulator. The discfile control shifts these data characters right, and transmits them to the discfile in serial fashion one bit at a time. As the last bit of each data character is transmitted out to the discfile, the discfile control strobes in the next 18-bit data character from the Data Control 136, and sends the data control a GIVE A CHARACTER pulse, thus signalling the data control to shift the following character into position for transfer out. After every 36 data bits, the discfile control generates an appropriate parity bit and transmits it to the discfile. At the discfile, all 37 bits (36 data bits and the parity bit) are written out onto the addressed location. This process continues until a full 128-word sector is written. If the output data is exhausted before the end of the sector is reached, 0s are written for the rest of the sector.

(3) Read Compare Command - The read compare command compares output data from the processor with corresponding input data read from the discfile. The comparison is executed on a bit-by-bit basis, but does not include parity bits. The output data is sent to the discfile control through the Data Control 136 (as during the write command described above) and the input data is brought in from the discfile (as during the read command described above). The RCE flip-flop in the discfile control is set whenever an output data bit from the Data Control 136 fails to match the corresponding input data bit read from the discfile. The set of the RCE flip-flop can be used to initiate a priority interrupt request, thereby notifying the programmer that a read compare error has been detected.

No output parity bits are generated during read compare commands, and for read comparison purposes the incoming parity bits from the discfile are ignored. (Note, however, that the incoming data from the discfile is subject to the normal parity test during the read compare commands just as during read commands.)

Input data from the discfile is read in serially, one bit at a time, starting at the beginning of the 128-word sector addressed by the preceding DATAO 270 instruction. Eighteen-bit output data characters from the Data Control 136 are parallel-loaded into DA2-19 of the discfile control data accumulator. The discfile control shifts these output data characters right one bit at a time, comparing the contents of DA19 with the corresponding input data bit at each shift. The comparison proceeds one bit at a time, ignoring parity bits. At the first pair of data bits that are not

identical, the comparison fails, causing the RCE flip-flop to be set, and (if the programmer so desires) initiating a priority interrupt request on a predetermined priority interrupt channel.

As the last bit of each output data character is compared and shifted out, the discfile control strobes in the next 18-bit data character from the Data Control 136, and sends the data control a GIVE A CHARACTER pulse (just as during the write command described above). This process always continues until a full 128-word data sector is processed. If the output data is exhausted before the end of the input sector is reached, a read compare error results (unless the remainder of the input sector also contains only 0 data bits).

e Parity

During write commands, after transmitting each 36-bit data word to the discfile, the discfile control generates and transmits an appropriate parity bit. At the discfile, this parity bit is written onto the disc immediately after the 36 associated data bits.

During both read and read compare commands, the parity of each incoming 37-bit word is checked by the discfile control. Whenever a parity error is detected, the PER flip-flop is set. The set of the PER flip-flop can be used to initiate a priority interrupt request, thereby notifying the programmer that a read compare error has been detected.

f Priority Interrupt

Seven priority interrupt channels are available. See Chapter 3, Priority Interrupt, for a description of interrupt conditions.

2.4 SYSTEM DATA

System operating specifications, physical characteristics, and power requirements are listed in Tables 2-4, 2-5, and 2-6 below.

TABLE 2-4 SYSTEM OPERATING SPECIFICATIONS

Storage Capacity	
Data words per sector (37-bit words)	128
Sectors per position	44
Positions per disc	64
Sectors per disc	2816
Discs per disc unit	16
Sectors per disc unit	45,056
Data words per disc unit (37-bit words)	5.76 million
Bits per disc unit	213.4 million
Disc units per Discfile 5022	1 to 4

Operating Speed		
Time per disc revolution		52 ms
Data Transfer Rates:	Inner Zone	Outer Zone
Time per bit	2.4 μsec	1.4 μsec
Time per 37-bit data word	88.8 μsec	51.8 μsec
Bit frequency (nominal)	418 kc	714 kc

Note: Bit-to-bit timing may vary as much as ± 30% from the average of $\frac{1}{\text{Disc RPS} \times \text{bits per track}}$.

TABLE 2-4 SYSTEM OPERATING SPECIFICATIONS (continued)

Components of Access Time	
Positioner Switching Time	20-30 ms
Head Switching Time	100 μsec
Positioning Time (including confirmation time)	Minimum (1 position): 80 ms Maximum (64 positions): 225 ms
Confirmation Time (Confirmation time is required whenever positioner switching or motion takes place.)	39 msec
Latency Time	Maximum: 52 ms

Typical Total Access Times	
Access to address at same position on same disc. (Head switching time + latency time.)	Average: 26 ms
Access to address at same position on different disc. (Positioner switching time + confirmation time + latency time.)	Average: 91 ms
Random Access. (Positioner switching time + motion time + confirmation time + latency time; see Figure 2-4.)	
Average. (16-position move, average switching and latency):	202 ms
Maximum. (63-position move with worst-case switching and latency):	307 ms
Minimum. (1-position move, average switching and latency):	132 ms

TABLE 2-4 SYSTEM OPERATING SPECIFICATIONS (continued)

Recording Parameters

Track separation (center to center)	37.5 mils	
Read/write track width	25 mils	
Erase track width	40 mils	
Density of recording (average mid-zone)	Inner zone bits/inch	Outer zone bits/inch
	____	____
Header to data gap	____ inches (bits)	
Inter-sector gap	____ inches (bits)	

Priority Interrupt Conditions

When enabled by the corresponding flip-flop switch each of the following interrupt conditions can initiate a priority interrupt request on a predetermined interrupt channel (seven priority interrupt channels are available).

Interrupt Condition	Signifies	Enabled by Switch
DFR	Discfile Ready	EFR
IDS	Idle State	EIS
SEF	Sector End Flag	EES
FER	File Error	EFE
PER	Parity Error	EFE
RCE	Read Compare Error	EFE
DRL	Data Request Late	EFE

Parity

Odd; each 37-bit word contains 36 data bits followed by a parity bit.

TABLE 2-5 PHYSICAL CHARACTERISTICS

Construction of Discfile Control 270

One standard, all-steel construction DEC cabinet containing three FLIP-CHIP mounting panels.

Modules

Standard DEC FLIP-CHIP modules, series B, G, R and W.

Logic

Solid state. Transistors and crystal diodes utilizing static logic levels (0 vdc and -3 vdc).

Dimensions

	Discfile Control 270	Discfile 5022
Height	69-1/2 inches	63-1/4 inches
Width	22-1/4 inches	87 inches (including logic unit)
Depth	27-1/8 inches	38 inches
Service Clearances		
Front	8-3/4 inches	33 inches
Rear	14-7/8 inches	33 inches
Sides	-	33 inches

Weight and Loading

Discfile Control 270:	
Cabinet	160 lbs.
Logic (approx.)	60 lbs.
Total	220 lbs.

TABLE 2-5 PHYSICAL CHARACTERISTICS (continued)

Weight and Loading (continued)

Discfile 5022	
Logic Unit	490 lbs.
(Floor Loading	134 psf)
Disc Unit	2575 lbs.
(Floor Loading	107 psf
Pad Loading	33 psf)
Total	3065 lbs.

Environmental Requirements

Discfile Control 270:		
Operating Temperature	Maximum:	100°F
	Minimum:	50°F
Discfile 5022:		
Operating Temperature	Maximum:	85°F
	Minimum:	60°F
Relative Humidity	Maximum:	80 %
	Minimum:	20 %

TABLE 2-6 POWER REQUIREMENTS

Discfile Control 270:	
Power Equipment	DEC Power Supply Type 728 DEC Power Control Type 834-836
Line Voltage Input	105 to 125 volts, 60 cycle, single phase
Power Plug	Hubble Twist-Loc 3-prong, 30 ampere, 250 volt
Current Consumption	Surge: 4.5 ampere Normal: 3 ampere
Power Consumption	360 watts
Heat Dissipation	1230 btu/hour
Discfile 5022:	
Line Voltage Input	208 volt ± 10%, 3 phase, 60 cps
Current Consumption	Start current: 35 ampere maximum on any one line. Run current: 15 amp. maximum on one line, 10 amp. maximum on other two lines.
Power Consumption	Logic unit and one disc unit: 4.5 kw. Additional disc units 3.5 kw each.
Heat dissipation	Logic unit and one disc unit: 15,300 btu/hour. Each additional disc unit 11,900 btu/hour.

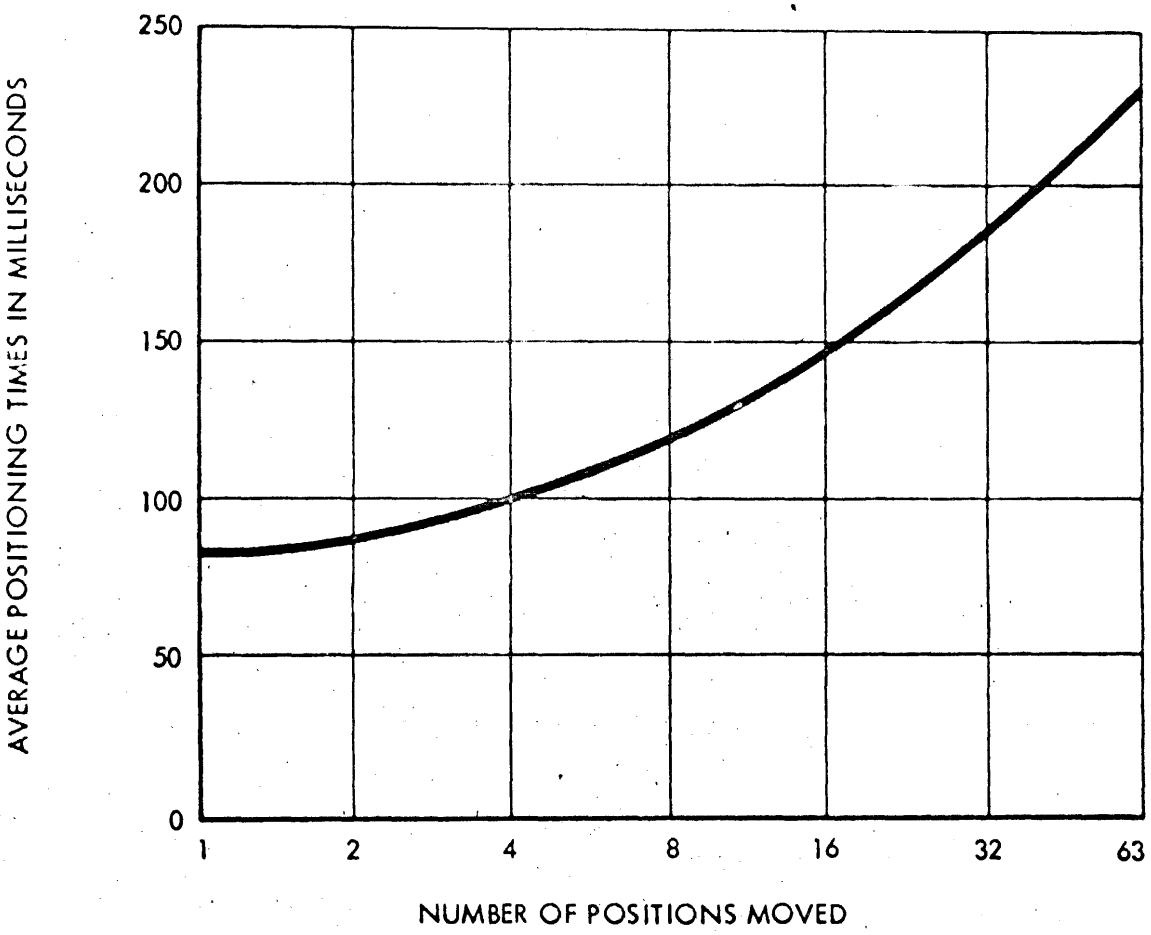


FIGURE 2-5

TYPICAL POSITIONING TIMES INCLUDING CONFIRMATION

CHAPTER 3

SYSTEM LOGIC

This chapter is organized horizontally by areas of the system logic. Its purpose is to present a brief overall description of the major signals and circuits used in the Discfile Control 270. All sections of this chapter include tables listing the signals relevant to the section by mnemonic abbreviation. Listed with each signal is the engineering logic diagram designation and coordinate at which the signal is generated. This tabular listing directly corresponds to the entry headings in the Chapter 8 signal glossary. Each signal is there described in full detail.

It is not recommended that the reader attempt to refer to a large number of the glossary entries while reading this chapter for the first time. For most people a more efficient approach is to read this chapter rapidly to get a general understanding of the various functional areas of the system; then to read Chapter 4, System Operating Sequences, to learn the step-by-step time sequence of the major system operations; and only then to return for a second pass through the two chapters. On this second pass it may be helpful to refer to the glossary whenever the operation of a given circuit or the execution of a given operation seems unclear.

3.1 INTERFACE SIGNALS

The Discfile Control 270 interacts with three other systems: 1) the PDP-6 Processor; 2) the Data Control 136; and 3) the Discfile 5022.

a Processor Interface

This paragraph describes the control signals applied to the discfile control from the processor.

TABLE 3-1 PROCESSOR SIGNALS TO DISCFILE CONTROL

When Occurs	Control Signal from Processor	Signal Produced Within the Discfile Control 270
INITIAL CLEAR:	IOB RESET (-SC:C1)	IBR (SC:C1) CCB (-CM:C4) CDA (-AD:B8) FCL (-AD:C5) SCL (-SC:C6)
ALL COMMANDS:	IOS3-9 (-SC:D1)	SEL (-SC:D2)
CONO:	IOB CONO CLR (-CM:C3) IOB CONO SET (-CM:D3)	CCB (-CM:C4) PTC (-CM:C4)
DATAO:	IOB DATA CLR (-AD:B1) IOB DATA SET (-AD:B1)	CDA (-AD:A8) PTA (-AD:B8)
CONI:	IOB STATUS (-CM:D7)	STP (-CM:C8)
DATAI:	IOB DATA I (-AD:A1)	DTP (-AD:A8)

The IOB RESET pulse (and the resulting IBR) are generated when computer power goes on or when the operator presses the IO reset key on the computer console. The program may also generate a reset. The

IBR pulse and the four additional clear pulses it initiates (CCB, CDA, FCL and SCL) clear the entire discfile control.

The discfile control responds to the four IOT instructions (CONO, DATAO, CONI, and DATAI) only when the 14 IOS lines represented by IOS3-9 assert the device code 270. Whenever the device code portion of an IOT command contains 270, the SEL level is asserted within the discfile control. The SEL level enables the control to respond to the processor control signals associated with the instruction.

The CONO 270 instruction applies two control pulses to the discfile control one microsecond apart. The IOB CONO CLR pulse initiates CCB, which clears the discfile command buffer. The IOB CONO SET pulse then initiates PTC, which loads new control information into the discfile command buffer.

The DATAO 270 instruction also applies two control pulses to the discfile control one microsecond apart. The IOB DATA CLR pulse initiates CDA, which clears the discfile data accumulator DA0-19. The IOB DATA SET pulse then initiates PTA, which loads IOB17-35 into the data accumulator; this data provides 19 bits of the 21-bit discfile address. One of the remaining two bits is always 0, and thus is generated within the discfile control. The final bit is the address parity bit, which is also generated within the discfile control.

The CONI 270 and DATAI 270 instructions permit the processor to sample the state of the discfile control. The 2.5-μsec IOB STATUS level enables STP to gate out a variety of CONI status information (including the

contents of the command buffer, the file-control octoflop, the error-status flags, and the relay-meter test switches). Similarly, the 2.5-μsec IOB DATA I level enables DTP to gate out the contents of the data accumulator DA0-19 and the bit shift counter BSC0-4.

b Data Control 136 Interface

The discfile control applies the following signals to the Data Control 136:

GIVE A CHARACTER (-SC:C7)

TAKE A CHARACTER (SC:C7)

The GIVE A CHARACTER pulse signals the data control that the discfile control has accepted the 18-bit output character currently stored in the high order end of the data control data accumulator, and that the data control must bring another character into position for transfer out to the discfile control. The TAKE A CHARACTER pulse signals the data control that the discfile control has presented an 18-bit input character to the low order strobe data inputs of the data control data accumulator. This indicates to the data control that it may strobe the character into the low order bits of the data control data accumulator.

The Data Control 136 applies the following signals to the discfile control:

DA RQ (-CM:C1)

SEL 5 (-SC:B7)

During the output commands, write and read compare, the assertion of the DA RQ level indicates that the data control has sent out the last of

its output data. The discfile control uses the DA RQ level to detect the end of output data and to terminate the output command.

If the Data Control 136 has not been selected to the discfile in time for the beginning of a given command, the negation of the SEL 5 level causes an error condition.

c Discfile 5022 Interface

The listing below summarizes the exchange of control information and data between the discfile and the discfile control.

TABLE 3-2 CONTROL SIGNALS TO THE DISCFILE FROM THE DISCFILE CONTROL

Signal	Function
MCL (-CM:D7)	Master Clear signal generated by pushbutton at the discfile control. Clears the discfile logic.
SELECT (-TR:A6)	Prepares the discfile to receive and store an address.
ALERT (-TR:A7)	Prepares the discfile logic to receive the forthcoming command signal (READ or WRITE).
READ (-TR-A7)	Commands the discfile to read a sector.
WRITE (-TR:A7)	Commands the discfile to write a sector.
END (-TR:A5)	Disconnects the discfile from the discfile control. After receiving END, the discfile is ready to receive a new SELECT.
CLEAR (-TR:A4)	Starts the turn-off of positioner power at the discfile. An END always accompanies a CLEAR, but the converse is not true.

TABLE 3-3 CONTROL SIGNALS FROM THE DISCFILE TO THE DISCFILE CONTROL

Signal	Function
READY (-TR:A4)	Indicates that the discfile has found the header of the "get ready" sector next preceding the sector to be operated upon, and is ready to be alerted for a read or write command.
WRITE CLOCK EVEN (-TR:D5)	During write operations, the discfile sends the discfile control one of these pulses to clock out every even-numbered bit.
WRITE CLOCK ODD (-TR:D5)	During write operations, the discfile sends the discfile control one of these pulses to clock out every odd-numbered bit.
ERROR SIGNAL (-SC:B1)	Indicates that an error has been detected at the discfile.
WRITE LOCKOUT WARNING (-SC:B1)	Indicates that the processor has addressed a locked out disc.
ALARM (-SC:B2)	Indicates presence of an alarm condition at the discfile.
OPERABLE (-SC:C2)	Indicates that the discfile is operable, i.e. ready to receive commands from the discfile control.

TABLE 3-4 DATA SIGNALS TO OR FROM THE DISCFILE

Signal	Meaning
EDO (-TR:C3)	Even-numbered input or output bit a 1.
EDZ (-TR:C4)	Even-numbered input or output bit a 0.
ODO (-TR:C1)	Odd-numbered input or output bit a 1.
ODZ (-TR:C2)	Odd-numbered input or output bit a 0.

3.2 COMMAND LOGIC

The command logic accepts control information from the PDP-6 processor, decodes the information, and applies the resulting command levels to various circuits within the discfile control. The command logic includes an 11-bit command buffer register and a decoding network.

TABLE 3-5 COMMAND LOGIC SUMMARY

CM0-1 (-CM:B1, 2)	CCB (-CM:C4)	CLR (-CM:B3)
NOP (-CM:A2)	PTC (-CM:C4)	END flip-flop (-CM:B4)
RED (-CM:A2)	CCE (-CM:D2)	EFE (-CM:B4)
WRT (-CM:A1)		EES (-CM:B5)
RDC (-CM:A1)		EIS (-CM:B6)
		PIA0-2 (-CM:B7, 8)

During the CONO 270 command, the contents of the CONO effective address determines the control information that is loaded into the command buffer. The two most significant bits of the command buffer are CM0 and CM1. The contents of these two bits determine the nature of the command to be performed by the discfile. The outputs of the CM0 and CM1 flip-flops are used directly to control certain functions of the discfile control. These outputs are also applied to a half binary-to-octal decoder which in turn produces one of the four outputs, depending on the contents of CM0-1:

TABLE 3-6 COMMAND CODE DECODING

CM0-1 Contain	Command Level Asserted
0	NOP - No operation.
1	RED - Read command.
2	WRT - Write command.
3	RDC - Read compare command.

The NOP, RED, WRT, and RDC command levels are used widely in the discfile control to govern the operations performed during the corresponding operating sequences.

During the CONO 270 command, a CCB pulse clears the command buffer, and then one microsecond later a PTC pulse loads new control information into the buffer. The two command-code bits CM0 and CM1 cannot be affected by this operation unless the change commands enable level CCE is asserted. This level is only asserted during those portions of the discfile control operating cycle when it is appropriate to change commands.

Whenever the END flip-flop is set, the command code bits CM0-1 are automatically cleared at the end of the first data sector to be processed; two μsec later an END pulse is sent to the discfile. For the write and read compare commands, the current sector will continue for the full 128 words, regardless of whether or not END is set. For the read command, however, the command is terminated immediately after END is set, and the command code bits are cleared at that time. Whenever the CLR flip-flop is set, the END flip-flop is automatically set at the same time. An END pulse is sent to the discfile at the end of any sector during which the END flip-flop is set; if CLR is set, a CLEAR pulse is sent to the discfile as well.

The remaining seven bits of the command buffer are associated with the priority interrupt system. The four flip-flop switches EFE, EES, EFR, and EIS determine which interrupt conditions are to be enabled; the

three priority interrupt assignment flip-flops PIA0-2 determine the priority interrupt channel to be used by the discfile control (and hence the priority of any interrupt break which the discfile control may request).

3.3 FILE-CONTROL LOGIC

The file-control logic consists of an 8-state control device (the file-control octoflop) and associated delays and control networks. The file-control logic governs the transmission of control pulses to the discfile and also controls various operations that occur within the discfile control.

a Octoflop Structure

The file-control octoflop is composed of two R284 quadraflops. In engineering logic diagram BS-D-270-0-FC the octoflop is shown by a schematic representation resembling eight DEC flip-flops connected in a ring. Logically, the octoflop can be regarded as though it were a set of eight flip-flops one of which (and only one) is in the one state, and the remaining seven of which contain 0. When any one of the eight "flip-flops" is set to the 1 state, the remaining seven are reset to the 0 state.

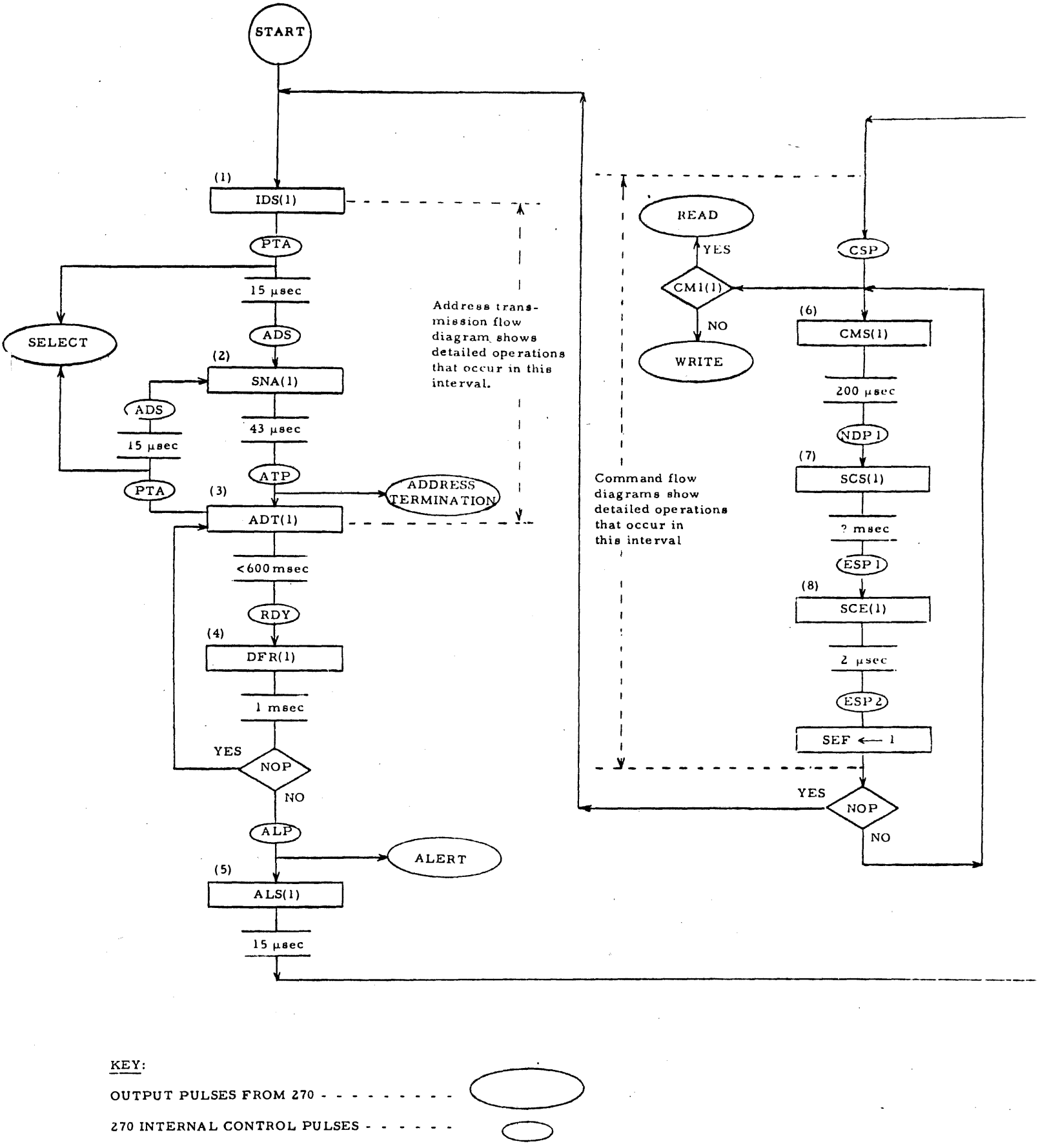
TABLE 3-7 FILE CONTROL LOGIC SUMMARY

Octoflop State	Set at Pulse
1) IDS (-FC:B2)	IBR (-SC:C1), ESP 2 (-FC:C8)
2) SNA (-FC:B2)	ADS (-FC:C2)
3) ADT (-FC:B3)	ATP (-FC:C3)
4) DFR (-FC:B4)	RDY (-TR:C5)
5) ALS (-FC:B4)	ALP (-FC:B4)
6) CMS (-FC:B5)	CSP (-FC:C4)
7) SCS (-FC:B6)	NDP 1 (-FC:B6)
8) SCE (-FC:B7)	ESP 1 (-FC:C6)
Flip-flop	**Set at Pulse**
SEF (-SC:A7)	ESP 2 (-FC:C8)
Other	
PTA (-AD:B8)	
FCL (-AD:C5)	
NDP 2 (-FC:B8)	
END flip-flop (-CM:B4)	

b Octoflop Cycle

A detailed flow diagram of the octoflop cycle is shown in Figure 3-1. When the system is turned on, the IBR pulse resets the octoflop to IDS (idle state). This is the quiescent condition that the octoflop holds while the discfile control is waiting for an address.

The DATAO 270 instruction is used to send a new address to the discfile control. At the IOB DATAO SET pulse, the discfile control produces



KEY:

OUTPUT PULSES FROM 270 - - - - - - - - -

270 INTERNAL CONTROL PULSES - - - - - -

FIGURE 3-1 FILE-CONTROL OCTOFLOP CYCLE

a PTA pulse. This PTA sends a SELECT to the discfile and starts a 15-μsec delay. At the termination of this delay, an ADS pulse is generated (to start the address clock) and the octoflop is advanced to its second state SNA (Select New Address). During the SNA state, 21 address bits are clocked out to the discfile in a sequential transfer, one bit each 1.33 μsec. When the final address bit is transferred out, a 15-μsec delay is started. The termination of this delay produces the ATP pulse; ATP is thus generated about 43 μsec after the octoflop enters SNA state (28 μsec + 15 μsec = 43 μsec).

The ATP pulse steps the octoflop to its third state ADT (Address Terminated). During the address terminated state, the discfile seeks the sector specified by the address that it received during SNA. A different address can be sent to the discfile control by programming another DATAO instruction while the control is in ADT state. This induces a seek interrupt, and returns the octoflop to SNA state while the new address is transferred out to the discfile.

When the discfile finds the addressed sector (the search normally requires less than 600 ms) a RDY pulse advances the octoflop from ATP to its fourth state DFR (Discfile Ready). The octoflop remains in DFR state for 1 msec. If no command is present in CM0-1, NOP is asserted, and the octoflop returns to ADT state at the termination of DFR state. The octoflop then oscillates between ADT state and DFR state, advancing to DFR each time the addressed sector is sensed, and returning to ADT after a one-millisecond delay.

If, however, there is a command code present in CM0-1, an ALP pulse is generated at the expiration of DFR state. The ALP pulse sends an ALERT to the discfile, and advances the octoflop from DFR state to its fifth state ALS (Alert State). The octoflop remains in ALS state for 15 μsec.

A CSP pulse is generated at the expiration of the 15-μsec ALS state delay. This CSP pulse sends a READ or a WRITE command pulse to the discfile (the choice depending on the contents of command code bit CM1) and advances the octoflop from ALS state to its sixth state CMS (Command Selected). The octoflop remains in CMS state for 200 μsec.

An NDP 1 pulse is generated at the expiration of the 200-μsec CMS delay. This NDP 1 pulse advances the octoflop from CMS state to its seventh state SCS (Sector Started). The NDP 1 pulse advances the octoflop from CMS state to its seventh state SCS (Sector Started). The NDP 1 pulse also generates an FCL pulse, and,after a one-μsec delay, an NDP 2 pulse. (These pulses are used to initialize the bit shift counter BSC0-4 in preparation for the forthcoming data sector.)

The octoflop is set to SCS state before the first bit of the sector to be processed is detected. (Note that the RDY pulse that stepped the octoflop to DFR state is initiated by the header of the "get ready" sector immediately preceding the sector to be processed; hence there is an appreciable delay between the time when the octoflop is stepped to SCS state and the time when the first data bit of the next sector is read or written.

During the write and read compare commands, the octoflop remains in SCS state throughout the entire interval that is required to process the 128 data words of the current sector. Five microseconds after the final (37th) bit of the final (128th) word is processed, an integrating delay times out, producing the ESP 1 pulse. The ESP 1 pulse advances the octoflop from SCS to its eighth and final state SCE (Sector End).

During the read command, as during write and read compare, the ESP 1 pulse always occurs at the end of the 128-word data sector. However, unlike the write and read compare commands, the read command can also be terminated at any point within a sector by setting the END flip-flop. Approximately five μsec after END is set, the integrating delay times out, producing the ESP 1 pulse. As in the case of the ESP 1 that occurs at the end of the sector, the ESP 1 that is induced by setting END also advances the octoflop from SCS to the SCE (Sector End) state.

The octoflop remains in SCE state for only two μsec. Two μsec after the ESP 1 pulse advances the octoflop to SCE state, the ESP 2 pulse terminates the SCE state. If ESP 1 has cleared CM0-1 (indicating that no further sectors of data are to be processed) NOP is asserted, and ESP 2 resets the octoflop to IDS state. However, if more data is to be processed, the current command code is left in CM0-1, NOP is not asserted, and the ESP 2 pulse returns the octoflop to CMS state in preparation for processing the next sector of data.

Since the 2-μsec SCE state is too brief to be available for the programmer to sample, the ESP pulse that terminates SCE state sets the sector end

flag SEF to let the programmer know that the end of the sector has been passed. The SEF remains set until it is reset by an SCL pulse.

3.4 DATA LOGIC

During the read command, the discfile control receives 37-bit data words from the discfile through a sequential bit-by-bit transfer. These words are shifted right in the data accumulator DA2-19 and assembled into 18-bit data characters. Two such characters are parallel-transferred out to the Data Control 136 for each 37-bit input data word. The 37th bit of the input word is a parity bit. The discfile control must distinguish this bit from the data bits and apply it only to the parity accumulator flip-flop PAR, but not to the data accumulator.

During the write command and the read compare command, this process is essentially reversed. The Data Control 136 loads 18-bit data characters into the discfile control data accumulator DA2-19 by parallel transfer. The characters are shifted right and are sequentially available bit-by-bit at DA19. For the write command, two full 18-bit characters are shifted out to the discfile control from DA19, and then a parity bit is sent out from PAR, thus writing a full 37-bit data word. For the read compare command, each data bit is compared on a bit-by-bit basis with a corresponding bit read from the discfile. Every 37th incoming bit from the discfile is a parity bit. These bits are used to check the parity of the incoming data during the read compare, but they are not included in the read comparison operation.

The process of transmitting an address out to the discfile is somewhat similar to the write command, but 20 bits rather than 36 are shifted out before the final parity bit is sent to the discfile. Furthermore, the address source is different from that of the data. Data arrives in 18-bit increments from the Data Control 136 and is loaded into DA2-19 for shifting out the discfile. The address comes directly from the processor to the discfile control without being routed through the Data Control 136. Nineteen of the 21 address bits are loaded into DA0-11 and DA13-19 from the IO bus. (The address bit corresponding to DA12 is always 0, and so can be generated within the discfile control.) The 21st bit of the address is the parity bit. This address parity bit is generated by the discfile control, and is sent out to the discfile as the final bit of the address.

The data logic governs all the discfile control operations described above. The data logic is composed of three basic functional sections: 1) data clock logic; 2) data receiving and transmitting logic; and 3) bit, character, and word control logic.

a Data Clock Logic

At each address or data bit, the data clock circuitry generates a sequence of five separate clock pulses: DCK 1, DCK 0, DCK 2, DCK 2.1 and DCK 3. With the exception of DCK 0, which is used only for loading purposes within the clock chain itself, all of these clock pulses initiate various logical functions within the discfile control.

TABLE 3-8 DATA CLOCK LOGIC SUMMARY

Data Clock Chain	Clock for Address Transmission
DCK 1 (-TR:B7)	CLK (-AC:D8)
DCK 0 (-TR:B7)	ADS (-FC:C2)
DCK 2 (-TR:C8)	ACE (-AC:D7)
DCK 2.1 (-TR:C8)	ACS (-AC:D8)
DCK 3 (-TR:C8)	WCE (-TR:D7)
Clock for Write Command	**Clock for Read and Read Compare Commands**
WRITE CLOCK EVEN (-TR:D5)	RDS (-TR:C6)
WRITE CLOCK ODD (-TR:D5)	CM1(1) (-CM:B2)
WCE (-TR:D7)	SCS(1) (-FC:B6)
WCO (-TR:C6)	WCT7(0) (-WC:A7)
RDS (-TR:C6)	END flip-flop (-CM:B4)
WCT7(0) (-WC:A7)	EDO (-TR:C3)
	EDZ (-TR:C4)
	ODO (-TR:C1)
	ODZ (TR:C2)

During address transmission, the data clock chain is triggered by CLK pulses generated by an R401 variable clock circuit. The ACE and ACS flip-flops make up a two-stage synchronizing network that controls the turn-on and turn-off of the address clock. The same ADS pulse that steps the file-control octoflop to SNA state also sets the ACE flip-flop. The first subsequent CLK pulse sets the ACS flip-flop. Each CLK pulse to arrive while ACS remains set is gated through to produce a WCE clock; that clock in turn triggers the DCK clock chain.

During the read and read compare commands, the data clock chain is triggered by incoming data pulses from the discfile. The four data pulse

inputs EDE, EDO, ODO and ODZ, are ORed together, and provided that the clock is enabled, each data pulse to arrive on any of the four inputs triggers the data clock chain. During read and read compare operations, the clock is enabled by the assertion of the RDS level. The RDS level is asserted when CM1(1) is asserted (specifying a read or read compare command) provided that SCS(1) is asserted (file-control octoflop in SCS state) and WCT7(0) is asserted (sector not yet completed, and if the command is read, END not set).

During the write command, the data clock chain is triggered by WRITE CLOCK EVEN and WRITE CLOCK ODD pulses from the discfile. These WRITE CLOCK pulses are inverted in the discfile control and redesignated WCE and WCO. Provided that RDS is negated (which it is during any write command), and that WCT7(0) is asserted (indicating that the end of the 128-word data sector has not yet been reached), each WRITE CLOCK EVEN and WRITE CLOCK ODD starts the sequence of DCK pulses.

As indicated by the above description, each address bit or data bit processed by the discfile control triggers the DCK clock chain and produces the five-pulse sequence: DCK 1, DCK 0, DCK 2, DCK 2.1, and DCK 3. The DCK entry in the Chapter 8 signal glossary includes a comprehensive table listing all functions performed by each of the five DCK pulses produced by the data clock chain. This table can be referred to to help resolve most questions involving the relative timing of the ordered logical steps making up the operating sequences of the discfile control.

b Data Receiving and Transmitting Logic

Addresses and data are transmitted to and from the discfile on four twisted-pair transmission lines. These four twisted-pair lines are used in sets of two; one line of each set is pulsed for a 1; the other line for a 0. Two such sets of lines are used. This allows alternate bits (odd and even) to be transmitted on different lines, thus halving the data transmission frequency. The even data lines are designated EDO and EDZ (Even Data One and Zero); the odd data lines are designated ODO and ODZ.

TABLE 3-9 DATA RECEIVING AND TRANSMITTING LOGIC SUMMARY

Discfile Interface	Within Discfile Control
EDO (-TR:C3)	RDO (-TR:D2)
EDZ (-TR:C4)	RDZ (-TR:D3)
----------	RDS (-TR:C6)
ODO (-TR:C1)	----------
ODZ (-TR:C2)	SDR (AD:B8) and -AC:C1)
	PAR (-AC:A8)
	MSB (-TR:A2)

During the write command the WCO and WCE pulses clock alternate bits out to the discfile. Data bits are taken from DA19, parity bits are taken from the PAR flip-flop. Each outgoing bit is amplified by a pulse amplifier and sent to the discfile over the appropriate twisted-pair transmission line. The odd-numbered bits are clocked by WCO, and transmitted

The MSB line provides a terminating supply for the transmission line

over the ODO or ODZ twisted pair (the choice depending on whether the bit is a 1 or a 0). Similarly, the even-numbered bits are transmitted over the EDO and EDZ twisted pairs.

For address transmission somewhat similar operations are executed. The major difference between the transmission of data and the transmission of address information is that no WCO pulses are generated during address transmission. Consequently all 21 address bits are clocked out by the WCE pulses and are transmitted to the discfile over only two of the four twisted-pair transmission lines, the EDO and EDZ lines.

During the read and read compare commands, the process described above is essentially reversed. Data from the discfile enters the discfile control on the four twisted-pair transmission lines. The incoming data pulses on each of the four lines are passed through a pair of series-connected inverters to ensure that they are at the correct voltage reference for use within the discfile control logic. Provided that the RDS level is asserted (indicating that the incoming pulses should be accepted by the discfile control) the odd and even pulses are ORed together to produce RDO and RDZ pulses.

An RDO pulse is generated whenever a 1 bit is read in from the discfile (on either the EDO or ODO line), and an RDZ pulse is generated whenever a 0 bit is read in on either EDZ or ODZ. During the read command, the RDO and RDZ pulses load 1s and 0s respectively into bit DA2 of the data accumulator. During read compare, the RDO pulses complement the RCT

flip-flop. (This is the first complement operation of the read compare operation; provided that DA19 contains a 1, a second complement operation occurs at the SDR pulse.)

An SDR pulse is generated at the DCK 2 pulse corresponding to every address or data bit except parity bits. Besides complementing the RCT flip-flop when DA19(1) is asserted, the SDR pulse shifts the contents of DA0-19 right one bit position. During address transmission only, the contents of DA19 are ring-shifted back into DA0. (Since DA0-19 is directly accessible to the processor by means of the DATAI 270 instruction, this ring-shift feature permits a convenient check on the operation of the accumulator shift logic. It has no direct effect on the address transmission sequence.)

c Bit, Character, and Word Control Logic

The bit, character, and word control logic is composed of two counters (BSC0-4 and WCT0-7) and two flip-flops (COH and WDC). These circuits control the sequencing of most operations performed by the discfile control.

TABLE 3-10 BIT, CHARACTER AND WORD CONTROL LOGIC SUMMARY

Control Circuits	Levels
BSC0-4 (-AC:C2-5)	BCO (-AC:D1)
WCT0-7 (-WC:A3-7)	WCT7(0) (-WC:A7)
COH (-AD:C6)	RDS (-TR:C6)
WDC (-AD:C7)	
END (-CM:B4)	

Pulses

TAKE A CHARACTER (-SC:C7)

DTC (-AD:D2)

SND (-AD:C8)

GIVE A CHARACTER (-SC:C7)

ESP 1 (-FC:C6)

ESP 2 (-FC:C8)

The bit-shift counter BSC0-4 is a 5-bit count-of-20 binary counter which starts counting in either the 0 state or the 2 state depending upon whether the discfile control is sending out an address or is executing a data operation (a read, write, or read compare command). During address transmission, the bit-shift counter counts the number of address bits that have been clocked out to the discfile; the overflow level BCO is asserted at the 20th bit. For data operations, the BCO occurs at the 18th bit of each 18-bit data character.

One of the functions that the bit-shift counter performs is to control the parallel transfer of 18-bit data characters between the discfile control and the Data Control 136. During read commands, the TAKE A CHARACTER

pulse is generated at the same DCK 1 pulse that initiates the BCO level. This DCK 1 pulse indicates that the 18th and final bit of the incoming data character has been stored in bit DA2 of the discfile control data accumulator. The TAKE A CHARACTER pulse can thus signal the data control to strobe the character into the low order bits of the data control data accumulator.

Similarly, during the write and the read compare commands, the BCO level indicates that the last bit of the previous character has been shifted out of DA19, and that the discfile control is free to receive another data character from the Data Control 136. The BCO level then enables the DCK 2.1 to initiate a DTC pulse. The DTC pulse in turn produces an SND pulse and a GIVE A CHARACTER pulse. The SND pulse causes the contents of the 18 high order bits of the data control data accumulator to be loaded into bits DA2-19 of the discfile control data accumulator. The GIVE A CHARACTER pulse signals the data control to advance the next 18-bit output character into position for the next output transfer.

The COH and WDC flip-flops are used in conjunction with the bit-shift counter to enable the discfile control to distinguish between parity bits and other address or data bits. These circuits interact in such a manner that the COH flip-flop contains 1 only when an address parity bit (the 21st bit of each address) or a data word parity bit (the 37th bit of each data word) is being, or is about to be, processed; during the processing of all other address or data bits, the COH flip-flop remains in the 0 state.

The WDC flip-flop indicates which of the BCO signals precede parity bits. The WDC is set throughout every address transmission. It is so set because the bit counter overflow that occurs at the 20th address bit always immediately precedes the arrival of the address parity bit (the 21st and final address bit). However, during the data operations (read, write, and read compare) every BCO does not signal the imminent arrival of a parity bit. A separate BCO signal is produced at the end of each of the two 18-bit characters making up the data word. Consequently, during data operations, the WDC flip-flop must be used as a count-of-two counter to enable the discfile control to discriminate between the first data character of the data word and the second. Only the second data character is followed by a parity bit. Therefore the COH flip-flop is set to 1 only at the end of that character.

The WDC flip-flop also determines when the word counter WCT0-7 is to be incremented. The word counter is not incremented at every character, but only at every other character, i.e. at every character during which WDC contains 1. The word counter thus counts the number of 37-bit data words that have been processed during the current data sector. When the word counter reaches its final value of 128 (or, in the case of the read command only, when the END flip-flop is set), the WCT7 flip-flop is set to 1. This ends the assertion of WCT7(0) level, and thus terminates the RDS level. The negation of RDS prevents the generation of any further RDO or RDZ pulses and also shuts off the DCK clock chain.

Five μsec after the last DCK 1 pulse, the R303 integrating delay AB01 times out, generating an ESP 1 pulse which advances the file-control octoflop to SCE state. The ESP 2 pulse that follows 2 μsec later either returns the octoflop to CMS state (if additional sectors of data are to be processed), or (if no further sectors are to be processed) resets the octoflop to IDS state, thus ending the current command.

3.5 ALARM AND ERROR DETECTION LOGIC

The discfile closes the ALARM relay (applying an ALARM level to the discfile control) when positioner power fails below normal, when there is a malfunction in the flying-head air supply, when ambient temperature rises too high, or when a test mode switch is left in test position during normal operate mode.

The discfile will not close the OPERABLE relay when an ALARM condition is present, nor will it do so if the logic unit or any disc unit is in test mode, nor if the POWER OFF pushbutton at the logic unit is depressed. The OPERABLE condition indicates that there is no ALARM, but the absence of an ALARM does not necessarily show that the discfile is OPERABLE.

TABLE 3-11 ALARM AND ERROR DETECTION LOGIC SUMMARY

Levels	Error-Status Flags	Parity Accumulator Flip-flop
ALARM (-SC:B2)	FER (-SC:A2)	PAR (-AC:A8)
ALM (-SC:B4)	WLE (-SC:A3)	Read Compare Test Flip-flop
OPERABLE (-SC:C2)	ADE (-SC:A4)	RCT (-SC:A8)
OPR (-SC:C5)	CME (-SC:A4)	
MRB (-SC:D3)	DCE (-SC:A5)	
MRG (-SC:D3)	PER (-SC:A5)	
Pulses	RCE (-SC:A6)	
WRITE LOCKOUT WARNING (-SC:B1)	DRL (-SC:A6)	
ERROR SIGNAL (-SC:B1)	Flip-flop Switch	
ERP (-SC:B5)	EFE (-CM:B4)	

The ALARM and OPERABLE levels are each applied to a switch filter and two series-connected inverters. The resulting logic levels are redesignated ALM and OPR. Whenever the OPR level is negated or the ALM level is asserted, the FER flip-flop is set. The FER flip-flop is also set under various error conditions described below; it thus assembles a number of alarm and error conditions which can be used to request a priority interrupt break.

The discfile sends a WRITE LOCKOUT WARNING to the discfile control whenever the processor has addressed a locked out disc. This pulse arrives at the discfile control with or before the READY pulse and immediately sets both the FER flip-flop and the WLE flip-flop. If the command is a read or read compare, there is no difficulty reading the data

from the locked out discs, and no further error indications result. However, if the discfile control should attempt to execute a write command in spite of the WRITE LOCKOUT WARNING, then the writing or erasure of data is prevented and a data check error is generated. The data check error produces a DC error indication at the discfile and also sends an ERROR SIGNAL to the discfile control.

When the discfile detects any error condition, it sends an ERROR SIGNAL to the discfile control. There the ERROR SIGNAL is inverted and redesignated ERP. The ERP pulse always sets the FER flip-flop. If an ERP pulse occurs during the ADT state of the file-control octoflop, it sets the ADE flip-flop as well as the FER; if it occurs during CMS state, it sets the CME flip-flop; and if it occurs during SCS state, it sets the DCE flip-flop. In this way, the error-status flags FER, ADE, CME and DCE not only tell the programmer that an error has been detected, but furthermore when the error has appeared, and therefore what general class of error has occurred. Indicators at the discfile itself can further categorize the error.

There are three additional error conditions that arise within the discfile control rather than at the discfile. These three conditions are parity error, read compare error, and data request late. The FER flip-flop is not set by any of these conditions. The corresponding flags are designated PER, RCE, and DRL. During the read and read compare commands, the discfile control ~~tests~~ accumulates and tests the parity of each incoming data word. The PER flip-flop is set whenever the data word currently being read shows incorrect

Insert Page 3-27a here

if the Data Control 136 is not selected to the discfile control in time to operate upon a forthcoming data sector.

Provided that EFE, the Enable File Error priority interrupt switch (a flip-flop switch) is set to 1, a PIE level is produced whenever FER, PER, RCE, or DRL is set. The resulting priority interrupt request may be used to notify the programmer that an error has occurred.

The MRB (Meter Reading Bad) and MRG (Meter Reading Good) signals originate from pushbuttons at the discfile. The address sequencing relays in the discfile can readily be tested with the aid of a relay test program run from the processor. The program cycles through all the relays in predetermined order. As each relay is tested, a voltmeter at the discfile indicates whether or not its contact resistance is normal. If the contact resistance is excessive, a high meter reading shows that the relay is bad. The maintenance engineer performing the test should then press the Meter Reading Bad pushbutton; this produces an MRB level at the discfile control. If there is a low meter reading (showing normal contact resistance for the relay under test) the Meter Reading Good pushbutton is pressed to produce an MRG level at the discfile control. The CONI 270 samples the MRB and MRG levels on IOB10 and IOB9 respectively.

parity. During the read compare command, the RCT flip-flop executes the actual bit-by-bit comparison of input data to output data; the RCE flip-flop indicates errors. Whenever an input data bit read from the discfile fails to match the corresponding output bit from the Data Control 136, the RCT flip-flop causes the RCE flip-flop to be set. The DRL flip-flop is set

3.6 PRIORITY INTERRUPT

The processor can assign a priority-interrupt request channel to the discfile control by means of a CONO 270 instruction which sets PIA0-2 to some octal number from 1 to 7. If all three PIA bits are 0, no priority interrupt channel is assigned. By asserting the PIE level the discfile control can request a priority interrupt break on the assigned channel. The PI request is sent to the processor by grounding the appropriate PIR (priority interrupt request) line. The lower the channel number, the higher the priority. Breaks on channel 1 take precedence over all other breaks. Breaks on channel 2 take precedence over all breaks except those on channel 1, etc.

TABLE 3-12 PRIORITY INTERRUPT LOGIC SUMMARY

Flip-flops	Levels
	PIE (-CM:A7)
PIA0-2 (-CM:B7, 8)	PIR1-7 (-CM:A7, 8)
EFE (-CM:B4)	FER(1) (-SC:A2) PER(1) (-SC:A5) RCE(1) (-SC:A6) DRL(1) (-SC:A6)
EES (-CM:B5) EFR (-CM:B6) EIS (-CM:B6)	SEF(1) (-SC:A7) DFR(1) (-FC:B4) IDS(1) (-FC:B2)
	Pulse
	SCL

The conditions under which the PIE level is asserted are themselves subject to program control. There are four flip-flop switches which determine which of four possible priority interrupt conditions is to be allowed to assert the PIE level. Every CONO 270 instruction sets the four priority interrupt enable switches EFE, EES, EFR and EIS to the states determined by bits 29, 30, 31, and 32 respectively of the CONO 270 instruction. When a given switch flip-flop is set to 1, the corresponding interrupt condition (or conditions) is(are) gated through to assert PIE; when the switch is left in the 0 state, the input condition is not allowed to produce a PIE.

The priority interrupt conditions that can be enabled to generate PIE are summarized in Table 3-13 below.

TABLE 3-13 PRIORITY INTERRUPT CONDITIONS

CONO 270 Bit No.	Sets Switch	Permitting Interrupt Condition(s) to Assert PIE
29	EFE	FER(1) $\vee$ PER(1) $\vee$ RCE(1) $\vee$ DRL(1)
30	EES	SEF(1)
31	EFR	DFR(1)
32	EIS	IDS(1)

The FER(1) level is asserted whenever the FER flip-flop is set; this occurs for all alarm and error conditions that arise at the discfile. The remaining interrupt conditions enabled by the EFE switch represent three additional error conditions that arise within the discfile control:

parity error, read compare error, and data request late.

The SEF flag is set 2 μsec after the file-control octoflop is advanced to SCE state. The SEF remains set until it is reset by an SCL pulse. The primary purpose of SEF is to save the SCE state for sampling. Even though SCE is no longer asserted, a 1 in SEF lets the programmer know that the end of the data sector has been passed.

The DFR(1) level is asserted for 1 msec after the discfile has found the addressed sector. The resulting priority interrupt request is often used to notify the programmer that the discfile control may need a command.

The IDS(1) level is asserted whenever the file-control octoflop is reset to IDS state. The resulting priority interrupt request often is used to notify the programmer that the discfile control has completed the preceding command and is ready to receive a new data address.

CHAPTER 4

SYSTEM OPERATING SEQUENCES

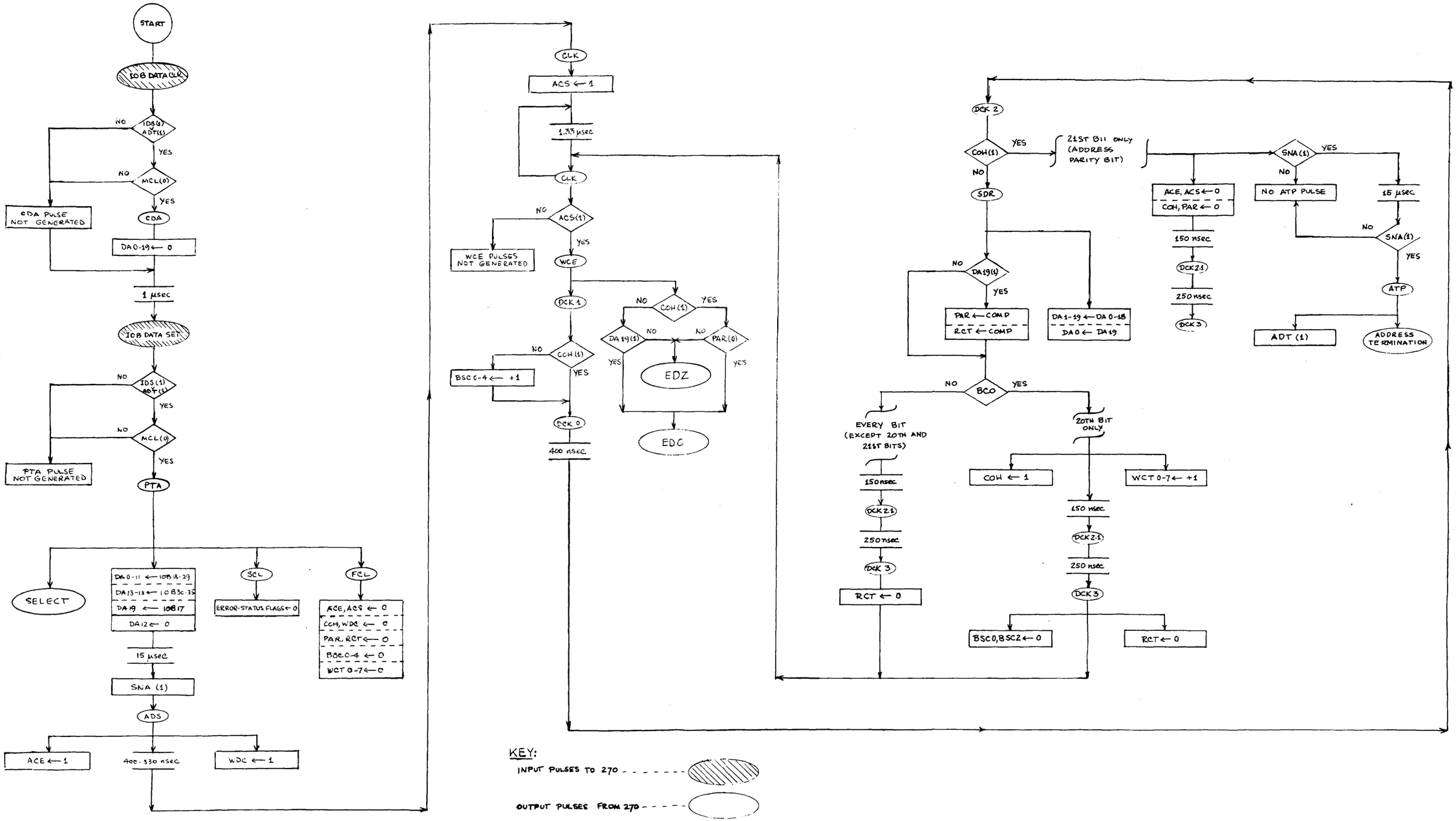
4.1 ADDRESS TRANSMISSION (Figure 4-1)

The address transmission sequence is the series of operations by which the Discfile Control 270 accepts a 19-bit address from the PDP-6 processor, and transmits a 21-bit address to the Discfile 5022. The address transmission sequence is initiated by a DATAO 270 instruction from the processor. The entire sequence occurs during the portion of the file-control octoflop cycle that precedes ADT state (refer to Figure 3-1).

a Load DA0-19

The IOB DATA CLR pulse of the DATAO 270 instruction starts the address transmission sequence. Provided that the file-control octoflop is in an appropriate state (IDS or ADT) and that the MCL pushbutton is not being operated, the IOB DATA CLR produces a CDA pulse, thus clearing the discfile control data accumulator in preparation for the receipt of a new address from the processor.

One microsecond later, the IOB DATA SET pulse arrives. The same conditions that enabled the generation of CDA also enable the generation of PTA. The PTA pulse sends a SELECT to the discfile, clears the discfile control by generating FCL and SCL, and loads a new address into bits DA0-19 of the data accumulator. Nineteen of these 20 address bits arrive from the processor on the IO bus. The remaining bit, DA12, is



KEY:

INPUT PULSES TO 270 - - - - - - -

OUTPUT PULSES FROM 270 - - - - -

270 INTERNAL CONTROL PULSES - - -

FIGURE 4-1 ADDRESS TRANSMISSION SEQUENCE

always 0, and thus can be generated within the discfile control.

b Clock Turn-on

The PTA pulse starts a 15-μsec delay. The expiration of this delay steps the octoflop from IDS (or ADT) state to SNA state, and produces the ADS pulse. The ADS pulse sets the ACE flip-flop and the WDC flip-flop. The ACE and ACS flip-flops make up a synchronizing network that controls the turn-on and turn-off of the address clock. The ADS pulse starts the turn-on sequence by setting the ACE flip-flop. The first CLK pulse to be generated after ACE is set sets ACS. The internal address clock (within the discfile control) produces one CLK pulse every 1.33 μsec, and each CLK that occurs while ACS(1) is asserted produces a WCE pulse.

The set of the WDC flip-flop by the ADS pulse governs the subsequent setting of the COH flip-flop; the COH(1) level enables the discfile control to detect the occurrence of the address parity bit and to distinguish it from the preceding 20 bits of the address.

c Address Transmission

All address bits are transmitted to the discfile over the even data twisted-pair transmission lines as EDO and EDZ pulses. The 21 WCE pulses that occur during the time that the ACS flip-flop remains set clock out the 21 bits of the address from the discfile control to the discfile. Each WCE pulse also starts the DCK clock chain, thus producing DCK 1, DCK 0, DCK 2, DCK 2.1, and DCK 3 in that order.

For the first 20 bits of the address, COH(1) is not asserted. Consequently, the contents of DA19 are transmitted to the discfile as the

current address bit and then the bit-shift counter BSC0-4 is incremented at DCK 1 time. For the final 21st bit of the address, COH(1) is asserted and the bit-shift counter is not incremented. The 21st address bit is the address parity bit that enables the discfile to check the address transmission. The source of the address parity bit is not DA19, but is rather the complement of the PAR flip-flop, the circuit that generates both address and data parity within the discfile control.

d Shift Address and Generate Parity

The DCK 1 pulse initiates DCK 0, and that pulse in turn produces DCK 2 after a 400-nsec delay. The COH(1) level is asserted during only that DCK 2 pulse that is initiated by the final 21st bit of the address (the parity bit). All other DCK 2 pulses that occur during the address transmission sequence (the 20 DCK 2 pulses corresponding to the 20 address bits that precede the parity bit) produce SDR pulses. The SDR pulse causes the contents of the data accumulator to be shifted right one bit position. The contents of DA19 are ring-shifted back into DA0. This ring-shift feature does not affect the actual address transmission to the discfile, but it does provide a useful maintenance aid for the discfile control. After the address transmission to the discfile is finished, the original address should be completely restored in DA0-19. There it can be sampled from the processor (by a DATAI 270 instruction) in order to test the operation of the data accumulator shift circuitry.

Besides shifting the contents of DA0-19 right, the SDR pulse may also complement the PAR flip-flop and the RCT flip-flop. The SDR

complements PAR and RCT whenever DA19 contains 1 before the shift. The complement of RCT is a don't-care condition during address transmission; its usefulness is confined to the read compare instruction. The FCL pulse ensures that the PAR flip-flop starts in the 0 state at the beginning of the address transmission. The PAR is then complemented at any SDR pulse that follows the transmission of a 1 address bit. In this way, PAR accumulates the parity of the address word. By transmitting PAR(0) as the 21st address bit, the discfile control ensures that the total address word transmitted to the discfile (including the parity bit) has odd parity.

e Bit-shift Counter Overflow

The return loop (designated "following every bit except the 20th and 21st") includes the remaining two pulses of the DCK clock chain, DCK 2.1 and DCK 3 and also includes the reset of RCT. Neither these two pulses nor the RCT reset have any logical relevance to the address transmission sequence; they are don't-care conditions included on the flow chart for maintenance purposes only.

The operations described above are identically repeated 19 times during the transmission of the first 19 bits of the address. Each CLK produces a WCE pulse; the WCE in turn transmits one address bit to the discfile and initiates the DCK clock chain.

At the DCK 1 pulse that is initiated by the 20th address bit, the bit-shift counter BSC0-4 is incremented to 20, and the overflow level BCO is asserted. The assertion of BCO causes the DCK 2 and DCK 3 pulses

that correspond to the 20th address bit to perform certain additional functions not performed during the preceding 19 bits of the address. These additional functions are all included in the right-hand ("yes") branch of the flow chart following the BCO decision box.

Most of these BCO "yes" functions can be considered to be don't-care conditions. (The clearing of WCT0-7 and of BSC0, 2 is unnecessary, because the WCT and BSC counters are cleared by FCL before the beginning of the first subsequent command sequence; the RCT clear is relevant only during read compare commands.)

The set of the COH flip-flop is, however, directly relevant to the address transmission sequence. The address termination functions initiated by the assertion of the COH(1) level are described below.

f Address Termination

The final 21st CLK of the address transmission sequence produces a final 21st WCE pulse. Because COH(1) is then asserted, that WCE pulse transmits the address parity bit taken from PAR(0) rather than the contents of DA19. The same WCE initiates the DCK clock chain for the last (21st) time during the address transmission. Because of the assertion of COH(1), the DCK 1 pulse does not increment BSC0-4.

At DCK 2, the assertion of COH(1) prevents the generation of SDR and instead causes the execution of the additional functions shown at the "yes" side of the COH(1) decision box (following DCK 2). The ACE and ACS flip-flops are reset, turning off the address clock. The clock turn-off prevents the generation of any further WCE pulses and so prevents any further triggering of the DCK clock chain.

The same DCK 2 pulse that resets ACE and ACS also starts a 15-μsec delay. Termination of this delay produces the ATP pulse. Note that the ATP pulse can be generated only if the file-control octoflop remains in SNA state until the completion of the 15-μsec

delay. (In the unlikely event that an IOB reset pulse were to be applied to the discfile control during the 15-μsec delay, it would be necessary to prevent the generation of ATP.) The ATP pulse advances the octoflop from SNA state to ADTstate, and sends an ADDRESS TERMINATION pulse to the discfile over the odd data one's line. This completes the address transmission sequence.

4.2 WRITE COMMAND (Figure 4-2)

The write command is the series of operations by which the Discfile Control 270 transmits one or more sectors of output data to the Discfile 5022. The entire write command occurs during the portion of the file-control octoflop cycle lying between the CSP pulse and the ESP 2 pulse (refer to Figure 3-1).

The Discfile Control 270 receives output data from the Data Control 136 in data characters of 18 bits. The discfile control then transmits the data to the Discfile 5022 in serial form, one bit at a time, over four twisted-pair transmission lines. These four twisted-pair lines are used in sets of two; one line of each set is pulsed for a 1, the other for a 0. Two such sets of two lines are used. This allows alternate bits (odd and even) to be transmitted on different lines, thus halving the data generation frequency.

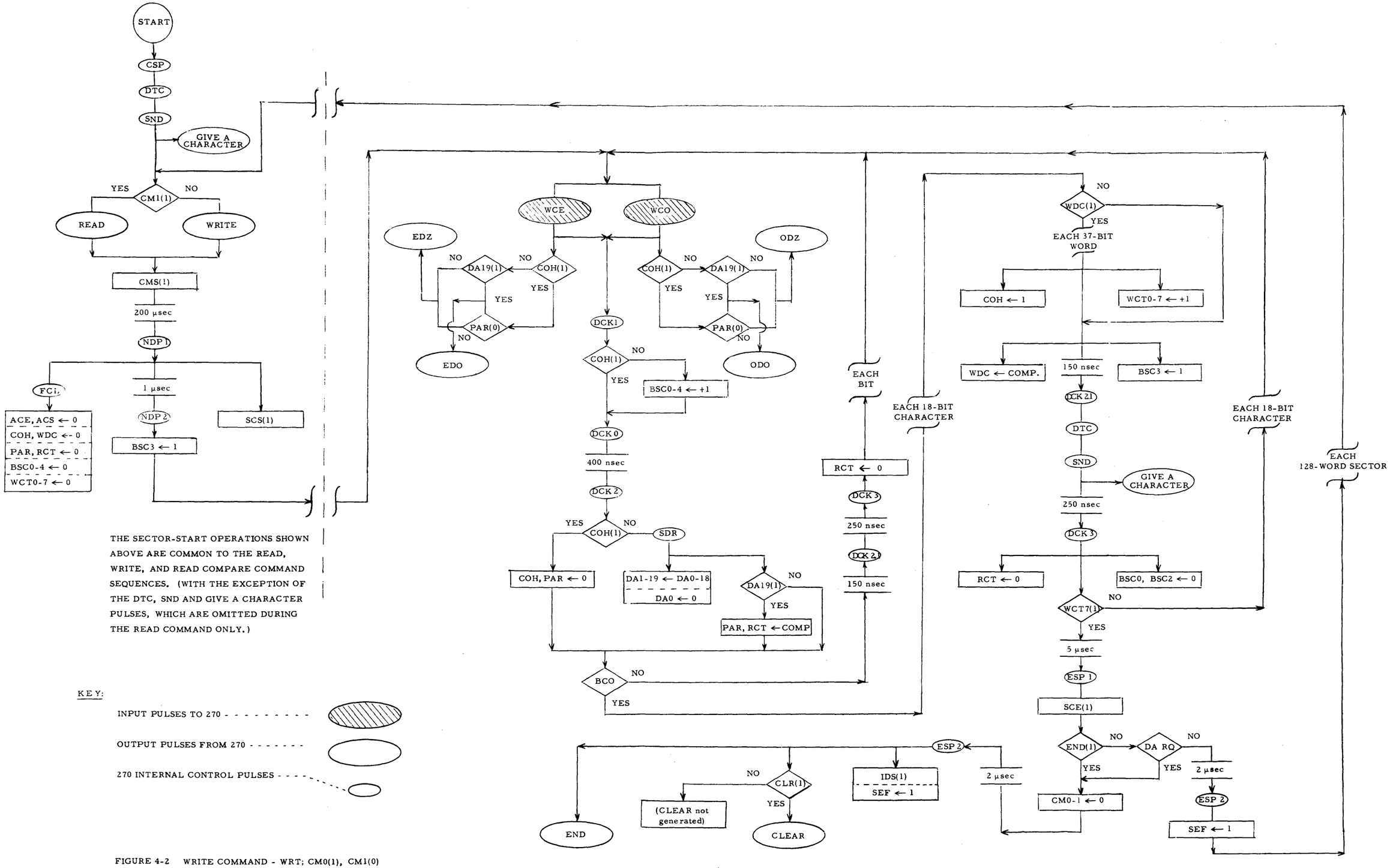
After transmitting each set of 36 data bits, the discfile control generates and transmits an accompanying parity bit to the discfile. The discfile writes the data in the form of 37-bit data words, each such word containing the 36 data bits and a parity bit of the correct value to yield odd parity for the entire 37-bit word.

a Sector-Start Operations

Most of the sector-start operations shown at the left of Figure 4-2 are common to all three commands, read, write, and read compare. (The only non-common operations are the DTC, SND and GIVE A CHARACTER pulses which are initiated by the CSP pulse during the write and read compare commands but not during the read command.) During the write command, the CSP pulse that advances the file-control octoflop from ALS state to CMS state produces a DTC pulse. That pulse in turn generates SND and GIVE A CHARACTER. The SND pulse causes the contents of the 18 high-order bits of the Data Control 136 data accumulator to be loaded into bits DA2-19 of the discfile control data accumulator. (This is the first character that will be transmitted bit-by-bit to the discfile for writing.) The GIVE A CHARACTER pulse signals the Data Control 136 that the discfile control has accepted the character currently stored in the high order end of the data control data accumulator (by clocking it in with an SND pulse), and that the data control should bring the following character into position to be transferred out.

The CSP pulse also sends either a READ or a WRITE pulse to the discfile, the choice depending upon the state of the CM1 command code bit. During the write command CM1 contains 0, so a WRITE pulse is sent to the discfile, commanding the discfile to write out the following sector of data.

Besides advancing the octoflop to CMS state, the CSP pulse also starts a 200-μsec delay. The expiration of this delay produces the NDP 1 pulse.



KEY:

INPUT PULSES TO 270 - - - - - - - - -

OUTPUT PULSES FROM 270 - - - - - - -

270 INTERNAL CONTROL PULSES - - - - - - - - -

FIGURE 4-2 WRITE COMMAND - WRT; CM0(1), CM1(0)

The NDP 1 pulse advances the octoflop to SCS state and produces the FCL pulse which clears the control flip-flops and registers of the discfile control. One microsecond after the NDP 1 pulse, the NDP 2 pulse sets bit BSC3 of the bit-shift counter. This ensures that the bit-shift counter starts the forthcoming data sector with its contents equal to 2. Since the counter overflows at a count of 20, this means that it overflows after counting 18 bits, the number of bits in a single data character. The sector start operations thus leave the discfile control in an appropriate state to begin writing the first word of the current sector of data.

b Operations That Recur at Each Bit That Is Written

The write command is synchronized to WCE and WCO pulses initiated by WRITE CLOCK EVEN and WRITE CLOCK ODD pulses generated at the discfile and transmitted to the discfile control over two twisted-pair transmission lines. The COH flip-flop contains 1 only during the final bit of each 37-bit data word (the parity bit). During the transmission to the discfile of the 36 data bits that precede the parity bit, COH(1) is not asserted. Consequently, these 36 data bits are taken from the contents of DA19. When DA19 contains 1, a 1 bit is transmitted to the discfile; the WCE pulse then sends the discfile an EDO pulse, or else the WCO pulse sends the discfile an ODO pulse. Similarly, when DA19 contains a 0, a 0 bit is transmitted to the discfile in the form of either an EDZ or an ODZ pulse.

The final bit of the 37-bit word transmitted to the discfile is the parity bit. This data-word parity bit is generated in much the same way

that the address parity bit is generated during the address transmission sequence described in the previous section. The COH flip-flop contains 1 when the parity bit is transmitted to the discfile. If PAR(0) is asserted, a 1 parity bit is transmitted as the 37th bit of the data word; if PAR(0) is not asserted, a 0 parity bit is transmitted.

Besides clocking out a data or parity bit to the discfile, every WCE pulse and every WCO pulse also triggers the DCK clock chain. As a result, there is a full set of DCK clock pulses corresponding to each bit transmitted to the discfile during the write command. The bit-shift counter BSC0-4 is incremented at every DCK 1 pulse except those corresponding to parity bits (when COH(1) is asserted).

Every DCK 2 pulse except those corresponding to parity bits initiates an SDR pulse. The SDR pulse shifts the data accumulator right one bit-position, and shifts a 0 into DA0. If DA19 contained 1 prior to the shift (i.e. if a 1 data bit has just been transmitted to the discfile), then the SDR pulse also complements the PAR flip-flop and the RCT flip-flop. The PAR complement accumulates parity for the current data word. During write commands the RCT complement is a don't-care condition; it is relevant only during the read compare command. The same is true of the RCT clear at DCK 3 time. The DCK 2 pulse corresponding to the final bit of the 37-bit word (the parity bit) clears both COH and PAR in preparation for the next word to be written.

c End of Character and End of Word

At the DCK 1 pulse corresponding to the 18th bit of each data character, the bit-shift counter is incremented to its overflow value of 20, and the BCO level is asserted. The assertion of BCO causes the DCK 2, DCK 2.1 and DCK 3 pulses that correspond to the 18th bit of the data character to perform certain additional functions not performed during the preceding 17 bits. These additional functions are all included in the "yes" branch of the flow chart following the BCO decision box (the right-hand portion of Figure 4-2).

The WDC flip-flop was cleared at the FCL pulse, so WDC(1) is not asserted during the first data character of the sector. The first character thus bypasses the COH set and the WCT0-7 increment operations. (This bypass is necessary because COH is set and the word counter incremented only at the end of a 37-bit data word, and not at the end of the first of the two 18-bit data characters included in the data word.)

When BCO is asserted, the same DCK 2 pulse that produces SDR also complements WDC and sets BSC3. The WDC flip-flop acts as a count-of-two counter. It is incremented (complemented) at the DCK 2 pulse corresponding to the final data bit of every 18-bit data character. The WDC contains 1 only during the second data character of the data word. Consequently, it can provide the necessary screening condition for allowing COH to be set and WCT0-7 to be incremented. (These two operations are executed at the end of the second data character of each data word, but not at the end of the first.)

The BSC3 set adds two to the contents of the bit-shift counter. This is done in order to enable the bit-shift counter to start the following count in the proper state. By adding two to the overflow count already in the counter, DCK 2 momentarily raises the contents of the counter to 22; however, the DCK 3 pulse that occurs 400 nsec later clears both BSC0 and BSC2, and thus subtracts 20 from the contents of the counter, leaving it in the 2 state for the beginning of the next count sequence. This ensures that the counter will again overflow at the 18th bit of the subsequent data character (when its contents once more reach the overflow value of 20).

The DCK 2 pulse is followed 150 nsec later by DCK 2.1. The DCK 2.1 pulse produces the DTC pulse. That pulse in turn generates SND and GIVE A CHARACTER. The SND pulse causes the contents of the 18 high order bits of the Data Control 136 data accumulator to be loaded into bits DA2-19 of the discfile control data accumulator. (This is the next character that will be transmitted bit-by-bit to the discfile.) The GIVE A CHARACTER pulse signals the Data Control 136 that the discfile control has accepted the character currently stored in the high order end of the data control data accumulator. The DCK 3 pulse also clears RCT. This is a don't care condition during the write command; it is relevant only during read compare.

The BCO level is again asserted at the 36th bit of each data word (the 18th bit of the second data character of the word). The same operations occur that are described above, and in addition to those operations, two more:

the COH flip-flop is set, and the word counter WCT0-7 is incremented. The set of COH enables the discfile control to distinguish the following bit (the 37th bit of the data word) as the parity bit. The incrementing of the word counter reflects the fact that one full data word (with the exception of the parity bit) has been transmitted to the discfile for writing.

The set of the COH flip-flop causes COH(1) to be asserted during transmission of the final 37th bit of the data word. Three results follow: 1) The output bit transmitted to the discfile is taken from PAR(0) rather than from DA19(1); 2) The bit-shift counter is not incremented; and 3) Instead of SDR being generated at DCK 2, the COH and PAR flip-flops are cleared at that time.

The entire sequence described above is repeated for each of the 128 data words making up the data sector being written. This is true even though the output data from the Data Control 136 may be exhausted before the full 128 words are written. In that event, the discfile control transmits 0s in lieu of data bits (and 1s for parity bits) for the remainder of the sector. At the last data bit of the 128th word of the sector, the word counter is incremented to its final value of 128. The WCT7(1) level is then asserted and the discfile control breaks out of the loop and executes the end-of-sector operations described below.

d End of Sector

As soon as WCT7(1) is asserted, indicating that the word counter has reached its final value of 128, the DCK clock chain is turned off. (Because the WCE and WCO pulses are not inhibited, the parity bit of the final

data word can still be transmitted to the discfile even after the DCK clock chain is turned off.)

Five microseconds after the final DCK 1 pulse of the sector, an integrating delay times out, producing the ESP 1 pulse. The ESP 1 advances the file-control octoflop from SCS state to SCE state. If the END flip-flop is not set, and if furthermore DA RQ is not asserted (indicating that the Data Control 136 still has more output data to send) the command continues for one or more additional sectors. Two microseconds after the ESP 1 pulse an ESP 2 pulse is generated. The ESP 2 pulse sets the sector end flag SEF to indicate to the programmer that the end of the first sector has been reached. The SEF remains set until being cleared by an SCL pulse. The ESP 2 pulse also returns the file-control octoflop from SCE state to CMS state and simultaneously sends a second WRITE pulse to the discfile, thereby initiating the sector-start operations for the second sector of data that is to be written.

The WRITE pulse automatically causes the discfile to increment its stored address by 1. The next sector is therefore at the next consecutive address following that at which the previous sector was written. This process can be continued for up to 44 consecutive sectors of data without any new DATAO 270 command being given. Note, however, that the programmer must keep count of the number of sectors that have been written since the last change of disc or position address. If more than 44 sectors are written without a new DATAO 270, data will be written over and thus destroyed.

At the end of the first sector during which the END flip-flop is set or the DA RQ level is asserted, the discfile control terminates the write command in the manner described below.

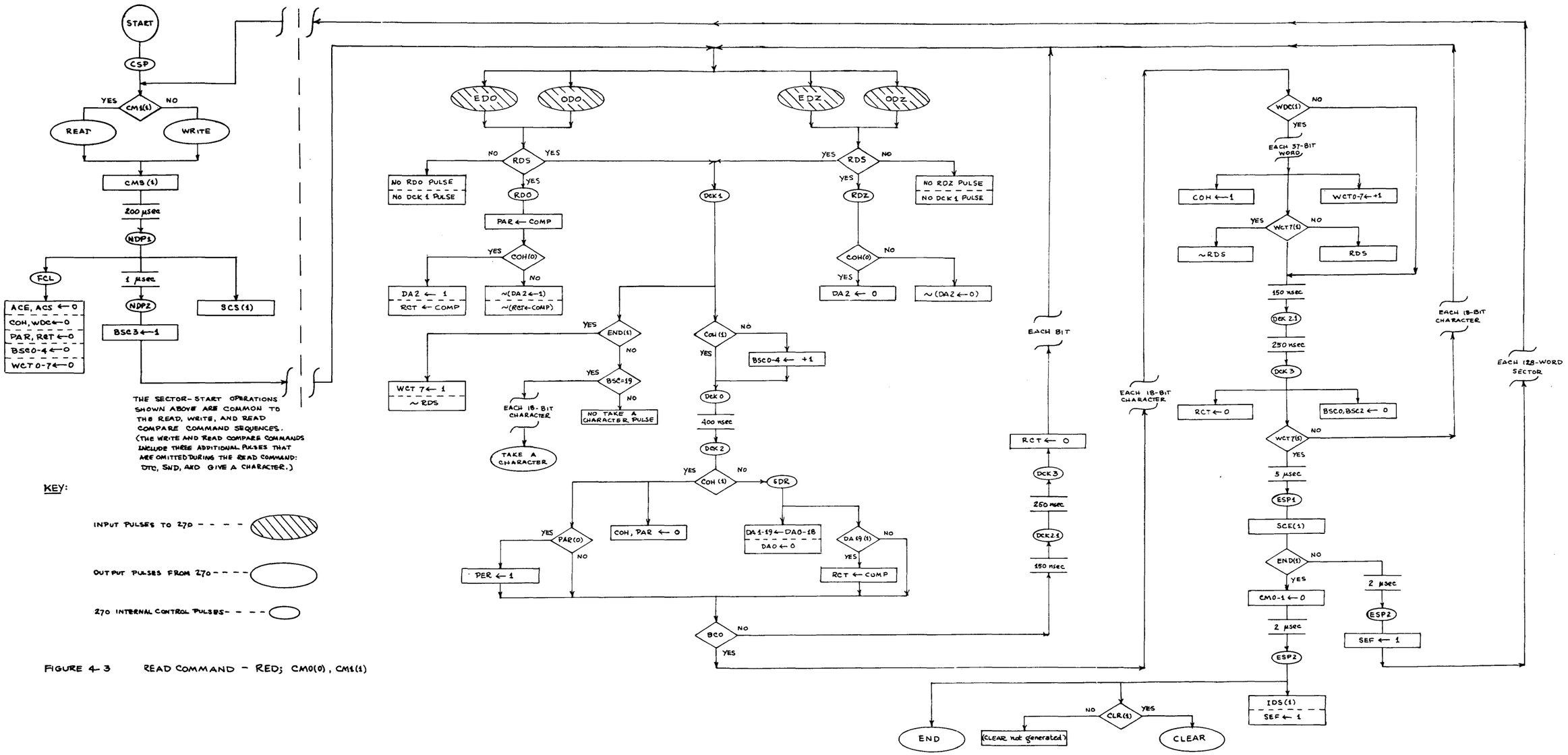
e Terminate Write Command

If, when the discfile control reaches the end of a sector, END is set, or DA RQ is asserted, then the ESP 1 pulse clears the two command code bits CM0-1. Two microseconds later the ESP 2 pulse resets the file-control octoflop from SCE state to IDS state and sends an END pulse to the discfile. If the CLEAR flip-flop is set, the ESP 2 pulse also sends a CLEAR pulse to the discfile.

4.3 READ COMMAND (Figure 4-3)

The read command is the series of operations by which input data is read in from the discfile. The entire read command occurs during the portion of the file-control octoflop cycle lying between the CSP pulse and the ESP 2 pulse (refer to Figure 3-1).

The Discfile 5022 starts to read data at the beginning of the 128-word sector addressed by the preceding DATAO 270 instruction. The data is loaded into DA2 as it is read in serial fashion, one bit at a time, and is then shifted right until a full 18-bit data character is stored in DA2-19. At the conclusion of this loading process, the discfile control sends a TAKE A CHARACTER pulse to the Data Control 136, causing the data control to strobe the character into its low order accumulator bits, and if necessary, to shift it left to make room for the second character of the data word. A



KEY:

INPUT PULSES TO 270 - - - -

OUTPUT PULSES FROM 270 - - - -

270 INTERNAL CONTROL PULSES - - - -

FIGURE 4-3 READ COMMAND - RED; CM0(0), CM1(1)

similar process is followed for the second 18-bit character and all succeeding data characters. Every 36 data bits from the discfile are followed by a parity bit, and the parity of each data word is checked at the discfile control.

a Sector-Start Operations

The sector-start operations shown at the left of Figure 4-3 are common to all three commands, read, write and read compare. The read command sector start operations differ from those of the write and read compare commands in only one respect: during the read command the CSP pulse does not produce DTC, SND and GIVE A CHARACTER (as it does during write and read compare). These three pulses are omitted during the read command because there is no need during that command to send output data from the Data Control 136 to the discfile control. Refer to paragraph 4-2a above for a description of the write command sector start operations.

b Operations That Recur at Each Bit That Is Read

The read command is synchronized to the read-data pulses from the discfile. These pulses are transmitted to the discfile control over four twisted-pair transmission lines. The EDO and ODO pulses bring in the 1 bits, and the EDZ and ODZ pulses bring in the 0 bits. So long as the read command continues, each read data pulse to arrive at the discfile control triggers the DCK clock chain. As a result, there is a full set of DCK clock pulses corresponding to each bit transmitted to the discfile control during the read command.

In flow chart Figure 4-3, the operations that recur at each bit that is read are shown below the four read data input pulses from the discfile: EDO, ODO, EDZ, and ODZ. Note that there are three major branches of the flow chart below these input pulses. The branch directly below the EDO and ODO pulses represents the results that are produced directly by the arrival of a 1 bit from the discfile; the branch directly below the EDZ and ODZ pulses shows the direct results of the arrival of a 0 bit from the discfile. The center branch reflects the triggering of the DCK clock chain and includes all the effects of the five DCK clock pulses, DCK 1, DCK 0, DCK 2, DCK 2.1 and DCK 3.

At each EDO or ODO pulse from the discfile that arrives while RDS is asserted, the DCK clock chain is triggered, and an RDO pulse is generated, indicating that a 1 bit (either a data bit or a parity bit) has been read in from the discfile. If RDS is not asserted, the triggering of the DCK clock chain is prevented, and so is the generation of RDO.

The read command is unlike the other two commands, write and read compare, in that it may be terminated at any point within a sector by merely setting the END flip-flop. (This can be done by giving a CONO 270 instruction with a 1 in bit 23.) The first DCK 1 pulse to occur after END is set sets WCT7, and thereby ends the assertion of RDS. The negation of RDS turns off the DCK clock chain as described above, and also prevents the generation of both the RDO and the RDZ pulses.

The left branch of the Figure 4-3 flow chart (immediately below the EDO and ODO inputs from the discfile) shows the generation and results

of the RDO pulses. Every RDO pulse complements the parity accumulator flip-flop, PAR, thus accumulating the parity of the incoming data word (by counting the number of 1 bits read into the discfile control). If the parity of the incoming word is correct (odd), the PAR flip-flop ends the word (after the 37th bit, the parity bit, is read in) in the 1 state. If PAR is not in the 1 state at the end of the incoming data word, a parity error is indicated.

The COH flip-flop contains 1 only during the final bit of each 37-bit data word (the parity bit). During the transmission to the discfile control of the 36 data bits that precede the parity bit, COH(1) is not asserted. Provided that the COH flip-flop contains 0, the RDO pulse performs two additional functions: it sets DA2 and complements RCT. The set of DA2 reads the incoming 1 bit into the left end of the discfile control data accumulator. (The complement of RCT is a don't-care condition during the read command; it is relevant only during the read compare command.) The right branch of the Figure 4-3 flow chart (immediately below the EDZ and ODZ inputs from the discfile) shows the generation and results of the RDZ pulses. If the COH flip-flop contains 0, the RDZ pulse loads a 0 into DA2, otherwise RDZ has no effect.

Whenever RDS is asserted, each EDO, ODO, EDZ and ODZ pulse triggers the DCK clock chain. The DCK 1 clock pulse is the first pulse of the DCK clock chain (center branch of Figure 4-3 flow diagram). The effects of the DCK 1 clock depend on the state of the END flip-flop and the COH flip-flop. The END flip-flop is set to 1 only when the read

command is to be terminated immediately. The first subsequent DCK 1 pulse then sets WCT7. The set of WCT7 ends the assertion of RDS, thereby preventing the generation of any further RDO or RDZ pulses and also preventing any further triggering of the DCK clock chain. (The final sequence of DCK clock pulses, that sequence beginning with the DCK 1 pulse that sets WCT7, is of course completed without interruption; a final DCK 0, DCK 2, DCK 2.1, and DCK 3 follow the last DCK 1 pulse.)

If the END flip-flop is not in the 1 state, the discfile control is periodically required to generate a TAKE A CHARACTER pulse at DCK 1 time. This pulse is generated at that DCK 1 pulse that corresponds to the final bit of each 18-bit data character read from the discfile. (Note that the BSC = 19 condition is used to enable the generation of this pulse rather than the normal overflow count of 20. This is done because the TAKE A CHARACTER pulse is generated by the same DCK 1 clock pulse that increments the bit-shift counter to the overflow count of 20.) The TAKE A CHARACTER pulse signals the data control that the discfile control has presented a 19-bit input character to the low order strobe data inputs of the data control data accumulator. This indicates to the data control that it may strobe the character in, and that if required if should shift the character left to make room for the next character.

The bit-shift counter BSC0-4 is incremented at every DCK 1 pulse except those corresponding to parity bits (when COH(1) is asserted). Every DCK 2 pulse except those corresponding to parity bits initiates an SDR pulse. The SDR pulse shifts the discfile control data accumulator

right one bit-position, and shifts a 0 into DA0. If DA19 contained 1 just prior to the shift, the SDR pulse complements RCT. (The RCT complement is a don't-care condition during the read command; it is relevant only during read compare. The same is true of the RCT clear at DCK 3 time.) The DCK 2 pulse corresponding to the final bit of the 37-bit word (the parity bit) tests the parity of the word and sets the parity error flip-flop PER if the parity accumulator flip-flop PAR is left in the 0 state (indicating that the parity is incorrect). The same DCK 2 also clears both COH and PAR in preparation for the next word to be read.

c End of Character and End of Word

At the DCK 1 pulse corresponding to the 18th bit of each data character, the bit shift counter is incremented to its overflow value of 20, and the BCO level is asserted. The assertion of BCO causes the DCK 2, DCK 2.1, and DCK 3 pulses that correspond to the 18th bit of the data character to perform certain additional functions not performed during the preceding 17 bits. These additional functions are all included in the "yes" branch of the flow chart following the BCO decision box (the right-hand portion of Figure 4-3).

Because these read command end-of-character and end-of-word functions are quite similar to the corresponding functions that are performed during the same phases of the write command, the present paragraph does not repeat the full description of the common operations, but instead describes only the few operations of the read command that differ from the write command. A comparison of the right hand portions of

Figures 4-2 and 4-3 illustrates the basic similarity of the two commands and also reveals the few respects in which they differ.

The read command flow chart includes an extra step at the DCK 2 pulse corresponding to the 36th data bit of each word. If after that DCK 2 pulse the WCT7 flip-flop contains 1 (indicating that the WCT counter has been incremented to its final value of 128), then the RDS level ceases to be asserted. (This decision is not included in the write command, because the RDS level is never asserted during write.)

Following the DCK 2.1 pulse, the read command omits the DTC pulse and the resulting SND and GIVE A CHARACTER pulses that are generated during the write command. These pulses are omitted during read because the direction of data flow is opposite to that during write. (The read command produces TAKE A CHARACTER pulses at DCK 1 time rather than GIVE A CHARACTER pulses at DCK 2.1.)

d End of Sector

The read command end-of-sector operations are almost identical to the corresponding operations of the write command. There is only one difference. The read command does not include the DA RQ decision box that causes the write command to terminate at the end of the first sector during which output data is exhausted. The read command always continues until the END flip-flop is set. If END is not set, the read command reads the contents of the 44 sequentially addressed data sectors at the discfile's presently addressed disc and position, and continues to reread the same 44 sectors of data until a new address is given. At the end of

each sector another READ pulse is sent to the discfile, and the discfile address is incremented by 1. (The discfile logic contains a count-of-44 address counter, which cycles through the 44 sector addresses in rotation.)

4.4 READ COMPARE COMMAND (Figure 4-4)

The read compare command is the series of operations by which the discfile control compares output data from the processor with corresponding input data read from the discfile. The entire read compare command occurs during the portion of the file-control octoflop cycle lying between the CSP pulse and the ESP 2 pulse (refer to Figure 3-1).

The comparison is executed on a bit-by-bit basis, but does not include parity bits. The output data is sent to the discfile control through the Data Control 136 (as during the write command described above), and the input data is brought in from the discfile (as during the read command described above). The discfile starts to read data at the beginning of the 128-word sector addressed by the preceding DATAO 270 instruction.

The RCE flip-flop in the discfile control is set whenever an output data bit from the Data Control 136 fails to match the corresponding input data bit read from the discfile. No output parity bits are generated during read compare commands, and for read comparison purposes the incoming parity bits from the discfile are ignored. Note, however, that the incoming data from the discfile is subject to the normal parity test during the read compare command just as during the read command.

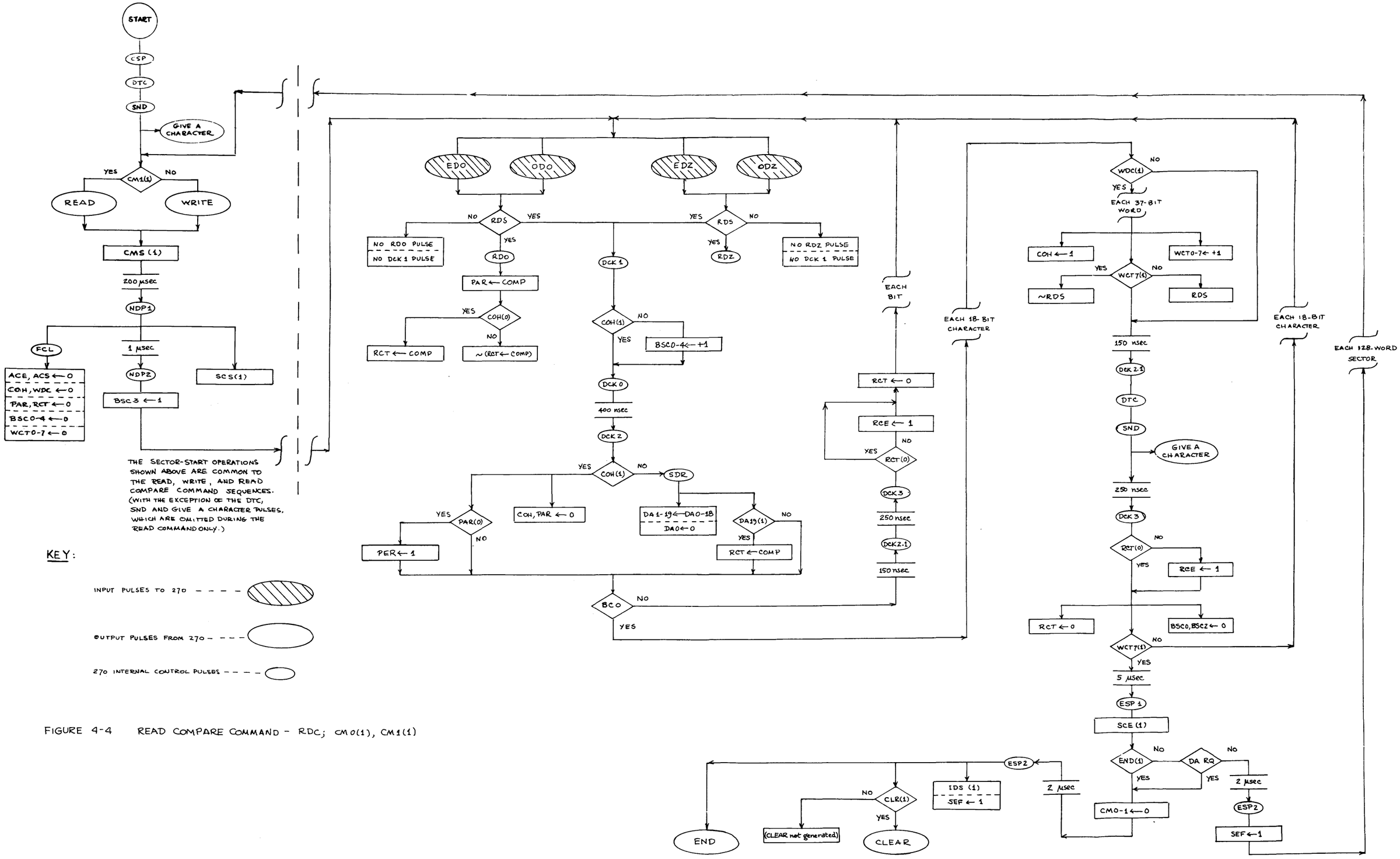


FIGURE 4-4 READ COMPARE COMMAND - RDC; CM0(1), CM1(1)

a Sector Start Operations

The sector-start operations shown at the left of Figure 4-4 are common to the write and read compare commands. For a description of these operations refer to paragraph 4-2a above.

b Operations That Occur at Each Bit That is Compared

The read compare command is very similar to the read command, and in most operations where it differs from the read command it is similar to write. Consequently, the present paragraph does not give a full description of the common operations, but instead describes only the few features of the read compare command that distinguish it from the read command described above. A comparison of Figure 4-4 with Figures 4-3 and 4-2 illustrates the basic similarities of the read compare command to the read and write commands, and also reveals the differences between the read compare and the other two commands. Like the read command, the read compare command is synchronized to the read-data pulses from the discfile: EDO, ODO, EDZ and ODZ. Although these incoming pulses are allowed to produce RDO and RDZ pulses whenever RDS is asserted, the RDO and RDZ pulses are not allowed to load data into the data accumulator as they do during the read command.

The RDZ pulse is not used during read compare. The RDO pulse complements the parity accumulator PAR as during the read command. Provided that the COH flip-flop is in the 0 state (indicating that the current bit is a data bit, not a parity bit, and hence is subject to read comparison) each RDO pulse also complements the RCT flip-flop. This is the first of

the two RCT complement operations that make up the comparison. The second RCT complement operation is initiated at DCK 2 time by the SDR pulse provided that DA19 contains a 1.

The two sets of complement operations work together to effect the bit-by-bit read comparison. If both the input and the output data are 0 bits, RCT will not be complemented at all, but will remain in its initial 0 state. If both bits are 1, the end result is the same; although RCT is complemented twice, it is still left in the 0 state after the DCK 2 pulse. However, if the input and output data bits are different, RCT is complemented only once, and is therefore left in the 1 state after DCK 2. The RCE flip-flop is then set by the DCK 3 pulse, thus saving the read compare error indication. The DCK 3 pulse clears RCT after each data bit comparison to prepare the discfile control for the next bit comparison. Once the RCE flip-flop is set, however, it remains set until cleared by an SCL pulse.

Like the write command, the read compare command always continues to the end of a 128-word data sector. It cannot be terminated in mid-sector by setting the END flip-flop as can the read command. Consequently the read compare flow chart (Figure 4-4) omits the DCK 1 set of WCT7 (enabled by END) that is included in the read command flow chart (Figure 4-3). The read compare command also omits the generation of TAKE A CHARACTER at DCK 1 time. The TAKE A CHARACTER pulse is omitted because during read compare no input data is sent to the Data Control 136.

With the exceptions described above, the remaining operations that recur each bit during the read compare command are substantially identical to those that occur during the read command.

c End of Character and End of Word

These operations are almost identical to the corresponding operations that occur during the write command. However, the read compare command, like the read command, includes an extra step at the DCK 2 pulse corresponding to the 36th data bit of each word. If after that DCK 2 pulse the WCT7 flip-flop contains 1 (indicating that the WCT counter has been incremented to its final value of 128), then the RDS level ceases to be asserted. (This decision is not included in the write command, because the RDS level is never asserted during write.) The read compare command also includes an extra step at every DCK 3 pulse (including those DCK 3 pulses that occur at the end of each data character and each data word). This extra step is the RCE set that occurs whenever RCT contains 1 at DCK 3 time.

d End of Sector

The read compare command end-of-sector operations are identical to the corresponding operations of the write command; refer to paragraph 4-2 above.

CHAPTER 5

MAINTENANCE

5.1 GENERAL

The Discfile Control 270 and the Discfile 5022 are ordinarily used as peripheral systems of the PDP-6 installation. The maintenance chapters of the PDP-6 Circuits manual and the PDP-6 Processor and Memory manuals include maintenance procedures and suggestions of general relevance not repeated in this manual. Familiarity with this material is essential for the efficient maintenance of the discfile control, the discfile, and all other DEC systems used with the PDP-6. The topics covered include: Tools and Test Equipment, Removal and Replacement of Modules, Module Troubleshooting, Use of Marginal Check, System Troubleshooting, and Maintenance Logs.

CAUTION

> The procedures described in the maintenance chapters of the PDP-6 Circuits manual and the PDP-6 Processor and Memory manuals should be thoroughly understood before undertaking troubleshooting and repair of the Discfile Control 270 and the Discfile 5022.

5.2 USE OF DRAWINGS

The complete system logic of the Discfile Control 270 is shown in the seven engineering logic drawings of Chapter 9. Because these engineering logic drawings are the most frequently used source of

troubleshooting information, it is important to be familiar with the conventions and symbols which they employ.

Figure 8-1 shows the standard cabling and module location numbering used within the DEC FLIP CHIP logic racks.

5.3 EQUIPMENT LAYOUT

The discfile control logic is contained in three DEC FLIP CHIP mounting panels designated from top to bottom A-B, C-D, and E-F. Within the A-B rack, the upper row of FLIP CHIP modules is designated A, and the lower row B; similarly row C is the upper row of the C-D rack, and row E is the upper row of the E-F rack. Each row has 32 module positions numbered from left to right. Thus, for example, module B7 is the seventh module from the left end of the lower row in the top FLIP CHIP logic rack. The module location and use are shown in the two module location diagrams in Chapter 9, UML-D270-0-11 (Sheets 1 and 2). Circuit schematics of all 27 FLIP CHIP module types used in the discfile control logic are presented in order of module type number in Chapter 9.

5.4 PREVENTIVE MAINTENANCE OF DISCFILE CONTROL 270

The cooling fan in the bottom of the bay should be checked daily for proper operation and free flow of air. Under normal operating conditions, the air filter at the bottom of the bay should be changed and cleaned monthly. The correct procedure for changing and cleaning filters is described in the PDP-6 processor manual maintenance chapter.

Preventive maintenance procedures for the system logic should also be scheduled on a regular basis. The same considerations that call for preventive maintenance of the PDP-6 processor apply equally to the peripheral systems. Appropriate maintenance procedures will detect and eliminate most potential malfunctions before they can cause operating errors. Refer to PDP-6 processor manual for suggested procedures and schedules. The same basic techniques that are used in maintaining the processor (trouble isolation methods, maintenance programs, marginal check, etc.) are equally useful for maintaining the discfile control and interface units.

5.5 PREVENTIVE MAINTENANCE OF DISCFILE 5022

Routine scheduled preventive maintenance procedures normally require about 8 man-hours per month. This maintenance should be performed by suitable trained and competent customer personnel. The schedule for routine preventive maintenance should be in accordance with the preventive maintenance procedures recommended by Data Products Corporation in the Discfile Instruction Manual and Customer Information Bulletins; refer to paragraph 1.5 above, REFERENCE DOCUMENTS.

If parts and maintenance personnel are available, unscheduled maintenance normally will not exceed an average repair time of one hour per month per 5022 discfile. Spare parts should be functionally interchangeable with like assemblies, sub-assemblies, and replaceable parts as found within the system. They should also be physically interchangeable, one with another. A list of spare parts is supplied to the customer.

5.6 ADJUSTMENT AND CALIBRATION

The Discfile Control 270 contains seven adjustable delays and an RC-coupled variable clock. Normally no adjustment or calibration of these circuits is required. They are set to the proper time constants and frequency before the system leaves the factory and there is no appreciable drift. It is desirable to check the delay time constants and the clock frequency once after the system is installed. Thereafter adjustment and calibration should be undertaken only if necessary. These procedures should not be made a part of routine maintenance schedules.

Calibration of all seven delays is done with a screwdriver adjustment and an oscilloscope. The R302 one-shot delay has two delays per module. The adjustment screws are both located at the edge of the card; one at the top, the other at the bottom. The R303 delay module includes only one delay; the adjustment screw is also at the edge of the card. The R401 clock module is calibrated to the correct pulse repetition frequency by an adjustment screw at the edge of the card.

TABLE 5-1 DISCFILE CONTROL DELAY TIME CONSTANTS

Delay Controls	Remarks	Module and Output Terminal	Module Type	Time Constant
ADS	PTA to ADS	B09-M	R302	15 μsec
ATP	Precedes ATP	B09-V	R302	15 μsec
CSP	ALP to CSP	B07-V	R302	15 μsec
DFR	DFR duration	B07-M	R302	1.0 msec
ESP 1	Integrating delay senses absence of DCK 1 clock	AB01-BS	R303	5.0 μsec
ESP 2	ESP 1 to ESP 2	B06-M	R302	2.0 μsec
NDP 1	Precedes NDP 1	B06-V	R302	0.2 msec

TABLE 5-2 DISCFILE CONTROL ADDRESS CLOCK FREQUENCY AND PERIOD

Output Pulses	Frequency pps	Period
CLK	750 kc	1.33 μsec

5.7 PROGRAM-CONTROLLED PULSE GENERATION

Three bits of the CONO 270 instruction are reserved for the purpose of generating certain pulses within the discfile control. These program-controlled pulses are useful for maintenance purposes.

TABLE 5-3 PROGRAM-CONTROLLED PULSES

When a 1 is Coded in CONO Bit:	The following Discfile Control Pulse(s) are Initiated by the PTC pulse	Function of Pulse
19	DTC	Generates GIVE A CHARACTER.
	SND	Causes the contents of the 18 high order bits of the Data Control 136 data accumulator to be loaded into bits DA2-19 of the discfile control data accumulator.
	GIVE A CHARACTER	Causes the Data Control 136 to advance a new character into position for transfer out to the discfile control.
20	TAKE A CHARACTER	Causes the Data Control 136 to strobe the 18-bit character in DA2-19 into the low order bits of its data accumulator and (if necessary) to shift the character left 18 bit-positions.
28	SCL	The SCL pulse clears the nine error-status flags ADE, CME, DCE, DRL, FER, PER, RCE, SEF, and WLE.

5.8 MAINTENANCE PROGRAMS

The MAINDEC 670 program MAGNETIC DISC TEST provides a comprehensive set of test procedures for detecting and isolating malfunctions in the discfile and the discfile control. The test procedures include individual component tests as well as complete system tests.

The MAINDEC 670 program consists of a compiler and disc exerciser routines. Operation is under control of DDT. The program contains numerous routines to manipulate data. It is stored on a single 16K DECtape File. The file contains a 5633-word data buffer, DDT, the test program, and selected portions of the symbol table. A more detailed description of the MAINDEC 670 program is distributed with the program.

5.9 RECOMMENDED SPARE PARTS

The most economical quantity of spare parts to be maintained depends on the requirements of the individual user, but the following listing is suitable for most installations.

a Module Spares

Suggested quantity of spares: one module of each type.

Module	Quantity in Discfile Control 270
B155	1
B171	1
G980	5
R001	4
R002	1

Module	Quantity in Discfile Control 270
R107	7
R111	20
R151	1
R181	2
R201	3
R202	9
R203	6
R205	11
R284	2
R302	3
R303	1
R401	1
R601	2
R602	6
R603	8
W005	2
W100	5
W101	10
W300	1
W607	1
W640	1
W700	1

b Component Spares

Component	Total Quantity in all Modules	Suggested Quantity of Spares
DEC2894-1	89	16
DEC2894-3	17	10
1N645	2116	48

Component	Total Quantity in all Modules	Suggested Quantity of Spares
1N3605	190	20
1N3606	2583	54
2N3009	5	6
2N3639	376	24
16J1	40	12

c Mechanical Spares

Part Number and Description	Suggested Quantity of Spares
53E168, Type CFG: Rotron fan with #2R blade	1
X-1431, 10" x 10" x 2" EZ Kleen Filter	1
Type 418 Super Filter Coat, pints	1

CHAPTER 6

INSTALLATION AND PREOPERATIONAL CHECKOUT

6.1 SITE SELECTION

Before installing the discfile control and discfile, select a suitable location near the PDP-6 arithmetic processor. The discfile control should be located within 25 feet of the processor, and the discfile should be located within 25 feet of the discfile control.

The discfile control is located in a cabinet 69-1/2 inches high, 22-1/4 inches wide, and 27-1/8 inches deep. A 3-foot clearance should be allowed on all sides of the equipment for access during maintenance. A level floor is required because the equipment frames are mounted on casters. The floor should be capable of supporting 150 psf. The system is designed to operate efficiently from 50° to 100°F. The plug-in modules are cooled by blowing air out the front of the bay. No additional cooling equipment is required.

The discfile control runs on ordinary 115-volt, 600 cycle current. A 10-ampere line is sufficient. The discfile control power cable is equipped with a Hubble Twist-Loc 3-prong, 30-ampere, 250-volt plug. Although the discfile control draws only about 3 amperes in normal operation, turn-on surges may reach almost 5 amperes, and system power capacity should be planned accordingly.

6.2 UNPACKING

The discfile control cabinet is shipped on a skid and may be crated or not, depending on the mode of transportation. For truck shipment, it may be left uncrated. A crate is furnished for air shipment. The crate is approximately 74 inches high, 27 inches wide, and 32 inches deep. The skid upon which it rests is about 6 inches high and 3 feet square. Interconnecting cables are specially made up for each installation and are ordinarily shipped with the equipment.

1. If the discfile control is crated, carefully remove all crating and strapping, and any packing material. If the unit is shipped uncrated, remove the skid and any protective padding.

2. The plenum doors at the rear of the bay have spring catches. To reinforce these doors during shipment, two bolts are used to hold each door shut. Remove these bolts and store them in the plastic loops provided.

3. Remove any packing material, shipping blocks, etc. from the inside of the discfile control cabinet.

4. The plug-in modules are taped into the mounting panels to prevent damage in shipment. Remove the tape.

> NOTE: If the user plans to reship the equipment (or to move it more than a short distance) in the near future, special containers and packing materials should be saved for reuse. These items are designed especially to accommodate the equipment and are the safest means of packing it for reshipment.

6.3 INSPECTION

The discfile control is thoroughly tested and checked before it leaves the factory. However, the discfile control should be inspected and checked again when installed to make sure that no damage has occurred during shipment. After the equipment is unpacked, check the following:

1. Have all the shipping blocks, packing materials, tape, etc. been removed? If not, remove them.
2. Are all plug-in units inserted firmly in position? Secure any that are loose.
3. Are there any loose nuts or bolts? If so, tighten them.
4. Are there any loose or broken wires? If so, repair them.

6.4 CABLE CONNECTIONS

Complete the inspection procedure as given above before connecting the cables.

1. Route the four cables from the PDP-6 processor into the cabinet, and connect the two W028 cable connectors at the end of each processor cable to the corresponding receptacles at the left of mounting panel E-F. The eight cable connectors can be plugged in either positions E4-7 and F4-7 or E11-8 and F11-8, depending on the layout of the specific installation. These positions are paralleled as shown in Figure 6-1 and as listed in Table 6-1.

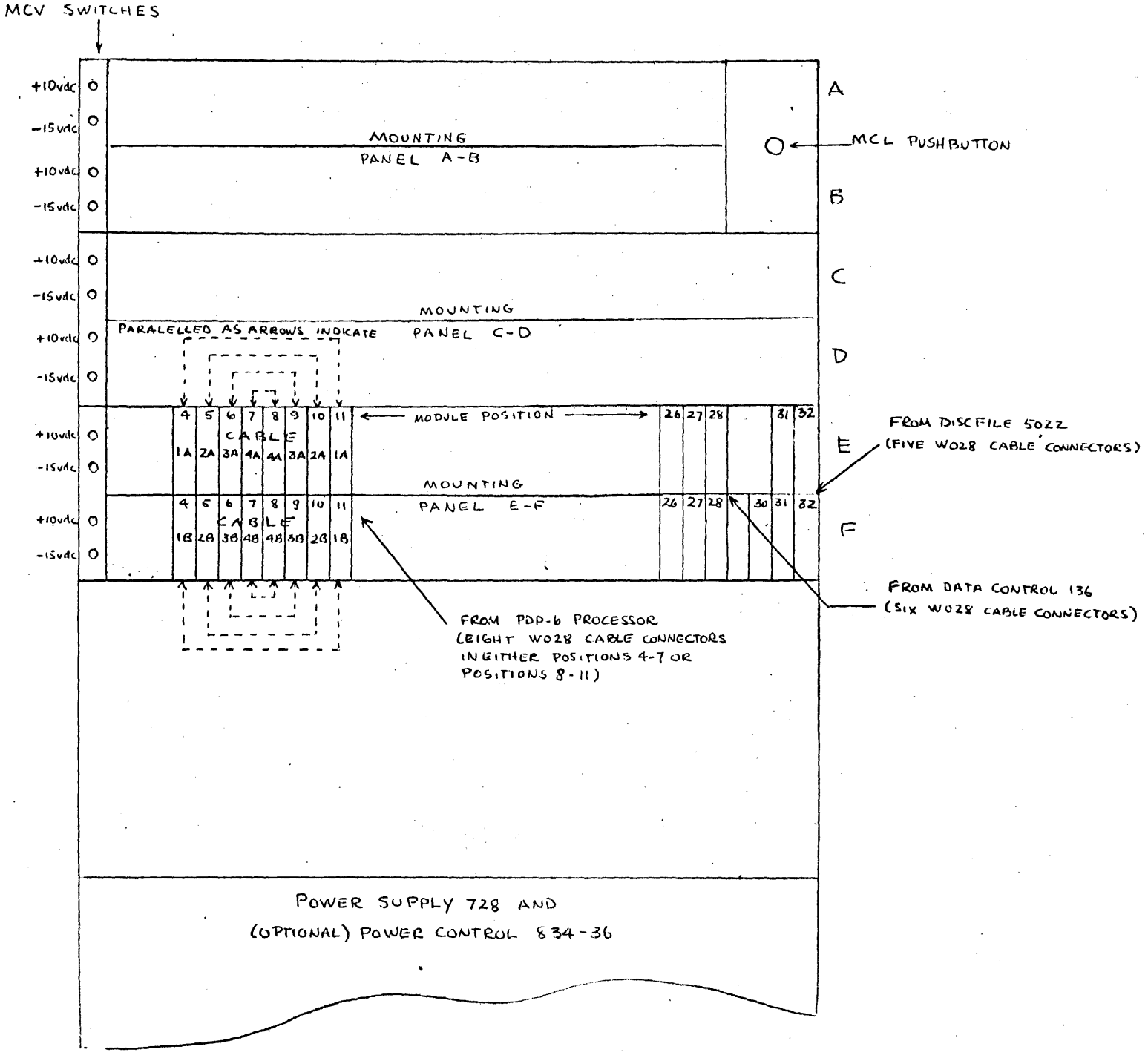


FIGURE 6-1 FRONT VIEW LAYOUT, DISCFILE CONTROL 270

CAUTION

It is essential that each of the eight W028 cable connectors be connected to the correct parallel pair of receptacles. The cable connectors should each be marked with the correct cable number. Check this cable number against Table 6-1 below or against module location diagram UML-D-270-0-11-sheet 2.

TABLE 6-1 PROCESSOR CABLE CONNECTIONS

Cable Connector	To Position	Or To Paralleled Position
1A	E4	E11
1B	F4	F11
2A	E5	E10
2B	F5	F10
3A	E6	E9
3B	F6	F9
4A	E7	E8
4B	F7	F8

2. Route the three cables from the Data Control 136 into the cabinet, and connect the six W028 cable connectors at the cable ends to positions E26-28 and F26-28 as shown in UML-D-270-11-sheet 2.

3. Route the cable from the discfile 5022 into the cabinet, and connect the five W028 cable connectors at the cable end to positions E31-32 and F30-32 as shown.

4. Secure the tie-down brackets over all cable connectors to prevent them from being accidentally loosened or dislodged.

5. A coiled ac power cable is taped to the fan at the bottom of the bay. Remove the tape and route the power cable out through the hole in the bottom of the bay, but do not plug it in.

6.5 POWER CONTROLS

To perform the preoperational check-out of the discfile control, the operator should be familiar with the controls described below.

a. Power Control Type 834-836 (Optional Equipment)

There are two controls on this unit, a circuit breaker and a LOCAL/REMOTE switch. When the switch is in the LOCAL position, the processor has no effect on discfile control power turnon. Power can then be turned on or off by means of the circuit breaker. The LOCAL position is used primarily for maintenance purposes.

For normal operation, it is usually more convenient to leave the circuit breaker on and the switch in the REMOTE position. Discfile control power is then controlled by the processor. When the processor is turned on, power is also applied to the discfile control. If the circuit breaker is turned off, no power is applied to the discfile control regardless of the position of the LOCAL/REMOTE switch.

b MCV Switches

There are four MCV (Marginal Check Voltage) switches at the left of each mounting panel. The top switch in each set governs the +10 vdc power lines for the upper row of modules, the second switch governs the -15 vdc lines for the upper row of modules, the third and bottom switches respectively govern the +10 vdc and -15 vdc lines for the lower row of modules.

This arrangement is an aid to troubleshooting because it permits selective application of marginal check voltages. In the FIXED position (down), each MCV switch connects the associated module row to the fixed supply voltage from the internal 728 Power Supplies. In the MARGINAL position (up), each MCV switch connects the associated module row to the variable MCV voltage from the processor. This variable MCV voltage can be adjusted at the MCV controls over the operator's control panel of the PDP-6.

6.6 PREOPERATIONAL CHECKOUT

Before using the discfile control, make sure that the system turns on and off properly and that the correct voltages are present at all mounting panels. The following check-out procedure should be carried out after completing the cable connections described under Cable Connections. All voltages are measured from chassis ground with a Multimeter (Simpson Model 260A, Triplett Model 630NA, or equivalent).

1. Put the LOCAL/REMOTE switch in LOCAL position.
2. Turn power circuit breaker OFF.
3. Plug in power cable to 110-volt ac outlet.
4. Put all MCV switches in FIXED position.
5. Turn power circuit breaker ON.
6. Check the fixed (internal) supply voltages at terminals A, B, and C of the modules that occupy the extreme right positions (viewed from the front) in both the upper and lower module rows of each mounting panel.

Terminal	Voltage
A	+10 vdc
B	-15 vdc
C	(Ground)

The +10-vdc fixed supply voltages should be between +9.5 vdc and +11.5 vdc. The -15 vdc should be between -14.5 vdc and -16.5 vdc. If either voltage falls outside the specified range, the 728 Power Supply probably needs maintenance.

7. With the PDP-6 processor turned off, put the LOCAL/REMOTE switch in the REMOTE position. Check each of the two fixed voltages at the terminals specified in step 6. No voltage should be present.

8. Turn on the PDP-6 processor. Again check the two fixed voltages. The same voltage noted in step 6 should be present at each of the terminals checked.

9. Put the LOCAL/REMOTE switch in the LOCAL position.

10. Put the MCV switches that control the +10-vdc lines (top switch and third switch at the left of each mounting panel) in the MARGINAL position.

11. Make the following settings at the MCV controls above the PDP-6 operator's control panel:

a. Set polarity switch to +10-volt position.

b. Adjust Variac until marginal check voltage meter indicates +5 vdc.

12. Check the +10-vdc voltage at the terminals specified in step 6. It should coincide with the +5-vdc voltage shown on the marginal check voltage meter.

13. Return the MCV switches for +10 vdc to the FIXED position.

14. At the PDP-6 MCV controls, set the MCV polarity switch to the -15-volt position and adjust the Variac until the marginal check voltage meter indicates -8 vdc.

15. Set the MCV switches which control the -15-vdc lines (second switch and bottom switch at the left of each mounting panel) to the MARGINAL position.

16. Check the -15-vdc voltage at the terminals specified in step 6. It should coincide with the -8-vdc voltage shown on the marginal check voltage meter.

17. Return the MCV switches for -15 vdc to the FIXED position.

18. Return the LOCAL/REMOTE switch to REMOTE.

19. Return the polarity switch at the PDP-6 to the off position.

20. Turn off the PDP-6.

CHAPTER 7

PROGRAMMING

This chapter tabulates the coding of the four I/O instructions used with the Discfile Control 270, and gives a listing and brief explanation of five short subroutines.

7.1 INSTRUCTION CODING

The discfile system is programmed by applying the four basic PDP-6 I/O instructions to the Discfile Control 270. The coding and bit-significance of each of these four instructions are tabulated in Tables 7-1 through 7-4 below.

a DATAI 270 Instruction

The DATAI 270 instruction permits the processor to sample the contents of the data accumulator DA0-19 and the contents of the bit-shift counter BSC0-4. This instruction is used primarily for maintenance purposes.

TABLE 7-1 DATAI 270 CODING

Bit No.	Samples	Bit No.	Samples	Bit No.	Samples
11	DA12	19	DA1	28	DA10
12	BSC0	20	DA2	29	DA11
13	BSC1	21	DA3	30	DA13
14	BSC2	22	DA4	31	DA14
15	BSC3	23	DA5	32	DA15
16	BSC4	24	DA6	33	DA16
17	DA19	25	DA7	34	DA17
18	DA0	26	DA8	35	DA18
		27	DA9		

b CONI 270 Instruction

The CONI 270 instruction permits the processor to sample the contents of the command buffer, the file-control octoflop, the nine error-status flags, and five miscellaneous signals: MCL, MRG, MRB, ALM, and ~OPR. The instruction is perhaps most frequently used for sampling the states of the error-status flags and the file-control octoflop, but it also provides a useful aid for maintenance. Note that the CONI 270 bits that sample the command buffer do not correspond to the CONO 270 bits that load the command buffer; the CONI 270 instruction samples the buffer into the left half of the word, but the buffer is loaded from the right half of the CONO instruction (see Table 7-2 below).

TABLE 7-2 CONI 270 CODING

CONI Bit No.	Samples Command Buffer Flip-flop or Miscellaneous Signal Marked with an *.		Set by CONO Bit No.
3	MCL*	Master Clear	
4	CLR	Clear	22
5	END	End	23
6	xxx	(Presently unused)	
7	CM0	Command Code Bit 0	25
8	CM1	Command Code Bit 1	26
9	MRG*	Meter Reading Good	
10	MRB*	Meter Reading Bad	
11	EFE	Enable File Error	29
12	EES	Enable End of Sector	30
13	EFR	Enable File Ready	31
14	EIS	Enable Idle State	32
15	PIA0	Priority Interrupt Assignment Bit 0	33
16	PIA1	Priority Interrupt Assignment Bit 1	34
17	PIA2	Priority Interrupt Assignment Bit 2	35

TABLE 7-2 CONI 270 CODING (continued)

CONI Bit No.	Samples File-Control Octoflop State	
18	IDS	Idle State
19	SNA	Select New Address
20	ADT	Address Terminated
21	DFR	Discfile Ready
22	ALS	Alert State
23	CMS	Command Selected
24	SCS	Sector Started
25	SCE	Sector End
CONI Bit No.	Samples Error-Status Flag or Miscellaneous Signal Marked With *.	
26	DCE	Data Clock Error
27	CME	Command Error
28	WLE	Write Lock Error
29	ADE	Address Error
30	ALM*	ALARM
31	DRL	Data Request Late
32	RCE	Read Compare Error
33	PER	Parity Error
34	FER	File Error
35	~OPR*	NOT OPERABLE

c DATAO 270 Instruction

The DATAO 270 instruction loads a 19-bit address from the processor into the data accumulator of the Discfile Control 270, and then causes the discfile control to transmit a 21-bit address to the Discfile 5022. (One of the two additional bits is DA12; this bit is always 0, so it can be generated within the discfile control. The second additional bit is the address parity bit, also generated within the discfile control.) After receiving all 21 bits of the address, the discfile initiates an address seek operation to locate the data sector specified by the address.

The discfile control can accept a DATAO 270 instruction only when the file-control octoflop is in either the IDS state or the ADT state, and when furthermore the MCL pushbutton is not operated.

TABLE 7-3 DATAO 270 CODING

DATAO Bit No.	Is Loaded Into	DATAO Bit No.	Is Loaded Into
17	DA19 (Read-Next-Sector Bit)	27	DA9
		28	DA10
18	DA0	29	DA11
19	DA1	--	DA12 (Loaded with 0 generated within discfile control)
20	DA2		
21	DA3		
22	DA4	30	DA13
23	DA5	31	DA14
24	DA6	32	DA15
25	DA7	33	DA16
26	DA8	34	DA17
		35	DA18

d CONO 270 Instruction

The processor sends 15 bits of control information to the discfile control at each CONO 270 instruction. Twelve of these 15 control bits are loaded into the command buffer. The remaining three control bits set no flip-flops in the discfile control, but are instead used directly as control inputs.

Note that a CONO 270 instruction to the discfile control will not affect the command code bits CM0 and CM1 unless the Change Commands Enable level CCE is asserted. This level is not asserted when any of the following conditions are present:

ALM	ALARM (discfile in alarm state)
~OPR	NOT OPERABLE (discfile not operable)
ALS	Alert State

CMS	Command Selected State
SCS	Sector Started State
SCE	Sector End State

The command codes can be changed at any of the first four states of the file-control octoflop cycle, IDS, SNA, ADT, or DFR state, but once the ALP pulse has advanced the octoflop to ALS state, no further change of command code is permissible.

TABLE 7-4 CONO 270 CODING

CONO Bit No.	Sets Command Buffer Bit (except *)	Logical Function
19	Direct Control Input*	When coded 1, produces DTC, SND, and GIVE A CHARACTER pulses at PTC time. (Loads an 18-bit character from the Data Control 136 into the discfile control.)
20	Direct Control Input*	When coded 1, produces a TAKE A CHARACTER pulse at PTC time. (Signals the Data Control 136 to accept an 18-bit character from the discfile control.)
21	Presently unused*	
22	CLR	Causes the current command to terminate at the end of the first 128-word data sector that is processed, and sends the discfile an END and a CLEAR at ESP 2 time.
23	END	Causes the current command to terminate at the end of the first 128-word data sector that is processed, and sends the discfile an END at ESP 2 time.
24	Presently unused*	

TABLE 7-4 CONO 270 CODING (continued)

CONO Bit No.	Sets Command Buffer Bit (except *)	Logical Function
25	CM0	Command Code bit 0
26	CM1	Command Code bit 1 CM0 CM1 Command Specified 0 0 NOP - No Operation 0 1 RED - Read 1 0 WRT - Write 1 1 RDC - Read Compare
27	Presently unused*	
28	Direct Control Input*	When coded 1, produces an SCL pulse at PTC time. (Clears the nine error-status flags: ADE, CME, DCE, DRL, FER, PER, RCE, SEF, and WLE.)
29	EFE	Enables FER(1), PER(1), RCE(1), or DRL(1) to initiate a priority interrupt request.
30	EES	Enables SEF(1) to initiate a priority interrupt request.
31	EFR	Enables DFR(1) to initiate a priority interrupt request.
32	EIS	Enables IDS(1) to initiate a priority interrupt request.
33 34 35	PIA0 PIA1 PIA2	The discfile control is assigned a priority interrupt request channel corresponding to the octal contents of PIA0-2. The lower the channel number, the higher the priority. Channel 1 has the highest priority, channel 7 the lowest. If all three PIA bits are left 0, no priority interrupt channel is assigned.

7.2 PROGRAMMING EXAMPLES

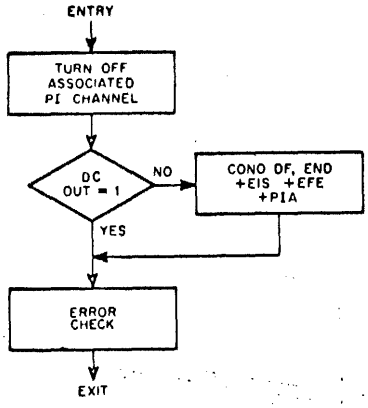
Five representative examples are included here to illustrate how the discfile programming information given above is used. The examples are intended to acquaint persons new to PDP-6 with several standard programming techniques. The reader is urged to review the programming discussions for the priority interrupt and input-output systems and the 136 Data Control in F-65 (pp. 40, 41-43 and 65-58) before proceeding.

Example 1: Interrupt service routine for the 136 Data Control on BLKO/BLKI overflow.

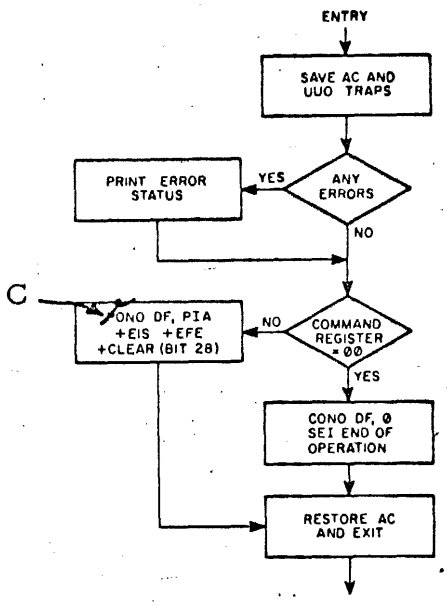
```
DCINT: 0                              JSR at PI channel stores PC.
       CONO PI 1000 + PIA             PIA signifies a bit for one of the seven PI channels
       CONSO DC, OUT                  Turn off PI channel.
       CONO DF, EIS + EFE + PIA + END Bit   Send END signal if reading.
       CONSZ DC, DCLB                 Check DC error flag.
       JSR ERROR                      Go to an error routine.
       JEN,@ DCINT                    EXIT.
```

The first example is a subroutine that returns the discfile's 136 Data Control to inactive status upon completion of a block data transfer (BLKO or BLKI). When the last word is transferred, a jump to this subroutine (JSR) will occur. The 1000 in the first CONO instruction sets a 1 in bit 26 of the PI effective address, which turns off the PI channel specified by the PIA bits (see F-65, p. 41).

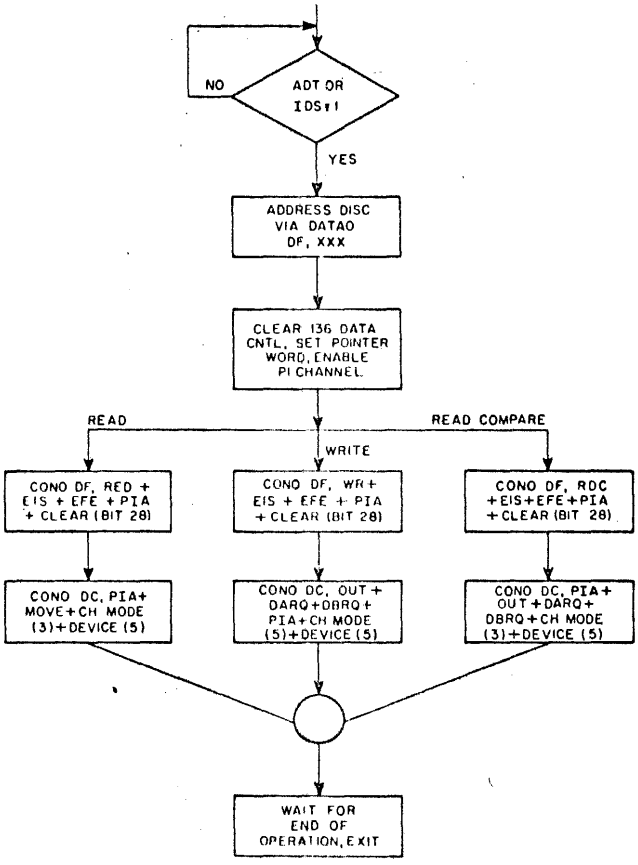
The significant feature of this program occurs at the decision block labeled DC OUT = 1 on the accompanying flow chart (Figure 7-1). The CONSO DC, OUT instruction says that if bit 26 of the control register is 0 (0 = In = write), skip the next instruction which turns off the discfile. The discfile turns itself off automatically except when reading.



Example 1
Data Control Interrupt Service



Example 2
Discfile Interrupt Service



Examples 3, 4 and 5
Read, Write, Read-Compare

Figure 7-1 Programming Examples

The remaining instructions check the DC error flag (DATA CLOBBERED bit 23). A 1 causes a jump to an error subroutine; all 0's permit a skip to the final jump and enable instruction, which causes the program to resume at DCINT using information at the PC location.

Example 2: Discfile interrupt service routine.

```
DFINT:  0                                  JSR at PI channel stores PC.
        MOVEM, DFSAVE                      Save AC 0.
        CONI DF, 0                         Get status.
        CONSZ DF, 1777                     Check error flags.
        JSR ERROR                          Go to error routine.
        TLNE RED + WR                      If command register is cleared,
        JRST DFINT2                        TERMINATE OPERATION.
        CONO DF, 0                         CLEAR DISC FILE.
        MOVE, DFSAVE                       Restore AC.
        JEN, @ DFINT                       EIT, operation complete.
DFINT2: CONO DF, EIS + EFE + 200 + PIA     Clear error flags.
        JRST DFINT2-2                      AND continue.
```

The second programming example gives an instruction sequence for terminating a TRANSFER at the 270 Discfile Control. The first instruction saves the contents of accumulator zero. The following three instructions get discfile status and check for errors just as in the previous example. Next, the left half of the command register is tested for 1's in the read and write bit positions. If no operation is set, the discfile is cleared. If an operation is in progress, the error flags are cleared via DFINTZ (CONO DF, + Bit 28) where 200 octal is clear bit 28. After the AC is restored, control reverts to the instruction indicated by the PC back at DFINT.

The remaining three examples show the three principal discfile operations read, write, and read compare, to be variations on a single theme (see flow diagram). All three begin with a wait cycle which lasts until the

discfile is free from any operation already in progress. When discfile status goes from busy to IDLE or ADDRESS TERMINATED, the address contained in UUO location 40 is transmitted to the discfile. The next instruction causes a jump to subroutine INIT which initializes the 136 Data Control and discfile control, that is, clears the 136 and sets up their PI channels. When the initialization subroutine is complete, a CONO Discfile sets up the required operation, read, write or read compare. The octal number 200 in the CONO Discfile instruction signifies bit 28, the clear bit. Octal 350 in the CONO DC instruction sets 1's in bits 28, 29, 30 and 32 of the 136 command register, signifying packing mode 3 (18-bit word), device number 5 (the discfile) DARQ and DBRQ are mnemonics for bits 25 and 26 of the 136 command register. Notice that in the read compare routine, the CONO DC instruction is the same as for write, where in both cases "out" means the 136 receives data from the CP. In the read compare case, however, the 270 Control Unit is conditioned to read. The result is that a data word read from the discfile into the 270 meets a data word from the 136 where the two are compared one bit at a time in the 270.

Data control uses character mode 3, device 5 to transfer disc information. Routines assume a BLKO for write and read compare, a BLKI for read in the interrupt location.

Example 3: Read, called via UUO, effective address of UUO equals DISC ADDRESS.

```
DFRED:  0                                   JSR PC stored by UUO subroutine
        CONSO DF, IDS + ADT                 Wait if operation
        JRST .-1                            in progress.
        DATAO DF, 40                        Send disc address.
        JSR INIT                            Jump to initialize subroutine INIT, below.
        CONO DF, RED + 200 + EIS + EFE + PIA  Command disc.
        CONO DC, 350 + MOVE + PIA           Setup 136 Data Control.
        JRST EXIT
```

Example 4: Write

```
DFWRT:  0                                   JSR PC stored by UUO subroutine
        CONSO DF, IDS + ADT                 Wait if operation
        JRSR, .-1                           is in progress.
        DATAO DF, 40                        Send disc address.
        JSR INIT                            Jump to initialize subroutine INIT, below.
        CONO DF, WR + 200 + EIS + EFE + PIA Command disc.
        CONO DC, OUT+DARQ+DBRQ+350+PIA      Setup 136 Data CNT control.
        JRST EXIT
```

Example 5: Read Compare

```
DFRDC:  0                                   JSR PC stored by UUO subroutine
        CONSO DF, IDS + ADT                 Wait if operation
        JRST .-1                            is in progress.
        DATAO DF, 40                        Send disc address.
        JSR INIT                            Jump to initialize subroutine INIT, below.
        CONO DF, RDC + 200 + EIS + EFE + PIA  Command disc.
        CONO DC, OUT+DARQ+DBRQ+350+PIA      Setup 136 data control.
        JRST EXIT
INIT:   0
        MOVE AC (BLKI/BLKO)                 ;initialize DC with BLKI for
        MOVEM AC, 40 + 2J                   ;input or BLKO for output
        MOVE AC, (JSR DCINT)                ;initialize interrupt channels
        MOVEM AC, 40 + 2J + 1               ;interrupt return
        MOVE AC, (JSR DFINT)                ;initialize discfile
        MOVEM AC, 40 + 2J
        CONO DC, 0                          ;clear data control
        CONO PI, 4000 + PI                  ;enable PI circuits in CP; bit 25 plus one of seven PI bits
        JRST @ INIT                         ;EXIT
```

CHAPTER ~~11~~ 8

SIGNAL GLOSSARY

~~(110 Ms. pages 1-113, 1a-c, 53a-b)~~

(122 Ms. pages 1-113, 1a-e, 8a-b, 51 a-b, 53 a-b, 71 a-b, 112 a-b).

CHAPTER ~~III~~ 8

SIGNAL GLOSSARY

The signal glossary contained in this chapter provides a comprehensive reference source for all input and output signals of the discfile system, for all logic levels and pulses used within the system, and for all control and status flip-flops. Any signal or flip-flop can be referenced alphabetically. Each signal description includes the logical input conditions that generate the signal as well as the logical functions performed by the signal. Thus by cross-referencing in the glossary, a complete understanding of both signal source and signal use can be obtained. The glossary entries also include the proper drawing reference for each signal; therefore, the glossary can be used in three ways: as a coordinate index to the engineering drawings; to trace a signal back to its source; and to trace a signal forward to its effects. Examples of each of these three uses are presented below.

Example 1

Problem: What is the ATP pulse, and where is it generated? Solution: Alphabetical lookup of ATP shows that ATP is the address termination pulse and that it is generated at -FC:C3.

Example 2

Problem: What types of malfunction could prevent the generation of the NOP level at the end of a read command? Solution: Alphabetical

lookup of NOP shows (under the caption "Generated by:") the input conditions that produce NOP: CM0(0) ∧ CM1(0). These conditions indicate that both the CM0 flip-flop and the CM1 flip-flop are in the 0 state. The next step is the alphabetical lookup of CM0-1 to determine the logical conditions that normally clear these two flip-flops. These logical conditions (under the caption "Cleared by:") are as follows:

IBR ∨:

CCB ∧ CCE . ∨:

ESP 1 ∧ END(1) . ∨:

ESP 1 ∧ CM0(1) ∧ DA RQ

Of these conditions, only ESP 1 and END(1) are relevant during the read command (when the CM0(1) level is not asserted, and when neither the IBR pulse nor the CCB pulse occurs). Evidently the trouble involves either a failure to generate the ESP 1 pulse, or else a failure to set the END flip-flop. For practical troubleshooting purposes, the specific signal traced would depend on further information obtained at each stage of the process. In the example given, the ESP 1 pulse might not appear at the proper time. The next step would then be to turn to glossary entry ESP 1, and to determine which of the ESP 1 generating conditions might be defective. This type of referencing gives the maintenance engineer random access to only the information he needs.

Example 3:

Problem: If the discfile control intermittently fails to generate the CSP pulse, what consequences might result? Solution: Alphabetical lookup

of CSP shows the uses to which CSP is applied. During the read command CSP initiates the READ pulse. During the write command CSP initiates the WRITE pulse, the SND pulse and the DTC pulse. During the read compare command CSP initiates the READ pulse, the SND pulse and the DTC pulse. (The same process can be carried out again for READ, WRITE, SND, or DTC, etc., for as many steps as are relevant to the particular inquiry.

Note to editor: Might be useful to leave out numbers below but to add page numbers – eg: "ALP ..... page 10"

SIGNAL GLOSSARY

DISCFILE CONTROL 270

Index to Signal Glossary

(1) ACE - Address Clock Enable Flip-flop (BS-D270-0-AC:D7)

The ACS and ADE flip-flops make up a 2-stage synchronizing network that controls the turn-on and turn-off of the address clock. During address transmission, this network ensures that the WCE pulses that clock the address out to the discfile are complete and in phase with the address clock CLK. The ACE(1) level enables a DCD gate at the set input of the ACS flip-flop. The first CLK pulse to arrive after the setup delay of the gate sets the ACS flip-flop. For a detailed description of the ACS-ACE synchronizing network refer to ACS below.

The outputs of the ACE flip-flop also control the input levels applied to bit DA0 of the data accumulator. During data operations, when ACE(0) is asserted, 0's are shifted into the left end of the accumulator as data is shifted right. However, during address transmission, when ACE(1) is asserted, the accumulator operates as a ring-shifter. Shortly after each address bit is clocked out of DA19 and into the discfile, the same address bit is also loaded back into DA0. All 20 address bits (but not the address parity bit) are thus reloaded into the data accumulator at the conclusion of the address transmission. Since the contents of the data accumulator can be directly sampled at the processor (by means of

the CONI 270 instruction), this ring-shift feature of the accumulator operation provides a useful means of checking the operation of the shift logic.

Set by:

ADS

Reset by:

FCL $\vee$:

DCK 2 $\wedge$ COH(1)

The FCL pulse provides the initial clear that ensures that the ACE flip-flop starts each operation in the 0 state. The DCK 2 pulse that is produced by the final data bit of the address (the 20th bit) sets the COH flip-flop. This enables the DCK 2 that is initiated by the 21st address bit (the parity bit) to reset both ACE and ACS, thereby turning off the address clock.

(2) ACS - Address Clock Sync Flip-flop (-AC:D8)

The ACS and ACE flip-flops make up a 2-stage synchronizing network that controls the turn-on and turn-off of the address clock. During address transmission, this network ensures that the WCE pulses that clock the address out to the discfile are complete and in phase with the address clock CLK.

The DATAO 270 instruction IOB DATA SET pulse produces a PTA pulse and thus starts a 15-μsec delay. The termination of this delay initiates the ADS pulse. The ADS pulse in turn starts the address sequence

by setting the ACE flip-flop. The first CLK pulse that arrives in time sets ACS to 1. (The transition of ACS is not necessarily in exact synchronization with the leading edge of the CLK pulse.) The next CLK pulse, and every succeeding CLK which occurs while ACS(1) is asserted, causes pulse amplifier C24 to produce a WCE clock in sync with the CLK.

The WCE clocks out the address data bits and parity bit to the discfile on the EDO and EDZ twisted pairs. During address transmission, every WCE clock also starts the DCK clock chain, producing in sequence the following pulses: DCK 1, DCK 0, DCK 2, DCK 2.1, and DCK 3.

Set by:

CLK $\wedge$ ADE(1)

Reset by:

FCL $\vee$:

DCK 2 $\wedge$ COH(1)

The FCL pulse provides the initial clear that ensures that the ACS flip-flop starts each operation in the 0 state. The DCK 2 pulse that is produced by the final data bit of the address (the 20th bit) sets the COH flip-flop. This enables the DCK 2 that is initiated by the 21st address bit (the parity bit) to reset both ACE and ACS, thereby turning off the address clock.

(3) ADE - Address Error Flip-flop (-SC:A3)

The CONI 270 samples the contents of ADE on IOB29.

Set by:

ERP $\wedge$ ADT(1)

Reset by:

SCL

The ERROR pulse is transmitted from the discfile through a twisted pair to the discfile control; there it is inverted and redesignated ERP. At the discfile control, the ERP pulse always sets the FER flip-flop. When the ERROR pulse arrives during the ADT state of the file status octoflop, the ADE flip-flop is also set. This not only tells the programmer that an error has been detected, but furthermore when the error has appeared, and therefore what general class of error has occurred.

Although the fact that ADE has been set somewhat narrows the range of possible errors, there are still at least six different types of address error that can set ADE. Indicator lights at the discfile often may further identify the specific type of address error that has caused the ERROR SIGNAL. Six such types of address error are tabulated below.

Address Error Tabulation

Error Type	Discfile Indicator Light	Error Description
Address Parity Error	PE	Address parity bit received by discfile does not yield odd parity. (Parity checked at receipt of address termination pulse.)
Address Termination Error	AT	Address termination pulse not received during specified time interval (10 ± 9 μsec after final bit of address).
Incomplete Address Error	IA	Address termination pulse received by discfile prior to receipt of complete address.
Address Transfer Error	T (Transfer)	Incorrect or non-existent address received at discfile, or contents of discfile shift register do not correspond to contents of discfile storage register (after parallel transfer from shift register to storage register).
Positioning Error	FAULT and DP (Disc Position)	Positioner unable to settle on correct track within 600 milliseconds after ATP pulse. This condition prevents transmission of the READY pulse to the discfile control.
Sector Error	FAULT and S (Sector)	Desired record not found within approximately 120 milliseconds after positioner settles on correct track. This condition also prevents transmission of the READY pulse.

(4) ADS - Address Selected (-FC:C2)

The ADS pulse sets the ACE flip-flop and the WDC flip-flop. The ACE and ACS flip-flops make up a synchronizing network that controls the turn-on and turn-off of the address clock. The ADS pulse starts the turn-on sequence by setting the ACE flip-flop.

The set of the WDC flip-flop by the ADS pulse governs the subsequent setting of the COH flip-flop. For both addresses and data, the assertion of the COH(1) level enables the discfile control to detect the occurrence of parity bits and to distinguish them from other bits of the address or data.

Generated by:

PTA plus 15-μsec delay

The DATAO 270 instruction sends new addresses to the discfile via the discfile control. The IOB DATA SET pulse of the DATAO instruction produces a PTA pulse and thus starts a 15-μsec delay. The termination of this delay steps the file-controloctoflop from IDS or ADT state to SNA state and simultaneously initiates the ADS pulse.

(5) ADT - Address Transmitted (-FC:B3)

Third state of the file-control octoflop. The octoflop is advanced from SNA state to ADT state after the last of the 21 address bits is transmitted to the discfile. For the octoflop to be advanced to DFR state, the ADT(1) level must be asserted.

New address data can be transferred into the data accumulator only when the octoflop is in either IDS state or ADT state. Neither the CDA pulse

nor the PTA pulse can be generated unless the octoflop is in one of these two states. For the octoflop to be stepped to the SNA state, it must also be in either IDS or ADT state. The CONI 270 samples ADT on IOB20.

Generated by:

ATP $\vee$

RDY plus 1.0-ms delay $\wedge$ ADE(0) $\wedge$ DFR(1) $\wedge$ NOP

The octoflop may enter the ADT state from either of the adjacent states, SNA or DFR. Approximately 15 μsec after the final address bit is transmitted to the discfile, the ATP pulse advances the octoflop from SNA state to ADT state.

If CM0-1 contains no command code (NOP), the octoflop will (provided that no address errors are present, i.e. that ADE(0) is asserted) oscillate between ADT state and DFR state. The octoflop is held in DFR state for one millisecond after each RDY pulse arrives from the discfile.

When the read-next-sector bit is 0, the octoflop is held in DFR state during only one millisecond of each 52-millisecond disc revolution. When the read-next-sector bit is 1, the octoflop is set to DFR for one millisecond after the header of each sector of the addressed track is sensed. But regardless of the state of the read-next-sector bit, if NOP is asserted, the octoflop is always returned to ADT state after one millisecond in DFR state.

The ADE(0) level must be asserted (indicating no address errors) in order to start the 1.0 millisecond delay that controls the duration of the DFR state. When no address errors are present, the termination of this

delay causes the octoflop to return from DFR to ADT state (if NOP), or else causes the ALP pulse to advance the octoflop from DFR to ALS state (if ~NOP). Note, however, that if there were address errors present (so that ADE(0) were not asserted), the octoflop could not even enter DFR state, but would instead end its cycle at ADT state.

→ Insert page 8b here

(6) ALERT (-TR:A7)

The ALERT pulse is sent to the discfile through a twisted-pair transmission line. It causes the discfile to increment the stored address by 1, and prepares the discfile logic to receive the forthcoming command signal (READ or WRITE as the case may be).

Generated by:

ALP

The discfile control produces an ALP pulse to advance the file-control octoflop from DFR state to ALS state (See ALP below.) After being amplified by pulseamplifier F29, the ALP pulse is redesignated ALERT and sent to the discfile.

(7) ALM - Alarm (-SC:B4)

When the ALM level is asserted, the FER flip-flop is set immediately. Provided that the priority interrupt enable switch EFE (a flip-flop switch) is set to 1, FER(1) produces a PIE level. The resulting priority interrupt request can notify the programmer that an alarm condition has occurred.

It is not permissible to change the command code in CM0-1 while an alarm condition is present. For this reason, ~ALM, the negation of

(5A) ALARM (-SC:B2)

The discfile sends an ALARM signal to the discfile control through a twisted-pair transmission line to indicate that an alarm condition has occurred. At the discfile control, the ALARM signal is applied to a switch filter and a pair of series-connected inverters and redesignated ALM. (See ALM below.)

the ALM level, is used as a necessary input condition for the generation of the CCE level. The CONI 270 samples ALM on IOB30.

Generated by:

ALARM (from discfile)

This signal is generated by the closure of a relay within the discfile. This relay closure shunts the discfile end of a twisted-pair line. (For normal operation, the circuit remains open.) Closure of the relay applies +10 vdc to a switch filter and two series-connected inverters, and thereby generates the ALARM signal at both polarities; see Figure'-SC:B2-5.

The discfile closes the ALARM relay under any of the following four conditions:

1) If the output of the positioner power supply in any disc unit falls below normal.
2) If a malfunction occurs in the flying-head air supply.
3) Should the ambient temperature in the logic unit or in any disc unit rise to the alarm level.
4) When any test mode switch is left in test position during normal operate mode.

Note that although any of these four ALARM conditions will always terminate the OPERABLE signal, absence of the ALARM does not guarantee the presence of OPERABLE. The ALARM signal usually implies the existence of an abnormal condition that carries with it the hazard of equipment damage should operations be continued without taking corrective action. The OPERABLE signal goes beyond this minimum standard, and indicates

not only that there is no ALARM, but furthermore that the discfile is operable - that it is ready to receive commands from the discfile control. See OPR below.

(8) ALP - Alert Pulse (-FC:B4)

The ALP pulse steps the file-control octoflop from DFR state to ALS state. Furthermore, the ALP pulse (amplified and redesignated ALERT) is sent through a twisted-pair transmission line to the discfile. The ALERT pulse causes the discfile to increment the stored address by 1 and prepares the discfile logic to receive the forthcoming command signal (READ or WRITE as the case may be).

The discfile control always sends the discfile the address of the sector immediately preceding the sector to be processed. The READY pulse is returned to the discfile control ~~when~~ after the header of this preceding "get ready" sector is sensed. The ALERT pulse then increments the stored address of the "get ready" sector, and thereby produces the correct address of the sector actually to be processed. The discfile compares the header of this sector with the incremented address, the extent of the compare depending on the state of the read-next-sector bit.

Within the discfile control, the ALP pulse starts a 15-μsec delay. Termination of this delay produces the CSP pulse that ends the ALS state of the octoflop and that actually transmits the READ or WRITE command pulse to the discfile.

Generated by:

RDY plus 1.0 ms delay ∧ ADE(0) ∧ DFR(1) ∧ ~NOP

Since the file status octoflop is stepped to ALS state by the ALP pulse, the same conditions of generation apply to the pulse as to the state. The significance of these generating conditions is described under ALS below.

(9) ALS - Alert State (-FC:B4)

Fifth state of the file-control octoflop. The ALS(1) level enables the CSP pulse to advance the octoflop from ALS to CMS state.

The octoflop remains in the ALS state for the 15-μsec interval between the ALP and the CSP pulses. The ALP pulse that begins the alert state prepares the discfile logic to receive a command signal (the READ or WRITE pulse as the case may be); the CSP pulse that ends the ALS state 15 μsec later actually transmits the READ or WRITE pulse to the discfile.

It is not permissible to change the command code in CM0-1 after the ALP has advanced the octoflop from DFR to ALS state. For this reason the ALS(0) level is used as a necessary input condition for the generation of the CCM level. The CONI 270 samples ALS on IOB22.

Generated by:

RDY plus 1.0 ms delay ∧ ADE(0) ∧ DFR(1) ∧ ~NOP

The octoflop enters DFR state at the arrival of the RDY pulse from the discfile. It remains in DFR state for one millisecond. This one-millisecond

window gives the programmer time to set up the command buffer and thereby to specify what operations are to be performed during the forthcoming sector. After one millisecond in DFR state, either the ALP pulse advances the octoflop to ALS state (if ~NOP is asserted, indicating that there is a command code in bits CM0-1 of the command buffer) or else the octoflop is returned to ADT state (if NOP is asserted, indicating that no command code is stored in CM0-1).

The ADE(0) level must be asserted (indicating no address errors) in order to start the 1.0-millisecond delay that controls the duration of the DFR state. When no address errors are present, the termination of this delay causes the octoflop to return from DFR to ADT state, or, in the alternative, produces the ALP pulse (at RDY plus 1.0 ms) which advances the octoflop from DFR to ALS state. Note, however, that if there were address errors present (so that ADE(0) were not asserted), the octoflop could not even enter DFR state, but would instead end its cycle at ADT state.

(10) ATP - Address Termination Pulse (-FC:C3)

The ATP pulse signals the completion of the address transmission to the discfile. It is sent to the discfile through a twisted-pair transmission line approximately 15 μsec after the transmission of the parity bit, the final bit of the 21-bit address. Within the discfile control, the ATP pulse advances the file status octoflop from SNA state to ADT state.

Generated by:

SNA(1) ∧ COH(1) . ∧ :

DCK 2 plus 15 μsec delay ∧ SNA(1)

Each of the 21 address bits is transmitted to the discfile by a DCK 1 clock pulse. Every DCK 1 clock is followed approximately 400 nsec later by a DCK 2 clock. The DCK 2 pulse shifts the contents of the data accumulator right one bit-position, thereby bringing the next address bit into position to be transmitted to the discfile. The DCK 2 pulse that shifts the parity bit of the address into position for transmission to the discfile also sets the COH flip-flop to the 1 state. The next DCK 1 clock then completes the address transmission by clocking the parity bit into the discfile; about 400 nsec later the final DCK 2 pulse of the address transmission cycle turns off the address clock (this DCK 2 pulse clears ACE, ACS, COH, and WDC) and starts a 15-μsec delay. Termination of this delay produces the ATP pulse. Note that the ATP pulse can be generated only if the octoflop remains in SNA state until the completion of the 15-μsec delay. (In the unlikely event that an IOB reset pulse were to be applied to the discfile control during the 15-μsec delay, it would be necessary to prevent generation of ATP.)

(11) BCO - Bit Counter Overflow (-AC:D1)

During address transmission, the BCO level indicates that all 20 data bits of the address have been clocked out to the discfile. For data operations (read, write, or read compare), the BCO level is asserted at the 18th bit of each 18-bit data character.

The single DCK 2 pulse that occurs during the short period while BCO is asserted always complements the WDC flip-flop. If the WDC flip-flop

contains 1 before being complemented, then the same DCK 2 pulse also sets the COH flip-flop and increments the word counter WCT. Note that although the BCO level is not itself applied to the COH flip-flop set logic, two other ANDed conditions which are so applied are the exact logical equivalent of BCO. (The BSC0(1) and BSC2(1) inputs are used instead of BCO only to simplify wiring, and not from any logical necessity.)

During the two output commands, write and read compare, the BCO level is ANDed with SCS(1) to enable the DCK 2.1 clock to produce the SND and DTC pulses. The SND pulse gates an 18-bit character from the Data Control 136 into the data accumulator of the discfile control. The DTC pulse sends the Data Control 136 a Give a Character pulse to shift the next output character into position.

At the DCK 3 pulse that follows about 800 nsec after the DCK 1 pulse which first asserts BCO, the BCO level causes both the BSC0 and BSC2 bits of the bit counter to be reset (thus terminating the BCO level).

Generated by:

BSC0(1) $\wedge$ BSC2(1)

The BCO level is always first asserted at the DCK 1 pulse that causes the bit counter to reach the count of 20. For address transmission, the count starts at 0, and 20 bits are transmitted to the discfile before the count reaches 20. For data operations, the count starts at 2, so only 18 bits need be processed to generate the BCO.

(12) BSC0-4 - Bit-Shift Counter (-AC:C2-5)

The bit-shift counter is a 5-bit count-of-20 binary counter which starts counting in either the 0 state or the 2 state depending upon whether the discfile control is sending out an address to the discfile, or is executing a data operation (a read, write, or read compare command). During address transmission, the bit-shift counter counts the number of address bits that have been clocked out to the discfile; the overflow level BCO is asserted at the 20th bit. For data operations, the overflow occurs at the 18th bit of each 18-bit data character. Besides producing the BCO level, the outputs of the bit-shift counter are also used directly to set the COH flip-flop (see BCO above).

The outputs of the bit-shift counter also determine the time at which the Take a Character pulse is to be sent to the Data Control 136. During the read command, the DCK 1 clock that occurs while the counter contains 19 initiates the Take a Character pulse. This same DCK 1 pulse increments the counter to its overflow count of 20. The Take a Character pulse causes the data control to strobe the input character into its low order accumulator bits, and (if the character is the first of the two 18-bit characters making up the 36-bit data word) to shift it left to make room for the second character.

The CONI 270 instruction samples the contents of the bit-shift counter on IOB12-16.

Cleared by:

Entire counter BSC0-4: FCL

BSC0 and BSC2 only: DCK 3 $\wedge$ BCO

Incremented by:

DCK 1 $\wedge$ COH(0)

BSC3 Set by:

NDP 2 $\vee$:

DCK 2 $\wedge$ SCS(1) $\wedge$ BSC0(1) $\wedge$ BSC2(1)

The FCL pulse clears the bit-shift counter before each new address is transferred out, and at the beginning of each new data sector. When the count reaches 20, the BCO level is asserted; the BSC0 and BSC2 bits are then in the 1 state, and the other three bits of the counter contain 0s. The BSC0 and BSC2 bits are cleared by DCK 3 (about 800 nsec after the DCK 1 pulse that initiates the BCO level).

The counter is incremented only at DCK 1 time. The COH(0) condition ensures that no count is made for parity bits during either address transmission or data operations. (See COH below.)

The BSC3 set conditions enable the counter to act as a count-of-18 counter rather than a count-of-20 counter. For data operations, these set conditions cause the counter to begin its count in the two state. The NDP 2 pulse sets the counter to two at the beginning of each data sector. The second line of set conditions listed above sets the counter to two prior to the beginning of each 18-bit data character after the first one; BSC3 is set at the final overflow count of each character, thus preparing

the counter for the next character. The counter reaches 20, its overflow count, at DCK 1 time; it is incremented to 22 at DCK 2 time; it is reset to its correct initial state of 2 at DCK 3 time. This operating sequence is summarized below in tabular form.

Bit-Shift Counter Sequence - Data Operations

Counter State	Initiated by
0	FCL (Generated at NDP 1 time.)
2	NDP 2 (First character of sector.)
3, 4, ... 16, 17, 18, 19	DCK 1 corresponding to each data bit increments counter by 1.
20	DCK 1 for final (18th) data bit.
22	DCK 2 $\wedge$ SCS(1) $\wedge$ BSC0(1) $\wedge$ BSC2(1) (BSC3 set - to prepare counter for next data character.)
2	DCK 3 $\wedge$ BCO (Clear BSC0 and BSC2.)
3, 4, ... 16, 17, 18, 19	DCK 1 corresponding to each data bit (of second 18-bit character) increments counter by 1.

etc.

For address transmission, operation of the counter is somewhat more simple.

Bit-Shift Counter Sequence - Address Transmission

Counter State	Initiated by
0	FCL (Generated at PTA time.)
1, 2, ..., 17, 18, 19	DCK 1 corresponding to each address bit increments counter by 1.
20	DCK 1 for (20th) address bit (the bit immediately preceding the address parity bit).
0	DCK 3 ∧ BCO (Clear BSC0 and BSC2).

Note that the counter is not stepped from 20 to 22 as in the case of data operations. This is because the SCS(1) level is not asserted.

(13) CCB - Clear Command Buffer (-CM:C4)

The CCB pulse always clears the nine low-order bits of the command buffer: CLR, EES, EFE, EFR, EIS, END and PIA0-2. If the CCE level is asserted, the CCB pulse also clears the two command code bits CM0 and CM1. Clearing these two bits permits the PTC pulse to load them with the next command.

Generated by:

IBR ∨ :

IOB CONO CLR ∧ SEL

Although CCE must be asserted for CM0-1 to be cleared by the IOB CONO CLR ∧ SEL condition, these two bits are also cleared by IBR alone (regardless of the state of CCE).

(14) CCE - Change Commands Enable (-CM:D2)

This level must be asserted to permit immediate program control of the contents of the two command code bits, CM0 and CM1. The CCE level enables CCB to clear CM0-1, and enables PTC to gate in a new command code on IOB7 and IOB8. (The IOB7 and IOB8 inputs correspond to bits 25 and 26 of the CONO instruction word.)

Generated by:

OPR ∧ ~ALM . ∧:

ALS(0) ∧ CMS(0) ∧ SCS(0) ∧ SCE(0)

To change commands, the discfile must be operable, and must not be sending an alarm signal to the control. Furthermore, the discfile control must be in an appropriate status for changing commands. During any of the first four states of the file-control octoflop (IDS, SNA, ADT, or DFR) it is permissible to send a CONO instruction which changes the command in CM0-1. However, once the Alert pulse has been sent to the discfile, the octoflop is stepped into ALS state, and no further change in command code is permissible. The command code in CM0-1 cannot be changed during any of the last four states of the octoflop (ALS, CMS, SCS, or SCE).

(15) CDA - Clear Data Accumulator (-AD:A8)

The CDA pulse clears the data accumulator DA0-19.

Generated by:

IBR $\vee$: IOB DATA CLR $\wedge$.

SEL $\wedge$.

MCL(0) $\wedge$.

IDS(1) $\vee$ ADT(1)

The IBR pulse always produces a CDA pulse as part of the initial clear operation. During the DATAO instruction, IOB DATA CLR also initiates a CDA provided that certain necessary conditions are met. The discfile control must be selected by the device selection code of the DATAO instruction. Furthermore, the master clear switch must be open, and the discfile control must be in either the idle state or in the address terminated state.

(16) CLEAR (-TR:A4)

At the end of any sector during which the CLR flip-flop is set, the discfile control sends a CLEAR pulse to the discfile. (See CLR below.) (The CLEAR pulse is sent to the discfile through the same twisted-pair line upon which the READY pulse is received.) The CLEAR pulse signals the discfile to end the power-hold state by removing power from the addressed positioner.

The switching time (exclusive of motion time, confirmation time, and latency time) required to change disc and/or position field ranges from about 20 to 30 ms. This switching time can be reduced to about 7 ms if power can be completely removed from the previously addressed

positioner before the discfile starts to reapply power for the new address.

The CLEAR pulse starts the positioner-power turn-off. Note that the programmer must not send a new address to the discfile control during the 25-ms powering-off period following the CLEAR. To do so would cause an error condition. The programmer must use this 25 ms elsewhere for programming some other device. After the 25 ms have elapsed, the programmer can return to the discfile control with a new address.

Generated by:

ESP 2 $\wedge$ CLR(1)

Regardless of when the CLR flip-flop is set, the CLEAR pulse is always sent to the discfile at the ESP 2 pulse that follows the end of the sector. Note that when the CLR flip-flop is set, the END flip-flop is also set automatically. As a result, a CLEAR pulse will never be sent to the discfile unless an END pulse is sent at the same time. (An END pulse may, however, be sent without a CLEAR pulse.)

(17) CLK - Clock (-AC:D8)

The CLK pulses are generated every 1.33 μsec so long as the discfile control is turned on. However, the CLK pulses are used only during the address transmission portion of the discfile control operating cycle. The ACS and ACE flip-flops make up a synchronizing network that controls the turn-on and turn-off of the address clock. (The CLK pulses are not turned off, but they are gated through to where they take effect only during the address transmission; see ACS above.) The ACE(1) level enables a

DCD gate at the set input of the ACS flip-flop. The first CLK to arrive after the gate is enabled sets the ACS flip-flop. So long as the ACS(1) level is asserted, each succeeding CLK pulse produces a WCE pulse. The WCE clocks out the address data bits and parity bit to the discfile and also starts the DCK clock chain.

Generated by:

Variable Clock B02

The CLK pulses are standard 100-nsec pulses that are generated at 1.33-μsec intervals by variable clock B02, an R401 stable RC-coupled oscillator.

(18) CLR - Clear Flip-flop (-CM:B3)

When the CLR flip-flop is set, the ESP 2 pulse that follows the end of the sector sends a CLEAR pulse to the discfile. At the discfile, the CLEAR pulse ends the power-hold state by removing power from the addressed positioner. This may save time in switching to the next address (See CLEAR Pulse above.)

When the CLR flip-flop is set, the END flip-flop is also set automatically by a diode shunt. The END flip-flop can, however, be set without setting CLR. The result of this arrangement is that whenever the discfile control sends a CLEAR pulse to the discfile, it also sends an END pulse.

When the CLR flip-flop is set, the negation of the CLR(0) level prevents the generation of a RDY pulse even though a READY may arrive from the discfile. The RDY pulse is inhibited when CLR is set ~~because~~ to prevent the CLEAR pulse from generating a RDY pulse.

Generated by:

PTC $\wedge$ IOB4(1)

The CLR flip-flop may be set by programming a CONO 270 instruction with a 1 in bit 22. To reset CLR, give a CONO 270 with 0 in bit 22. (The IOB4 input corresponds to bit 22 of the CONO instruction word.)

(19) CME - Command Error Flip-flop (-SC:A4)

The CONI 270 samples the contents of CME on IOB27.

Set by:

ERROR (from discfile) $\wedge$ CMS(1)

Reset by:

SCL

The ERROR pulse is transmitted from the discfile through a twisted pair to the discfile control; there it is inverted and redesignated ERP. At the discfile control, the ERP pulse always sets the FER flip-flop. When the ERROR pulse arrives during the CMS state of the file-control octoflop, the CME flip-flop is also set. This not only tells the programmer that an error has been detected, but furthermore, when the error has appeared, and therefore what type of error has occurred.

The End error is the most common discfile error condition that could cause the transmission of an ERROR pulse to the discfile control during CMS state and thus set CME. If no END, READ, or WRITE pulse is received by the discfile within the 8-μsec guard slot following the final pulse of a sector, an End error occurs. The ERROR pulse is then

transmitted to the discfile control, and the END indicator at the discfile is lighted. Since, for multi-sector operations, the discfile control is returned to CMS state seven microseconds after the final DCK 1 pulse of a sector, the control would reach CMS state prior to the arrival of the ERROR pulse induced by the End error.

(20) CM0-1 - Command Register (-CM:B1,2)

The command register CM0-1 includes 2 of the 11 bits making up the command buffer. These two bits store the command codes that govern most operations performed by the discfile control. For a description of the remaining nine bits of the command buffer, refer to CLR, EES, EFE, EFR, EIS, and PIA0-2 below.

The outputs of the two command register bits CM0-1 are applied to a half binary-to-octal decoder, which then asserts one of four output command levels depending upon the states of CM0 and CM1. These four command levels are tabulated below.

Command Register Decoding

CM0	CM1	Octal Contents	Output Level Asserted by Decoder	
0	0	0	NOP	(No Operation)
0	1	1	RED	(Read)
1	0	2	WRT	(Write)
1	1	3	RDC	(Read Compare)

For a description of the uses to which these command levels are put throughout the discfile control, refer to NOP, RDC, RED, and WRT below.

Besides serving as inputs to the binary-to-octal decoder, three of the four output levels from flip-flops CM0-1 also are used directly as control signals. The levels CM0(0), CM0(1), and CM1(1) are so used; CM1(0) is at present unused. The table below shows that each of these three levels is asserted whenever either of two command levels is asserted. Because certain operations of the discfile control are common to two commands, it simplifies the system logic to use these command-pair output levels as control functions.

Command-Pair Control Functions

Output Level	Asserted During Either	Used to Generate
CM0(0)	NOP $\vee$ RED	Take a Character, set WCT7
CM0(1)	WRT $\vee$ RDC	CM0-1 Clear, DTC, SND
CM1(1)	RED $\vee$ RDC	PER, RDS, READ

The use of CM0(0) to set WCT7 and to generate the Take a Character pulse is logically equivalent to the use of RED, because in both cases the CM0(0) level is ANDed with SCS(1), which can never be asserted during NOP. (The CM0(0) level is used instead of the RED level only to ~~simplify wiring~~ reduce signal propagation time and not from any logical necessity.)

The three signals controlled by CM0(1) are common to both the write and the read compare commands. The clear conditions for CM0-1 are

described below under the heading "Cleared by:". The DTC pulse sends a Give a Character pulse to the Data Control 136; that pulse shifts the next 18-bit data character into position for sampling. The SND pulse reads the current 18-bit data character into the data accumulator of the discfile control.

Similarly, the three signals enabled by CM1(1) are common to both the read and the read compare commands. The parity of incoming data from the discfile is tested during both commands. Consequently the PER flip-flop set logic requires the assertion of CM1(1). The RDS signal puts the DCK clock chain under control of incoming data pulses (ED0, EDZ, ODO, and ODZ pulses) during read and read compare commands. The third and final operation that is governed by the assertion of CM1(1) is the transmission of READ pulses to the discfile.

Set by:

CM0: PTC $\wedge$ CCE $\wedge$IOB7(1)

CM1: PTC $\wedge$ CCE $\wedge$IOB8(1)

The processor instruction CONO 270 loads command codes into CM0 and CM1 via IOB7 and IOB8. (The IOB7 and IOB8 inputs correspond to bits 25 and 26 of the CONO instruction word.) Note, however, that unlike the other nine bits of the command buffer, which can be loaded at any time, the two command code bits can be loaded only when the CCE level is asserted.

Cleared by:

IBR $\vee$:

CCB $\wedge$ CCE . $\vee$:

ESP 1 $\wedge$ END(1) . $\vee$:

ESP 1 $\wedge$ CM0(1) $\wedge$ DA RQ

The IBR clear condition is initiated by IOB reset and is an unconditional clear. The CCB clear pulse occurs at IOB CONO CLR during each CONO 270 instruction, but does not always clear CM0-1. The contents of these two command code bits are under immediate program control only during a specific portion of the operating cycle, i.e. that portion of the cycle when CCE is asserted.

It is, however, possible to clear CM0-1 at the conclusion of any sector of data. This can be done (third line of clear conditions above) by simply setting the END flip-flop (with a CONO 270 which contains 1 in bit 23; see END below). The next ESP 1 pulse then clears CM0-1, and thus enables the ESP 2 pulse, which follows 2 μsec later, to reset the discfile control to IDS state. For write and read compare commands, the current sector runs its normal course, and no ESP pulses will be generated until the entire sector is traversed. However, for read commands the assertion of END(1) immediately sets WCT7, turning off the clock and starting the 5 μsec delay that terminates with the ESP 1 pulse.

The fourth and last line of the clear conditions listed above governs the automatic clear that occurs at the end of the final sector of output data. The CM0(1) level (see table above) indicates that the current command is

either write or read compare. The DA RQ level from the Data Control 136 is asserted as soon as the data control has transmitted the last of its data to the discfile control.

Note that the output command nevertheless continues to the end of the current sector. Even if the output data ends prior to the normal end of sector, clock pulses continue to be generated, and the bit and word counters continue to count until the 128 full 36-bit words have been traversed. If the command is write, 0's are written for the rest of the sector after the output data ends. If the command is a read compare, a read compare error is generated when the output data ends (unless the input data from the discfile ends at the same point as the output data from the Data Control 136).

(21) CMS - Command Selected (-FC:B5)

Sixth state of the file-control octoflop. The octoflop may enter CMS state from either ALS state or SCE state. When processing the first sector of data, the octoflop advances from ALS state to CMS. At the beginning of each subsequent data sector, the octoflop returns from SCE state to CMS state.

The same conditions which step the octoflop to CMS state also start a 200-μsec delay. If still asserted at the termination of this delay, the CMS(1) level enables the transition of the delay output to produce an NDP 1 pulse and (1 μsec later) an NDP 2 pulse. In the unlikely event that an IOB reset pulse were to be applied to the discfile control during

the 200-μsec delay, it would be necessary to prevent both NDP 1 and NDP 2. (Because the IOB reset would reset the octoflop from CMS to IDS state, CMS(1) would not be asserted at the end of the delay, and therefore the NDP pulses would not be generated.)

It is not permissible to change the command code in CM0-1 after the ALP has advanced the octoflop from DFR to ALS state. For this reason, the CMS(0) level is used as a necessary input condition for the generation of the CCM level.

If an ERROR signal arrives from the discfile while the octoflop is in CMS state, the CME flip-flop is set, indicating that the error is a command error. The CONI 270 samples CMS on IOB23.

Generated by:

CSP $\wedge$ ALS(1) . $\vee$:

ESP $\wedge$ $\sim$NOP

At the beginning of each input or output sequence, the same CSP that transmits the READ or WRITE command pulse to the discfile also advances the octoflop from ALS state to CMS state. This is represented by the first condition above.

The second condition for CMS state represents the processing of additional data sectors. At the end of each sector, the discfile control senses the end of the data by means of a 5-μsec integrating delay. When this delay times out, an ESP 1 pulse is generated, and then, 2 μsec later, an ESP 2 pulse is generated. The ESP 1 pulse advances the octoflop from SCS state to SCE state. The ESP 2 pulse then performs one of two

alternate functions. If there is no command code present in CM0-1 (NOP), no further operations are required, and the ESP 2 pulse returns the octoflop to IDS state. However, if a command code is present (~NOP), additional sectors of data are to be processed. The ESP 2 pulse then sends another READ or WRITE pulse to the discfile, and simultaneously returns the octoflop from SCE state to CMS state.

(22) COH - Counter Overflow Hold Flip-flop (-AD:C6)

The state of the COH flip-flop enables the discfile control to distinguish between parity bits and other address or data bits. The COH flip-flop contains 1 only when an address parity bit (the 21st bit of each address) or a data word parity bit (the 37th bit of each data word) is being (or is about to be) processed; during the processing of all other address or data bits, the COH flip-flop remains in the 0 state.

During the write command, the state of the COH flip-flop determines what output data is returned to the discfile on the even and odd data lines. (See EDO, EDZ, ODO and ODZ below.) When COH(0) is asserted, the WCE and WCO pulses sample the output data bit in DA19, but when COH(1) is asserted, the WCE and WCO pulses transmit a parity bit determined by the state of the PAR flip-flop. (The parity bit transmitted is the complement of PAR at the single WCE or WCO that coincides with the COH(1) level.)

During the read command, every input bit read from the discfile (parity as well as data) produces an RDO or an RDZ pulse. So long as

COH(0) is asserted, indicating that the current RDO or RDZ pulse represents an input data bit (not an input parity bit), the pulse is gated through to load the data bit into DA2. However, for the single RDO or RDZ pulse of each data word that represents the 37th bit of the word (the parity bit), the COH(1) level is asserted; the negation of COH(0) prevents the parity bit from entering the accumulator. The incoming parity bits from the discfile are filtered out in this way because the parity test is completed within the data control. Once the test is made there is no further need to preserve the parity bits.

The read compare command compares one or more sectors of output data (or even some fraction of a full sector) with corresponding input data read from the discfile. The comparison is executed one bit at a time, but does not include parity bits. No output parity bits are generated during read compare commands, and, although a normal parity test is made, the incoming parity bits are ignored for purposes of the read comparison.

During the read compare command, as during the read command, every input bit read from the discfile (parity as well as data) produces an RDO or an RDZ pulse. So long as COH(0) is asserted, indicating that the current pulse represents an input data bit rather than an input parity bit, each RDO pulse complements the RCT flip-flop. (See RCT below.) However, for the single RDO or RDZ pulse of each data word that represents the 37th bit of the word (the parity bit), the COH(1) level is asserted. The negation of COH(0) prevents the parity bit from complementing RCT even if the parity is a 1 and therefore initiates an RDO.

For address transmission, and for all data operations (read, write, and read compare), the bit-shift counter counts all address or data bits that are processed, but not parity bits. Consequently, the BSC is incremented at each DCK 1 pulse so long as COH(0) is asserted. When the COH flip-flop is set, indicating that an address parity bit, or a data-word parity bit is the bit that has initiated the current DCK 1 pulse, the counter is not incremented. (See BSC0-4 above.)

Similarly, for both address transmission and data operations, the data accumulator DA0-19 is shifted right one bit position after each address or data bit is processed, but is not shifted for parity bits. The data accumulator is therefore shifted right at each DCK 2 pulse so long as COH(0) is asserted; it is not shifted right at DCK 2 if COH(1) is asserted.

Besides inhibiting those functions which must be prevented during the processing of the address or data word parity bits, the COH flip-flop also enables certain other operations which must occur during the processing of the address or data word parity bits.

During address transmission, the DCK 2 pulse corresponding to the 21st bit of the address (the address parity bit) initiates the address termination pulse ATP. The DCK 2 pulse is enabled to produce an ATP only when SNA(1) and COH(1) are asserted. (The DCK 2 does not produce the ATP directly, but rather triggers a 15-μsec delay. If the SNA(1) level continues throughout the delay, as it does except in the uncommon case of an intervening IOB reset, the termination of the delay produces the ATP pulse.)

The address clock is turned on and off by a two-stage synchronizing network comprising the ACE and ACS flip-flops. The COH(1) level enables DCK 2 to reset ACE and ACS, thus turning off the address clock. (See ACS above.) The COH(1) level is asserted at only a single DCK 2 pulse, at that DCK 2 that is initiated by the 21st and final bit of the address (the address parity bit).

For read and read compare commands, the parity error flip-flop PER can only be set by the DCK 2 pulse initiated by the 37th bit of the input data word (the parity bit). The COH(1) level is used as a necessary condition for the setting of the PER flip-flop.

Set by:

DCK 2 $\wedge$ WDC(1) $\wedge$ BSC0(1) $\wedge$ BSC2(1)

Reset by:

FCL $\vee$:

DCK 2 $\wedge$ COH(1)

Although the COH flip-flop set and reset conditions are somewhat complex, the end result of these conditions is not complicated. The COH flip-flop is always set to the 1 state at the DCK 2 pulse of the address or data bit immediately preceding the parity bit, and is always reset to the 0 state at the DCK 2 pulse that is actually initiated by the parity bit.

The FCL pulse provides the initial clear; it clears the COH before each new address is transferred out and at the beginning of each new data sector.

Throughout the address transmission, the WDC flip-flop contains 1, and the WDC(1) level is asserted. The DCK 1 pulse corresponding to the 20th bit of the address being sent to the discfile (the bit immediately preceding the address parity bit) leaves both BSC0 and BSC2 in the 1 state. (See BSC0-4 above.) The DCK 2 pulse that follows about 400 nsec later thus sets the COH flip-flop to the 1 state. The next DCK 2, the DCK 2 that is initiated by the address parity bit itself, resets COH.

During all data operations (read, write and read compare commands), the BSC0 and BSC2 bits of the bit shift counter both contain 1 (the counter then contains 20) on two separate occasions during the processing of each 37-bit data word. The BSC first reaches the count of 20 at the DCK 1 pulse corresponding to the final data bit of the first 18-bit data character. The BSC is counted to 20 a second time at the final bit of the second 18-bit data character of the 27-bit data word. However, the WDC flip-flop contains 1 only during the processing of the second data character. Consequently, only the DCK 2 pulse initiated by the final bit of that second 18-bit character (the 36th and final data bit of the data word) is allowed to set the COH flip-flop to 1. The next DCK 2 pulse, the DCK 2 initiated by the 37th and final bit of the data word (the data parity bit itself) resets COH.

(23) CSP - Command Select Pulse (-FC:C4)

At the expiration of the 15-μsec delay started by ALP, the CSP pulse advances the file-control octoflop from ALS state to CMS state.

Depending on the command code stored in CM0-1, the CSP pulse causes the transmission of either a READ pulse or a WRITE pulse to the discfile. If CM0-1 contains 10 (write), then CSP transmits a WRITE pulse to the discfile. However, if CM1 contains a 1, the command code must be either 01 (read) or 11 (read compare). In either case, CSP transmits a READ pulse to the discfile.

Again depending on the contents of CM0-1, the CSP pulse may also initiate an SND pulse and a DTC pulse. The CSP pulse produces both SND and DTC whenever the command code in CM0-1 specifies either of the two output commands, write or read compare; i.e. whenever CM0 contains 1. The SND pulse gates an 18-bit output data word from the Data Control 136 into the data accumulator of the discfile control. The DTC pulse sends the Data Control 136 a Give a Character pulse to shift the next output character into position for transfer into the discfile control.

Generated by:

ALP plus 15 μsec delay.

Summary of Pulses Initiated by CSP

Command			To Discfile		
Code	CM0	CM1	WRITE	READ	SND & DTC
NOP	0	0	(Neither ALP nor CSP generated)		
Read	0	1		X	
Write	1	0	X		X
Read Compare	1	1		X	X

(24) DA RQ - Data Accumulator Request (-CM:C1)

This level is applied to the discfile control from the Data Control 136. During the output commands, write and read compare, the assertion of the DA RQ level indicates that the data control has sent out the last of its output data.

At the discfile control, the DA RQ level is used to detect the end of the output data and to terminate the output command. The CM0(1) command-pair function is asserted whenever the command register specifies either a write or a read-compare command. (See CM0-1 above.)

The assertion of CM0(1) ANDed with the assertion of DA RQ enables the ESP 1 pulse to clear the command register CM0-1 at the end of the current sector. The ESP 2 pulse that occurs 2 μsec after ESP 1 then sends the discfile an END pulse and resets the discfile control to IDS state.

Generated by:

Status Bit DA RQ (in Data Control 136) contains 1.

During output operations, the Data Control 136 sets the DA RQ status bit (and thus asserts the DA RQ level) when the data control data accumulator becomes empty.

(25) DCE - Data Clock Error Flip-flop (-SC:A5)

The CONI 270 samples the contents of DCE on IOB26.

Set by:

ERP $\wedge$ SCS(1)

Reset by:

SCL

The ERROR pulse is transmitted from the discfile through a twisted pair to the discfile control; there it is inverted and redesignated ERP. At the discfile control, the ERP pulse always sets the FER flip-flop. When the ERROR pulse arrives during the SCS state of the file-control octoflop, the DCE flip-flop is also set. This not only tells the programmer that an error has been detected, but furthermore, when the error appeared, and therefore what general class of error has occurred.

Although the fact that DCE has been set somewhat narrows the possible range of errors, there are still at least five different types of data clock error that can set DCE. Indicator lights at the discfile often may further identify the specific type of data clock error that has caused the ERROR signal. Five such types of data clock error are tabulated below.

Data Clock Error Tabulation

Error Type	Discfile Indicator Light	Error Description
Write Clock Error	WC	Missing pulse from the write clock control track (Write operations only).
Data-In Error	DI	Two successive output data pulses separated by more than 150% of the normal time interval. (Note that the data-in terminology is only correct with respect to the discfile. The data pulses referred to are output pulses from the discfile control; the DI error can occur only during writing.)
Data-Out Error	DO	Two successive input data pulses separated by more than 150% of the normal time interval. (Note that the data-out terminology is only correct with respect to the discfile. The data pulses referred to are input pulses to the discfile control; the DO error can occur only during reading and read comparing.)
Clock Check Error	CC	1) Odd or even output data pulse dropped during write operation; or 2) Even outer-zone clock generated during during inner-zone write operation; or 3) Write amplifier malfunction.
Data Check Error	DC	1) Discrepancy detected in bit-by-bit comparison between data transmitted by discfile logic unit, and data content of the discfile write current (write operations only); or 2) Processor attempts to write on locked out disc. Note that when this condition arises, the discfile sends the discfile control a WRITE LOCKOUT WARNING before the ERROR ~~pulse~~ SIGNAL pulse. (See WLE below.)

(26) DCK - Data Clock (-TR:B, C7-8)

The data clock chain includes a sequence of five separate clock pulses: DCK 1, DCK 0, DCK 2, DCK 2.1 and DCK 3. With the exception of DCK 0, which is used only for proper loading within the clock chain itself, all of these clock pulses initiate logical functions throughout the discfile control. These functions are summarized below in tabular form (at the end of the present glossary entry). More detailed descriptions of most of the operations initiated by the DCK clocks can be found under other glossary entries corresponding to the abbreviations in the second ("Clocks Operation") column of the table below.

Whenever the DCK 1 pulse is triggered, the remaining four DCK pulses always follow with fixed delays in the order specified. The DCK 1 pulse is applied to two series-connected inverters and a W607 pulse amplifier B22; the output of this pulse amplifier follows approximately 20 nsec after DCK 1, and is designated DCK 0. The DCK 0 pulse is an auxiliary pulse used only in the clock chain and nowhere else in the discfile control. The DCK 0 pulse serves only one function; that function is to trigger the W300 clock-chain generator that produces the DCK 2 pulse, the DCK 2.1 pulse, and the DCK 3 pulse. Between these three pulses and the DCK 0 pulse that initiates them, there are delays of 400, 550, and 800 nsec respectively; see figure -TR:C8.

Because the DCK 1 pulse always initiates the remaining four pulses of the clock chain, the generating conditions described below refer only to DCK 1; the remaining four pulses always follow DCK 1 in the sequence

described above. The DCK 1 pulse is generated in three different modes: 1) address transmission; 2) write commands; and 3) read or read compare commands.

Address Transmission Mode: DCK 1 Generated by:

WCE $\wedge$ ~RDS $\wedge$ WCT7(0)

Write Command Mode: DCK 1 Generated by:

WCE $\vee$ WCO. $\wedge$:

~RDS $\wedge$ WCT7(0)

Read or Read Compare Command Mode: DCK 1 Generated by:

EDO $\vee$ EDZ $\vee$ ODO $\vee$ ODZ. $\wedge$:

RDS

During address transmission, the WCT7 flip-flop is always in the 0 state, and, because SCS(1) is not asserted, ~RDS is asserted. (See RDS below.) Consequently, each WCE pulse initiates a DCK 1 pulse. Note that during address transmission, the WCE pulses are not generated by one of the write clock control tracks in the discfile, but instead are generated by the internal address clock of the discfile control itself. (See WCE below.)

For write commands, the DCK 1 pulse is initiated by each WRITE CLOCK EVEN and WRITE CLOCK ODD pulse generated by the appropriate write clock control track in the discfile (inner-zone or outer-zone depending upon the location of the sector to be written; see WRITE CLOCK EVEN below.) The WRITE CLOCK pulses are inverted within the discfile control, and redesignated WCE and WCO. Even and odd WRITE CLOCK pulses from the discfile continue to trigger DCK 1 until the word count reaches 128 and the WCT7 flip-flop is set to 1 (at the end of the sector). As soon as WCT7(0) is no longer asserted, the DCK clock is turned off; no further DCK 1 pulses can be

generated. The ~RDS level is asserted throughout all write commands, because CM1(1) is then negated. (See RDS below.)

Note that every write command must continue to the end of the 128-word sector, even if the output data from the discfile is exhausted prior to that time. Incoming WCE and WCO clocks continue to arrive at the discfile control, and the DCK clock chain continues to function. If no further output data is available, 0's are written for the rest of the 128-word sector. Similar considerations apply to the read compare command. It is only the read command that can be terminated in mid-sector by setting the END flip-flop.

For the read and read compare commands, the DCK 1 is initiated by each incoming data pulse (EDO, EDZ, ODO, and ODZ) from the discfile. The RDS level is always asserted until the WCT7 flip-flop is set to 1. The WCT7 flip-flop is set to 1 only when the word count reaches 128, or (in the case of the read command only) at the first DCK 1 pulse after the END flip-flop is set to 1.

NOTE TO EDITOR: If it wouldn't introduce too many production problems, it would be useful to treat this entire two-page table as a B-sized tip-out page.

OPERATIONS INITIATED BY DCK CLOCK PULSES

Pulse	Clocks Operation	When Enabled	Remarks
DCK 1	DCK 0	20-nsec delay	DCK 0 always follows DCK 1.
	BSC count	COH(0)	BSC is incremented at every DCK 1 except those corresponding to parity bits.
	Take a Character	CM0(0) ∧ SCS(1) ∧ END(0) ∧ BSC0(1) ∧ BSC3(1) ∧ BSC4(1)	Read command, Take a Character initiated by DCK 1 corresponding to 18th data bit of the input character.
	Set WCT7	CM0(0) ∧ SCS(1) ∧ END(1)	If END set during read command, the first DCK 1 ends RDS and stops the clock.
DCK 0	DCK 2	400-nsec delay	DCK 2 always follows DCK 0
	DCK 2.1	550-nsec delay	DCK 2.1 always follows DCK 0
	DCK 3	800-nsec delay	DCK 3 always follows DCK 0

DCK 2	SDR	COH(0)	DA0-19 shifted right for all address and data bits except parity bits.
	ACE, ACS reset	COH(1)	Address clock turned off after sending address parity bit.
	WDC complemented	BCO	WDC in the 1 state for address transmission and for the 2nd data character of each data word.
	COH set	WDC(1) ∧ BSC0(1) ∧ BSC2(1)	COH is set by the DCK 2 of the address or data bit immediately preceding the address or data parity bit.
	COH reset	COH(1)	COH is reset again by the DCK 2 of the address or data parity bit.
	WCT count	WDC(1) ∧ BCO	WCT is incremented at the 36th bit of each data word.
	PER set	CM1(1) ∧ COH(1) ∧ PAR(0)	For read and read compare, parity bit DCK 2 pulse tests parity by sampling PAR.
	BSC3 set	SCS(1) ∧ BSC0(1) ∧ BSC2(1)	BSC set to 2-state in preparation for next 18-bit data character.
DCK 2.1	SND, DTC, and Give a Character	CM0(1) ∧ SCS(1) ∧ BCO	For output commands, a new 18-bit data character is transferred into DA2-19 150 nsec after the last SDR of the previous data character.
DCK 3	RCE set	RDC ∧ RCT(1)	RCE set when read comparison fails.
	RCT reset	---	RCT reset after each read comparison.
	BSC0, BSC2 reset	BCO	Bit counter decremented by 20 in preparation for next count.

(27) DFR - Disc File Ready (-FC:B4)

Fourth state of the file-control octoflop. Each sector of data begins with a 21-bit header. The address of the sector makes up the first 19 bits of these 21. The read-next-sector and parity bits are not included in the header. In place of these two bits, immediately after the 19 address bits, two 0 bits are written into the header. When a header that matches the last address sent to the discfile is sensed, the discfile sends a RDY pulse to the discfile control. The RDY pulse indicates that the discfile has sensed the "get-ready" sector (the sector with the matching header) and is ready to accept a command with respect to the next sector (the sector that immediately follows the "get-ready" sector). The RDY pulse advances the octoflop from ADT state to DFR.

The DFR(1) level is applied to the logic nets that govern octoflop entry to ADT state and ALS state. The octoflop remains in DFR state for one millisecond. This one-millisecond window gives the programmer time to set up the command buffer and thereby to specify what operations are to be performed during the forthcoming sector. After remaining in DFR state for one millisecond, the octoflop either advances to ALS state (if ~NOP is asserted, indicating that there is a command code in bits CM0-1 of the command buffer), or else the octoflop returns to ADT state (if NOP is asserted, indicating that no command code is stored in CM0-1).

When the read-next-sector bit is 0, the octoflop is held in DFR state during only one millisecond of each 52-millisecond disc revolution. However, when the read-next-sector bit is 1, the octoflop is set to DFR for one millisecond after the header from each sector of the addressed track is sensed.

Provided that the priority interrupt enable switch EFR (a flip-flop switch) is set to 1, DFR(1) produces a PIE level. The resulting priority interrupt request often is used to notify the programmer that the discfile has found the required address and that the discfile control may need a command.

The CONI 270 samples DFR on IOB21.

Generated by:

RDY $\wedge$ ADE(0) $\wedge$ ADT(1)

The octoflop can enter DFR state only from ADT state. If ADE(0) is asserted, indicating that no address errors are present, the RDY pulse advances the octoflop from ADT to DFR state. Note that ADE(0) must also be asserted in order to start the 1.0-millisecond delay that controls the duration of DFR state. When no address errors are present, the termination of this delay causes the octoflop to advance from DFR to ALS state, or to return from DFR to ADT state.

(28) DRL - Data Request Late Flip-flop (-SC:A6)

The DRL flip-flop is set if the Data Control 136 is not selected to the discfile control in time to operate upon a forthcoming data sector. Provided that the priority interrupt switch EFE (a flip-flop switch) is set to 1, DRL(1) produces a PIE level. The resulting priority interrupt request may be used to notify the programmer that the intended operation cannot be performed until an appropriate CONO command is sent to the data control. This CONO DATA CONTROL must set up the control-

status register bits IC30-32 to internal device address 5, and thus select the Data Control 136 to the discfile control.

Set by:

NDP 2 ^ SEL 5

Reset by:

SCL

If the Data Control 136 has not been selected to the discfile control by NDP 2 time (a number of milliseconds before the data sector that is to be processed begins), the DRL flip-flop is set. This provides ample time to send the processor a priority interrupt, and for the processor to respond with the requisite CONO DATA CONTROL.

(29) DTC - Data to Control (270) (AD:D2)

The DTC pulse is used only for the purpose of generating two additional pulses, SND, and GIVE A CHARACTER. The DTC pulse is inverted to produce the SND pulse. To produce the GIVE A CHARACTER pulse, the DTC pulse is amplified by pulse amplifier D26.

The SND pulse causes the contents of the 18 high order bits of the Data Control 136 data accumulator to be loaded into bits DA2-19 of the discfile control data accumulator.

The GIVE A CHARACTER pulse is sent to the Data Control 136 accumulator shift logic. The GIVE A CHARACTER pulse signals the data control that the discfile control has accepted the character currently stored in the high order end of the data control data accumulator (by

clocking it in with an SND pulse), and that unless that character was the last (second) character in the output data word, the data control must shift the contents of its accumulator left by 18 bit-positions in order to bring the next character into the high order bit-positions.

Generated by:

CSP $\wedge$ CM0(1). $\vee$:

DCK 2.1 $\wedge$ CM0(1) $\wedge$ SCS(1) $\wedge$ BCO. $\vee$:

PTC $\wedge$ IOB1(1)

The first two lines of the DTC generating conditions listed above both include the CM0(1) command-pair function. This function is asserted during the output commands, write and read compare. For most operations of the discfile control, the DTC pulse is required only for those two output commands.

The first line of the generating conditions includes the CSP pulse that precedes the beginning of each data sector; that pulse produces the first DTC pulse of the sector. That DTC in turn initiates the SND pulse that loads the first 18-bit data character of the sector into DA2-19; furthermore that DTC sends the Data Control 136 the first GIVE A CHARACTER pulse of the sector. That first GIVE A CHARACTER causes the second 18-bit data character of the sector to be shifted left 18 bit-positions in the data control data accumulator. The second character is then properly positioned for transfer out into the discfile control data accumulator.

The second line of the DTC generating conditions produces an additional

DTC pulse for each additional data character after the first one. The BCO level is asserted at the DCK 1 pulse initiated by the write clock or the input data pulse corresponding to the 18th and final data bit of the first data character of the sector. The DCK 2 pulse that follows about 400 nsec later shifts the first character right for the 18th and last time. The DCK 2.1 pulse then follows 150 nsec after the shift, and initiates the second DTC pulse of the sector. That DTC in turn initiates the SND pulse that loads the second 18-bit character of the sector into DA2-19, and furthermore sends the GIVE A CHARACTER pulse to the Data Control 136. This process continues until the end of the 128-word data sector when the DCK clock is turned off.

The third and final line of the DTC generating conditions listed above represents a programmed DTC that can be given at will from the processor. This DTC is generated at the IOB CONO SET time of any CONO 270 instruction that has a 1 in bit 19. (The IOB1 input corresponds to bit 19 of the CONO instruction word.)

(30) DTP - Data to Processor (-AD:A8)

This 2.5-μsec level gates the contents of the data accumulator DA0-19 and the bit shift counter BSC0-4 onto the IO bus. The processor reads in this information from the bus at the 2-μsec point. Note that for this transfer, as for the PTA transfer described below, the DA12 and DA19 bits are connected out of sequence. (This is done to match the format of the discfile address.)

Generated by:

IOB DATA I ∧ SEL

The processor DATAI instruction produces a 2.5-μsec negative IOB DATA I level. This command level is effective at the discfile control only when SEL is asserted, i.e. only when the DATAI instruction device selection code is 270 octal.

(31) EDO - Even Data Ones (-TR:C3)

Addresses and data are transmitted to and from the discfile on four twisted-pair transmission lines. These four twisted-pair lines are used in sets of two; one line of each set is pulsed for a 1; the other line for a 0. Two such sets of two lines are used. This allows alternate bits (odd and even) to be transmitted on different lines, thus halving the data transmission frequency.

For address transmission, and for write commands, each EDO pulse is amplified by pulse amplifier D31 and then transmitted to the discfile over the even data 1's twisted pair.

During read and read compare commands, if RDS is asserted (indicating that the word counter has not yet reached its final count of 128), each EDO pulse initiates an RDO pulse and starts the DCK clock chain. The RDO pulse in turn is used to complement RCT, and (for read commands only) to set DA2.

Generated by:

Even data 1 (read at discfile) . $\vee$:

WCE $\wedge$:

COH(0) $\wedge$ DA19(1) . $\vee$:

COH(1) $\wedge$ PAR(0)

When reading, or read comparing, the EDO pulses enter the discfile control on the even data 1's line from the discfile. The two series-connected inverters through which the EDO pulses are passed before generating RDO are used for noise isolation and for adjusting the voltage reference to the usual DEC logic levels (-3 vdc to ground).

During writing, the EDO pulses are clocked by the write clock even, WCE. So long as address bits or data bits are being sent to the discfile, COH(0) is asserted; COH(1) is asserted only when the parity bit is sent to the discfile. Because odd parity is generated for both address and data, a 1 parity bit is sent to the discfile when the total number of 1 bits in the address, or in the data word, is even; i.e. when PAR (0) is asserted.

(32) EDZ - Even Data Zeros (-TR:C4)

See EDO above.

For address transmission and for write commands, each EDZ pulse is amplified by pulse amplifier D31 and then transmitted to the discfile over the even data 0's twisted pair.

During read and read compare commands, if RDS is asserted, each EDZ pulse initiates an RDZ pulse and starts the DCK clock chain. The RDZ pulse resets DA2, but only during read commands.

Generated by:

Even data 0 (read at discfile) . ∨:

WCE ∧:

COH(0) ∧ DA19(0) . ∨:

COH(1) ∧ PAR(1)

So long as address bits or data bits are being sent to the discfile, COH(0) is asserted; COH(1) is asserted only when the parity bit is sent to the discfile. Because odd parity is generated for both address and data, a 0 parity bit is sent to the discfile when the total number of 1 bits in the address or data word is odd as it stands (without the parity bit); i. e. when PAR(1) is asserted.

(33) EES - Enable End of Sector Flip-flop (-CM:B5)

The EES flip-flop enables or disables the sector end flag priority interrupt condition. If the EES flip-flop contains 1, a priority interrupt enable level PIE is produced whenever the sector end flag SEF is set to 1. However, if the EES flip-flop contains 0, the SEF(1) level is not allowed to initiate PIE.

Set by:

PTC ∧ IOB12(1)

Reset by:

CCB

The EES flip-flop may be set by programming a CONO 270 instruction with a 1 in bit 30. To reset EES, give a CONO 270 with 0 in bit 30. (The IOB12 input corresponds to bit 30 of the CONO instruction word.)

(34) EFE - Enable File Error Flip-flop (-CM:B4)

The EFE flip-flop enables or disables the following four priority interrupt conditions:

1) File error (see FER below);

2) Parity error (see PER below);

3) Read compare error (see RCE below);

4) Data request late (see DRL above).

If the EFE flip-flop contains 1, a priority interrupt enable level PIE is produced whenever any of the four priority interrupt conditions, FER(1), PER(1), RCE(1), or DRL(1) is asserted. However, if the EFE flip-flop contains 0, these conditions are not allowed to initiate PIE.

Set by:

PTC $\wedge$ IOB11(1)

Reset by:

CCB

The EFE flip-flop may be set by programming a CONO 270 instruction with a 1 in bit 29. To reset EFE, give a CONO 270 with 0 in bit 19. (The IOB11 input corresponds to bit 29 of the CONO instruction word.)

(35) EFR - Enable File Ready (-CM:B6)

The EFR flip-flop enables or disables the discfile ready priority interrupt condition. If the EFR flip-flop contains 1, a priority interrupt level PIE is produced whenever the file status octoflop is stepped to the discfile ready state DFR. However, if the EFR flip-flop contains 0, the DFR(1) level is not allowed to initiate PIE.

Set by:

PTC ∧ IOB13(1)

Reset by:

CCB

The EFR flip-flop may be set by programming a CONO 270 instruction with a 1 in bit 31. To reset EFR, give a CONO 270 with 0 in bit 31. (The IOB13 input corresponds to bit 31 of the CONO instruction word.)

(36) EIS - Enable Idle State Flip-flop (-CM:B6)

The EIS flip-flop enables or disables the idle state priority interrupt condition. If the EIS flip-flop contains 1, a priority interrupt level PIE is produced whenever the file status octoflop is reset to the idle state IDS. However, if the EIS flip-flop contains 0, the IDS(1) level is not allowed to initiate PIE.

Set by:

PTC ∧ IOB14(1)

Reset by:

CCB

The EIS flip-flop may be set by programming a CONO 270 instruction with a 1 in bit 32. To reset EIS, give a CONO 270 with 0 in bit 32. (The IOB14 input corresponds to bit 32 of the CONO instruction word.)

(37) END (-TR:A5)

The discfile control sends an END pulse to the discfile (through a twisted-pair transmission line) as a signal that the current command is finished and that communication may be discontinued. The END pulse disconnects the discfile from the discfile control. After the END pulse has been received, the discfile is ready to receive a new SELECT (sent to the discfile at the PTA pulse of the next DATAO 270 address command).

After receiving the END pulse, the discfile transmits no more than one additional write clock pulse or data bit. Regardless of when the END flip-flop is set, the END pulse is always sent to the discfile at ESP 2 time. If the discfile does not receive either an END, a READ, or a WRITE within an 8-μsec guard slot following the last data pulse of the record, an end error is generated (usually causing the CME flip-flop to be set at the discfile control).

Generated by:

ESP 2 ∧ NOP

If the END flip-flop is set, the ESP 1 pulse clears the command register CM0-1. Consequently, the NOP level is asserted when the ESP 2 pulse arrives 2 μsec later, and an END pulse is then applied to the discfile.

Note that when the CLR flip-flop is set, the END flip-flop is also set automatically (although the converse is not true). As a result, a CLEAR pulse will never be sent to the discfile unless an END pulse is

sent at the same time. An END pulse may, however, be sent to the discfile without a CLEAR pulse.

(38) END Flip-flop (-CM:B4)

When the END flip-flop is set the ESP 1 pulse that marks the end of the current sector clears the command register CM0-1. The ESP 2 pulse, which follows ESP 1 two μsec later, then sends an END pulse to the discfile. The END pulse terminates the current operation. (See END Pulse above.)

Generated by:

PTC ∧ IOB5(1)

The END flip-flop may be set by programming a CONO 270 instruction with a 1 in bit 23. To reset END, give a CONO 270 with 0 in bit 23. (The IOB5 input corresponds to bit 23 of the CONO instruction word.)

(39) ERP - Error Pulse (-SC:B5)

An ERP is produced whenever the discfile detects an error condition. The ERP always sets the FER flip-flop. Provided that the priority interrupt enable switch EFE (a flip-flop switch) is set to 1, the FER(1) produces a PIE level. The resulting priority interrupt may be used to notify the programmer that an error has occurred.

When an ERP occurs during the ADT state of the file-control octoflop, it sets the ADE flip-flop as well as the FER; when it occurs during CMS state, it sets the CME flip-flop; and when it occurs during SCS state, it sets the DCE flip-flop. This arrangement not only tells the programmer that an error has been detected, but furthermore when the error has appeared, and therefore what general class of error has occurred. Indicators at the discfile itself can further categorize the error. For a listing and description of the various types of error that the discfile can detect, refer to ADE, CME, and DCE above.

Generated by:

ERROR SIGNAL (from discfile)

The ERROR SIGNAL is transmitted from the discfile to the discfile control through a twisted pair transmission line.

(40) ERROR SIGNAL (-SC:B1)

The discfile sends an ERROR SIGNAL to the discfile control through a twisted-pair transmission line to indicate that a ~~data-or-sequence~~ discfile error has been detected. At the discfile control the ERROR SIGNAL is inverted and redesignated ERP. (See ERP above.)

(41) ESP 1 - End of Sector Pulse 1 (-FC:C6)

The ESP 1 pulse, and the ESP 2 pulse which follows it 2 μsec later, mark the end of each data sector. The ESP 1 pulse advances the file-control octoflop from SCS state to SCE state. The ESP 1 pulse may also clear CM0-1. This clear function is performed if no further data sectors are to be processed. During output commands (write or read compare), ESP 1 will clear CM0-1 only when the Data Control 136 applies a DA RQ signal to the discfile control. (The DA RQ signal indicates that the data control data accumulator is empty and that no further data is to be sent to the discfile control.) During all commands, output or input, the ESP 1 pulse will clear CM0-1 whenever the END(1) level is asserted.

Generated by:

DCK 1 ∧ SCS(1) plus 5-μsec delay

Throughout SCS state, the SCS(1) level enables each DCK 1 clock pulse to restart integrating delay AB01. This delay permits the discfile control to sense the end of each sector of data. When no DCK 1 pulses have been applied to the delay for 5 μsec, the delay times out, producing the ESP 1 pulse that signals the end of the sector. The ESP 2 pulse is always generated 2 μsec after ESP 1.

(42) ESP 2 - End of Sector Pulse 2 (-FC:C8)

The ESP 2 pulse always occurs 2 μsec after the ESP 1 pulse that marks the end of each data sector. The ESP 1 pulse clears out command code bits CM0-1 if and only if no further data sectors are to be processed. If, however, the present operation is to be continued throughout additional sectors of data, the contents of CM0-1 are left intact.

The effect of ESP 2 depends upon whether or not ESP 1 has cleared out CM0-1. If ESP 1 has not cleared CM0-1 (~NOP), then ESP 2 returns the file-control octoflop from SCE state to CMS state and simultaneously sends an appropriate command pulse (READ or WRITE as the case may be) to the discfile. On the other hand, if CM0-1 are cleared at ESP 2 time (NOP), then ESP 2 resets the octoflop from SCE state to IDS state and sends an END pulse to the discfile. If the CLR(1) level is asserted, ESP 2 also applies a CLEAR signal to the discfile.

The ESP 2 pulse sets the SEF flag whenever SCE(1) is asserted. The SCE state of the octoflop lasts for only the 2-μsec interval between ESP 1 and ESP 2, but the SEF flag saves SCE for sampling. The SEF remains set until being reset by an SCL pulse. The programmer can reset the SEF flag at will by generating a programmed SCL pulse (by means of a CONO 270 instruction with a 1 in bit 28).

Generated by:

ESP 1 plus 2-μsec delay

(43) FCL - File Clear (-AD:C5)

The FCL pulse clears the bit-shift counter BSC0-4, the word counter WCT0-7, and the following six control flip-flops: ACE, ACS, COH, PAR, RCT, and WDC.

Generated by:

IBR ∨ PTA ∨ NDP 1

An FCL pulse is generated by IBR when computer power is first turned on, when the IO reset key is operated, and when a programmed reset is executed. This ensures that the discfile control begins operations in the correct initial state. Whenever a DATAO 270 instruction is used to send a new address to the discfile control, a PTA pulse is produced and the PTA in turn produces an FCL pulse to clear the discfile control. An additional FCL is produced at the NDP 1 pulse that signals the beginning of each sector to be processed.

(44) FER - File Error (-SC:A2)

Provided that the priority interrupt enable switch EFE (a flip-flop switch) is set to 1, the FER(1) produces a PIE level. The resulting priority interrupt may be used to notify the programmer that a file error has occurred. The CONI 270 samples the contents of FER on IOB 34.

Set by:

ERP ∨ ALM ∨ OPR ∨ WRITE LOCKOUT WARNING (from discfile)

Reset by:

SCL

(45) GIVE A CHARACTER (-SC:C7)

The GIVE A CHARACTER pulse is sent to the Data Control 136 accumulator shift logic. The GIVE A CHARACTER pulse signals the data control that the discfile control has accepted the 18-bit character currently stored in the high order end of the data control data accumulator (by clocking it in with an SND pulse), and that unless that character was the last (second) character in the output data word, the data control must shift the contents of its accumulator left by 18 bit-positions in order to bring the next character into the high order bit-positions. If the character just accepted by the discfile control was the last character of the output data word, the GIVE A CHARACTER pulse causes the data control to advance a new 36-bit output data word from its data buffer DB into its data accumulator DA.

Generated by:

DTC

(46) IBR - In-Out Bus Reset (-SC:C1)

At the discfile control, IOB RESET is inverted, and the resulting positive pulse is designated IBR. The IBR pulse itself directly clears the two command code bits CM0 and CM1. Furthermore, IBR produces four additional clear pulses: CCB, CDA, SCL, and FCL. These four pulses clear the command buffer and data accumulator, and reset all status and control flip-flops to the 0 state.

Generated by:

IOB RESET

The IOB RESET pulse (and the resulting IBR pulse) are generated when computer power goes on or when the operator presses the IO reset key on the computer console. The program may also generate a reset.

(47) IDS - Idle State (-FC:B2)

Initial state of the file-control octoflop. The DATAO 270 instruction can transfer a new address into the data accumulator only when the octoflop is in either IDS state or ADT state. Neither the CDA pulse nor the PTA pulse can be generated unless the octoflop is in one of these two states. For the octoflop to be stepped to the SNA state, it must also be in either IDS or ADT state.

Provided that the priority interrupt enable switch EIS (a flip-flop switch) is set to 1, IDS(1) produces a PIE level. The resulting priority interrupt request often is used to notify the programmer that the discfile control has completed the preceding operation, and is ready to receive a new data address. The CONI 270 samples IDS on IOB18.

Generated by:

IBR $\vee$.

MCL(1) $\vee$.

ESP 2 $\wedge$ NOP

The IBR pulse resets the file-control octoflop to IDS state when computer power is first turned on, when the IO reset key is operated, and

when a programmed reset is executed. This ensures that the discfile control begins operations with the octoflop in IDS state. The octoflop is also reset to IDS state when the master clear button is operated.

At the end of each sector of data the ESP 2 pulse causes the octoflop to return from SCE state to either IDS state or to CMS state. The octoflop is reset from SCE to IDS whenever command code bits CM0 and CM1 are both in the 0 state (specifying NOP). If CM0 and CM1 are not both in the 0 state (~NOP), then instead of returning from SCE to IDS state, the octoflop returns from SCE to CMS state.

(48) IOB CONO CLR - In-Out Bus Conditions Out Clear (-CM:C3)

When gated by the SEL level, the IOB CONO CLR pulse produces a CCB pulse which clears the entire command buffer if CCM is asserted (and which otherwise clears the nine low-order bits of the command buffer, but leaves CM0 and CM1 unchanged).

The processor CONO instruction produces an IOB CONO CLR pulse followed one microsecond later by an IOB CONO SET pulse. These two command pulses are effective only at that IO device or control unit specified by the CONO instruction device selection code.

(49) IOB CONO SET - In-Out Bus Conditions Out Set (-CM:D3)

When gated by the SEL level, the IOB CONO SET pulse produces a PTC pulse which gates new control information into the command buffer. (A new command code is gated into CM0 and CM1 only if CCM is asserted. Otherwise the contents of these two bits are left unchanged.)

The processor CONO instruction produces an IOB CONO SET one microsecond after IOB CONO CLR.

(50) IOB DATA CLR - In-Out Bus Data Clear (-AD:B1)

The first processor signal to reach the discfile control during the DATAO instruction is IOB DATA CLR. When gated by appropriate conditions, the IOB DATA CLR pulse produces a CDA pulse that clears the data accumulator DA0-19.

The processor DATAO instruction produces an IOB DATA CLR pulse followed one microsecond later by an IOB DATA SET pulse. These two command pulses are effective at the discfile control only when the DATAO instruction device selection code is 270 octal. (This causes the SEL level to be asserted within the discfile control.)

(51) IOB DATA I - In-Out Bus Data In (-AD:A1)

The IOB DATA I level is inverted to produce the DTP level. During the 2.5 μsec while DTP is present, the contents of the data accumulator and the bit shift counter are placed on the IO bus. The processor reads this information in from the bus at the 2-μsec point.

The processor DATAI instruction produces a negative 2.5-μsec IOB DATA I level. This command level is effective at the discfile control only when the DATAI instruction device selection code is 270 octal. (This causes the SEL level to be asserted within the discfile control.)

(52) IOB DATA SET - In-Out Bus Data Set (-AD:B1)

When gated by appropriate conditions, the IOB DATA SET pulse produces a PTA pulse which loads a discfile address from IOB17-35 into the data accumulator of the discfile control.

The processor DATAO instruction produces an IOB DATA SET one microsecond after IOB DATA CLR.

(53) IOB RESET - In-Out Bus Reset (-SC:C1)

The IOB RESET pulse is a negative ungated clear pulse which is applied to all IO devices and control units. It clears the control registers and data buffers of all equipment attached to the in-out bus. At the disc-file control, IOB RESET is inverted, and the resulting positive pulse is designated IBR.

The IOB RESET pulse is generated when computer power goes on or when the operator presses the IO reset key on the computer console. The program may also generate a reset.

(54) IOB STATUS - In-Out Bus Status (-CM:D7)

The IOB STATUS level is gated by SEL to produce an STP level. During the 2.5 μsec while STP is present, a variety of CONI status information (including the contents of the command buffer, the file-control octoflop, the error-status flags, and the relay-meter test switches) is placed on the IO bus. The processor reads this information in from the bus at the 2-μsec point.

The processor CONI instruction produces a negative 2.5-μsec IOB STATUS level. This command level is effective at the discfile control only when the CONI instruction device selection code is 270 octal. (This causes the SEL level to be asserted within the discfile control.)

(55) IOS3-9 - In-Out Selection Lines (-SC:D1)

Two IOS lines are associated with each of the seven bits 3-9. These 14 lines represent both states of the 7 bits of the device code contained in bits 3-9 of the instruction register (assertion is at ground). Every device control contains one or more diode nets, each of which receives inputs from seven of the 14 IOS lines, one from each bit in IR3-9. The input connections that are wired in determine the selection code for the device, and its control unit then responds to IOT commands only when the appropriate number appears in the device code portion of the instruction word. The discfile control uses the device code 270. The diode net output corresponding to this code is designated SEL.

(55A) M5B - Minus Five Volt Bus (-TR:A2)

(56) MCL - Master Clear (-CM:D7)

The M5B line provides a

When pressed, the MCL pushbutton closes two normally-open switches in the discfile control. One of these switches is connected across the two ends of a twisted-pair line to the discfile. The other switch applies +10 vdc through a switch filter to a pair of series-connected inverters, thus producing the MCL level at both polarities.

↓ terminating supply for the twisted-pair transmission lines.

When the normally open-circuited twisted pair is shorted by operating the MCL pushbutton, the discfile executes a master clear function. This includes the removal of power from the previously selected head positioner, and the clearing of the discfile logic and error indicators. The discfile requires 20 ms to complete the master clear operation; throughout this period the discfile will not accept any commands.

Within the discfile control, the MCL level prevents the generation of the CDA and PTA pulses, and resets the file-control octoflop to IDS state. The CONI 270 samples the MCL level on IOB 3.

Generated by:

MCL pushbutton (at discfile control)

The MCL pushbutton is located on the front of the discfile control logic at the right end of rack A. The master clear should be given when the system is first turned on and when one programmer replaces another at the console.

(57) MRB - Meter Reading Bad (-SC:D3)

The address sequencing relays in the discfile can readily be tested with the aid of a relay test program run from the processor. The program cycles through all the sequencing relays in a predetermined order. As each relay is tested, a discfile voltmeter indicates whether or not its contact resistance is normal. If the contact resistance is excessive, a high meter reading shows that the relay is bad. The maintenance engineer performing the test should then press the Meter Reading Bad

pushbutton, and thus produce an MRB level at the discfile control. The CONI 270 samples the MRB level on IOB10.

Generated by:

MRB pushbutton (at discfile)

When pressed, the MRB pushbutton opens a normally-closed switch in the discfile. This switch is connected across the two ends of a twisted-pair line to the discfile control. In the normal quiescent state the closed switch returns +10 vdc to a switch filter in the discfile control. When the MRB pushbutton is operated, the normally-closed switch is opened, removing the +10 vdc from the switch filter and causing the MRB level to be negatively asserted.

(58) MRG - Meter Reading Good (-SC:D3)

See MRB above. If there is a low meter reading, showing normal contact resistance for the relay under test, the maintenance engineer performing the test should then press the Meter Reading Good pushbutton and thus produce an MRG level at the discfile control. The CONI 270 samples the MRG level on IOB9.

Generated by:

MRG pushbutton (at discfile)

(59) NDP 1 - New Data Pulse 1 (-FC:B6)

The NDP 1 pulse, and the NDP 2 pulse which follows it 1.0 μsec later, precede (by a number of milliseconds) the beginning of each new sector of data. The NDP 1 pulse advances the file-control octoflop from

CMS state to SCS state. A READ or WRITE command pulse is sent to the discfile at the beginning of the CMS state; CMS state is ended 200 μsec later when NDP 1 advances the octoflop to SCS state. The NDP 1 pulse also generates an FCL pulse which clears the bit-shift counter BSC0-4, the word counter WCT0-7, and the following five control flip-flops: ACE, ACS, COH, WDC, and RCT.

Generated by:

CSP ∧ ALS(1) . ∨ . ESP(2) ∧ NOP :∧:

plus 200-μsec delay ∧ CMS(1)

The NDP 1 pulse is produced by the termination of a 200-μsec delay that is started whenever the octoflop is advanced to CMS state. The NDP 2 pulse is always generated 1.0 μsec after NDP 1. The first line of the conditions listed above is identical to the conditions that step the octoflop to CMS state. The requirement that CMS(1) still be asserted at the expiration of the delay is to guard against the unlikely contingency that an IOB reset pulse might occur during the 200-μsec delay. If that should happen, it would not be desirable to advance the octoflop to SCS state. The octoflop would instead be reset to IDS state and neither the NDP 1 nor the NDP 2 pulse would be required.

(60) NDP 2 - New Data Pulse 2 (-FC:B8)

The NDP 2 pulse always occurs 1.0 μsec after the NDP 1 pulse that steps the file status octoflop to SCS state. If the Data Control 136 has not yet been selected to the discfile control at NDP 2 time, then the SEL 5

level is ground, and the data request late flip-flop DRL is set by NDP 2. Provided that the priority interrupt switch EFE (a flip-flop switch) is set to 1, DRL(1) produces a PIE level. The resulting priority interrupt request may be used to notify the programmer that the intended operation cannot be performed until he has sent the data control an appropriate CONO command.

The NDP 2 pulse also is used to initialize the bit-shift counter prior to the commencement of each data sector. An FCL pulse that clears the bit-shift counter is initiated by NDP 1. The NDP 2 pulse then sets BSC3 to the 1 state, thus leaving the counter with contents equal to 2 at the beginning of the data sector. Although the counter always overflows when its contents are 20, if BSC3 is initially set before counting is begun, only 18 counts are required to reach the overflow count of 20. This ensures that overflow occurs at the DCK 1 clock corresponding to the eighteenth and final data bit of the first 18-bit character. (For the remaining characters of the data sector, SCS(1) enables BSC3 to be set to 1 at the end of each character.)

Generated by:

NDP 1 plus 1.0-μsec delay

(61) NOP - No Operation (-CM:A2)

The NOP level, and its complement, ~NOP, control two branch points of the file-control octoflop cycle. The first of these branch points occurs at the end of the 1-ms DFR state; at that time, the octoflop either advances

to ALS state or returns to ADT state, the choice depending upon whether or not NOP is asserted. If ~NOP is asserted, indicating that a command has already been given to the discfile control, then an ALP pulse is sent to the discfile, and the octoflop is advanced to ALS state. However, if NOP is asserted, indicating that as yet no command has been received from the processor, the octoflop returns to ADT state.

The octoflop remains in ADT state until the next RDY pulse initiates another DFR state. After 1 ms, that DFR state ends, and the branch-point decision is repeated. The discfile control continues to oscillate between ADT and DFR until a command code arrives from the processor. The first such command to arrive ends NOP and enables the octoflop to advance to ALS state.

The second branch point of the octoflop cycle that is controlled by the NOP level occurs at the ESP 2 pulse marking the end of the SCE state. If NOP is then asserted, indicating that the current operation is finished, the octoflop is immediately reset to IDS, and the discfile control is ready to accept a new address (which can be sent to it by means of a DATAO 270 instruction). On the other hand, if the current operation is not yet finished, but is to continue for one or more additional sectors of data, then ~NOP is asserted and the octoflop is stepped to CMS state rather than to IDS state from SCE state. At the same time, another READ or WRITE command pulse is sent to the discfile, causing the current discfile address to be incremented.

Generated by:

CM0(0) ∧ CM1(0)

The NOP level is asserted only if both bits of the command register CM0-1 contain 0. The other three possible configurations of these two bits each correspond to one of the three discfile command codes. (See RDC, RED, and WRT below.)

(62) ODO - Odd Data Ones (-TR:C1)

Addresses and data are transmitted to and from the discfile on four twisted-pair transmission lines. These four twisted-pair lines are used in sets of two; one line of each set is pulsed for a 1, the other line for a 0. Two such sets of two lines are used. This allows alternate bits (odd and even) to be transmitted on different lines, thus halving the data transmission frequency.

For address transmission, and for write commands, each ODO pulse is amplified by pulse amplifier D30 and then transmitted to the discfile over the odd data 1's twisted pair.

During read and read compare commands, if RDS is asserted (indicating that the word counter has not yet reached its final count of 128), each ODO pulse initiates an RDO pulse (Read Data Ones) and starts the DCK clock-chain. The RDO pulse in turn is used to complement RCT, and (for read commands only) to set DA2.

Generated by:

Odd data 1 (read at discfile). $\vee$. :

WCO $\wedge$:

COH(0) $\wedge$ DA19(1) . $\vee$.

COH(1) $\wedge$ PAR(0)

When reading, or read comparing, the ODO pulses enter the discfile control on the odd data 1's line from the discfile. The two series-connected inverters through which the ODO pulses are passed before generating RDO are used for noise isolation and for adjusting the voltage reference to the usual DEC logic levels (-3 vdc to ground).

During writing, the ODO pulses are clocked by the write clock odd, WCO. So long as address bits or data bits are being sent to the discfile, COH(0) is asserted; COH(1) is asserted only when the parity bit is sent to the discfile. Because odd parity is generated for both address and data, a 1 parity bit is sent to the discfile when the total number of 1 bits in the address or data word is even; i. e. when PAR(0) is asserted.

(63) ODZ - Odd Data Zeros (-TR:C2)

See ODO above.

For address transmission and for write commands, each ODZ pulse is amplified by pulse amplifier D30 and then transmitted to the discfile over the odd data 0's twisted pair.

During read and read compare commands, if RDS is asserted, each ODZ pulse initiates an RDZ pulse, and starts the DCK clock chain. The RDZ pulse resets DA2, but only during read commands.

Generated by:

Odd data zero (read at discfile) . ∨ . :

WCO ∧ :

COH(0) ∧ DA19(0) . ∨ .

COH(1) ∧ PAR(1)

So long as address bits or data bits are being sent to the discfile, COH(0) is asserted; COH(1) is asserted only when the parity bit is sent to the discfile. Because odd parity is generated for both address and data, a 0 parity bit is sent to the discfile when the total number of 1 bits in the address or data word is odd as it stands (without the parity bit); i.e. when PAR (1) is asserted.

→ Insert page 71b here

(64) OPR - Operable (-SC:C5)

When the OPR level is negated (~OPR), the FER flip-flop is set immediately. Provided that the priority interrupt enable switch EFE (a flip-flop switch) is set to 1, FER(1) produces a PIE level. The resulting priority interrupt request can notify the programmer that the discfile is not operable.

It is not permissible to change the command code in CM0-1 unless the OPR level is asserted. For this reason, OPR is used as a necessary input condition for the generation of the CCM level. The CONI 270 samples ~OPR, the negation of Operable, on IOB35.

Generated by:

OPERABLE (from discfile).

(63A) OPERABLE (-SC:C2)

The discfile sends an OPERABLE signal to the discfile control through a twisted-pair transmission line to indicate that the discfile is operable. At the discfile control, the OPERABLE signal is applied to a switch filter and a pair of series-connected inverters and redesignated OPR. (See OPR below.)

This signal is generated by the closure of a relay within the discfile. The relay closure shunts the discfile end of a twisted-pair line, applying +10 vdc to a switch filter and two series-connected inverters, and thereby generating the OPERABLE signal at both polarities; see Figure -SC:C2-5.

The discfile will not close the OPERABLE relay under any of the following five conditions:

1) If the output of the positioner power supply in any disc unit falls below normal.

2) If a malfunction occurs in the flying-head air supply.

3) Should the ambient temperature in the logic unit or in any of the disc units rise to the alarm level.

4) If the logic unit or any disc unit is in test mode.

5) If the POWER OFF pushbutton at the logic unit is depressed.

Note that there cannot be an ALARM signal when the OPERABLE signal is present, although absence of ALARM does not guarantee the presence of OPERABLE. ~~Conditions 1, 2, and 3 above generate an ALARM; conditions 4 and 5 ordinarily do not. (See ALM above.)~~

(65) PAR - Parity Accumulator Flip-flop (-AC:A8)

During the read and read compare commands, the PAR flip-flop accumulates the parity of the 36 data bits and the 37th (parity) bit of each input data word. The DCK 2 pulse that is initiated by the parity bit samples the final state of PAR, and sets the PER flip-flop if PAR contains 0 (indicating that the data word currently being read shows incorrect parity).

Although a separate parity test is made for each word of the sector being processed, once any parity error is detected, the PER flip-flop remains set until reset by SCL; see PER below.

The PAR flip-flop serves an entirely different function during address transfers and write commands. For these operations, the PAR flip-flop does not test parity, but instead generates it. During SNA state of the file-control octoflop, 21 address bits are transmitted to the discfile. This includes 19 address bits, the read-next-sector bit, and a final parity bit chosen so as to make the parity of the full 21-bit address odd. The inclusion of a parity check bit in the address format permits the discfile to verify the accuracy of each address transmission.

A parity check is also provided during write operations. Each output data word sent to the discfile includes a total of 37 bits: 36 data bits followed by a 37th (parity) bit chosen so as to make the parity of the entire 37-bit data word odd.

Cleared by:

FCL $\vee$:

DCK 2 $\wedge$ COH(1)

Complemented by:

RDO $\vee$:

SDR $\wedge$ DA19(1) . $\wedge$. SNA(1) $\vee$ WRT

The PAR flip-flop is initially cleared by the FCL pulse. An FCL occurs at the IOB reset, at PTA, and at the NDP 1 pulse that signals the beginning of each data sector. For both input and output operations, the

second clear condition listed above occurs immediately following the sampling or use of the contents of PAR. During read and read compare, the DCK 2 pulse that clears PAR is the same DCK 2 that samples the state of PAR and (if PAR contains 0) sets PER. For write commands, the DCK 2 pulse that clears PAR follows the parity-bit write clock by about 400 nsec.

The PAR complement conditions implement both the parity testing and the parity generating functions described above. For read and read compare, each RDO pulse represents the reading of a 1 bit (data or parity); therefore each RDO complements PAR.

During address transfers and write commands, the complement conditions are somewhat different. With the output operations, the purpose of the PAR complement logic is to generate an appropriate odd parity bit. Consequently, the PAR flip-flop must be complemented by each non-zero address or data bit that is transferred out, but not by either the address parity bit or the output data parity bit. (After these conditions are satisfied, the correct output parity bit can be generated by simply using the complement of the final state of the PAR flip-flop; see, for example, ODO and ODZ above.)

The second line of the complement conditions listed above accomplishes just this logical function. The SNA(1) level is asserted throughout the address transmission to the discfile, and the WRT level is asserted whenever a write operation is being executed. The SDR pulse is produced by DCK 2 $\wedge$ COH(0). For both address transmission and data operations,

the COH flip-flop is set to 1 state by the DCK 2 pulse associated with the the bit that immediately precedes the parity bit (the 20th address bit, or the 36th data bit as the case may be). Consequently, no SDR pulse accompanies the transmission of either the address parity bit or the parity bit of an output data word. Although the PAR flip-flop is complemented by all other non-zero address and data bits transferred out to the discfile, it is not complemented by the associated parity bits.

(66) PER - Parity Error (-SC:A5)

Provided that the priority interrupt enable switch EFE (a flip-flop switch) is set to 1, PER(1) produces a PIE level. The resulting priority interrupt request may be used to notify the programmer that a parity error has been detected during a read or a read compare operation. The CONI 270 samples PER on IOB33.

Set by:

DCK 2 $\wedge$ CM1(1) $\wedge$ COH(1) $\wedge$ PAR(0)

Reset by:

SCL

The CM1(1) level is asserted during read and read compare commands. The only DCK 2 pulses that coincide with the assertion of COH(1) are the DCK 2 pulses that are initiated by the final bits (parity bits) of the successive 37-bit data words making up the sector that is being processed. If the PAR flip-flop is left in the 0 state after such a parity bit has been read, then a parity error is indicated. (Odd parity is used throughout the

discfile control; see PAR above.)

Note that this parity test requires somewhat compressed timing. The DCK 2 pulse that samples PAR (and sets the PER flip-flop if PAR contains 0) is initiated by the very reading of the parity bit itself, and that parity bit may require PAR to be complemented. For example, assume that the 36 data bits of the word being read from the disc have even parity. Unless an error has occurred, the 37th (parity) bit should then be 1. If it is 1, the PAR flip-flop is complemented (from 0 to 1) at the parity bit RDO time, thus leaving only about 400 nsec for PAR(0) to be negated before it is sampled by the DCK 2 pulse.

(67) PIA0-2 - Priority Interrupt Assignment Flip-flops 0-2 (-CM:B7, 8)

The processor can assign a priority-interrupt request channel to the discfile control by means of a CONO 270 instruction that sets PIA0-2 to some octal number from one to seven. If all three PIA bits are left 0, no priority interrupt channel is assigned. By asserting the PIE level, the discfile control can request a priority interrupt break on the assigned channel. The lower the channel number, the higher the priority. Breaks on channel 1 take precedence over all other breaks. Breaks on channel 2 take precedence over all breaks except those on channel 1, etc.

The PI request is sent to the processor by grounding the appropriate PIR line, i.e. the line selected from PIR1-7 by the contents of PIA0-2. If all three PIA bits are 0, the assertion of PIE has no effect on PIR0-7. The discfile control is then disabled from requesting a priority interrupt break.

Set by:

PIA0: PTC ∧ IOB15(1)

PIA1: PTC ∧ IOB16(1)

PIA2: PTC ∧ IOB17(1)

Reset by:

CCB

Like all other bits of the command buffer, the three PIA bits can be set to any desired configuration by a CONO 270 command from the processor. (The IOB15-17 inputs correspond to bits 33-35 of the CONO instruction word.)

(68) PIE - Priority Interrupt Enable (-CM:A7)

By asserting the PIE level, the discfile control can request a priority interrupt break on the assigned channel, channel 1-7. The PI request is sent to the processor by grounding the appropriate PIR line. See PIA above, and PIR below.

Generated by:

IDS(1) ∧ EIS(1) . ∨ :

DFR(1) ∧ EFR(1) . ∨ :

SEF(1) ∧ EES(1) . ∨ :

FER(1) ∨ PER(1) ∨ RCE(1) ∨ DRL(1) . ∧ . EFE(1)

When set to 1, each of the four flip-flop switches, EIS, EFR, EES, and EFE, enables the corresponding interrupt condition, or conditions (in the case of EFE), to assert the PIE level and to request a priority interrupt break on the assigned channel.

(69) PIR1-7 - Priority Interrupt Request Lines 1-7 (-CM:A7, 8)

From the discfile control to the processor run seven priority interrupt request lines PIR1-7. The discfile control can request a priority interrupt break on the assigned channel by applying a ground level to the corresponding PIR line. See PIA above.

PIR_n generated by:

PIE $\wedge$:

Contents of PIA0-2 = n octal

Unless all three priority interrupt assignment bits PIA0-2 contain 0, the assertion of PIE initiates a priority interrupt request on the channel selected by PIA0-2. If PIA0-2 all contain 0, no priority interrupt channel is assigned, and no priority interrupt breaks can be requested.

(70) PTA - Processor to Accumulator (-AD:B8)

This pulse loads IOB17-35 into the data accumulator. Bits DA0-11 are loaded from IOB18-29, DA12 remains cleared, DA13-18 are loaded from IOB30-35, and DA19 is loaded from IOB17.

Although the logic of the Discfile 5022 provides for a seven-bit sector address, the present 44-sector format requires only six address bits. For this reason DA12 always remains cleared when the processor sends a new address to the discfile control. The read-next-sector bit is sent to the discfile control over IOB17. However, the format of the discfile logic requires this bit to be at the low-order end of the address. Consequently IOB17 is loaded into DA19.

The PTA pulse also initiates the SCL and FCL clear pulses which clear most of the status and control flip-flops within the discfile control (but not the file-control octoflop).

Furthermore, the PTA pulse starts a 15-μsec delay. At the termination of this delay, an ADS pulse is produced and the file-control octoflop is stepped from IDS or ADT state to the SNA state.

Generated by:

IOB DATA SET $\wedge$.

SEL $\wedge$.

MCL(0) $\wedge$.

IDS(1) $\vee$ ADT(1)

The processor instruction DATAO 270 sends an address to the discfile control and initiates the seek for the addressed sector at the discfile. The DATAO instruction produces an IOB DATA CLR pulse followed one microsecond later by an IOB DATA SET. The same gating conditions that permit the IOB DATA CLR to produce a CDA pulse (thereby clearing the data accumulator) also permit the IOB DATA SET to produce a PTA pulse.

In order for the IOB DATA CLR and IOB DATA SET pulses to produce CDA and PTA pulses respectively, the DATAO instruction device selection code must be 270 octal. (This causes the SEL level to be asserted within the discfile control). The master clear switch must be open, since it is not proper to send a new address to the discfile control while in the process of executing a master clear operation. Finally, the discfile

control must be in either the IDS state or the ADT state (the only two states in which it is permissible to send a new address to the discfile control).

(71) PTC - Processor to Command (-CM:C4)

The PTC pulse always loads the nine low-order bits of the command buffer (CLR, EES, EFE, EFR, EIS, END, and PIA0-2) with CONO command bits from the IO bus. If the CCE level is asserted, the PTC pulse also loads the two command bits CM0 and CM1 from the bus.

Generated by:

IOB CONO SET ∧ SEL

The processor instruction CONO 270 sends command information to the command buffer of the discfile control, thereby setting up the control to execute the required commands (read, write, etc.). The CONO instruction produces an IOB CONO CLR pulse, followed one microsecond later by an IOB CONO SET. These pulses in turn produce CCB, which clears the command buffer, and PTC, which loads the command buffer with the new command information.

(72) RCE - Read Compare Error Flip-flop (-SC:A6)

When this flip-flop is set to 1, it indicates that a read compare error has occurred. During RDC command, the RCE flip-flop is set whenever an output data bit from the Data Control fails to match the corresponding input data bit read from the discfile.

Provided that the priority interrupt enable switch EFE (a flip-flop switch) is set to 1, RCE(1) produces a PIE level. The resulting priority interrupt request may be used to notify the programmer that a read compare error has been detected. The CONI 270 samples RCE on IOB32.

Set by:

DCK 3 ∧ RDC ∧ RCT(1)

Reset by:

SCL

The RDC command compares one or more sectors of output data (or even some fraction of a full sector) with corresponding input data read from the discfile. The comparison is executed one bit at a time, but does not include parity bits. At the first pair of bits that are not identical, the comparison fails. The failure is detected by the fact that the DCK 2 pulse leaves the RCT flip-flop in the 1 state; see RCT below. The RCT remains set for only about 400 nsec; it is cleared at DCK 3, and the comparison then continues until the output data is exhausted. Although RCT is cleared soon after being set, the RCE flip-flop saves the read compare error for sampling. The same DCK 3 pulse that clears RCT sets RCE, and RCE remains set until being reset by SCL.

(73) RCT - Read Compare Test Flip-flop (-SC:A8)

If, during a read compare, RCT contains 1 at DCK 2 time, then the DCK 3 pulse sets the RCE flip-flop, thus indicating a read compare error.

Cleared by:

FCL $\vee$ DCK 3

Complemented by:

RDO $\wedge$ COH(0) . $\vee$:

SDR $\wedge$ DA19(1)

The RCT flip-flop is initially cleared by the FCL pulse. An FCL occurs at IOB reset, at PTA, and at the NDP 1 pulse that signals the beginning of each data sector. The DCK 3 pulse also clears RCT. This clear occurs after each data bit comparison. The same DCK 3 pulse that clears RCT also sets RCE if RCT contains 1 (indicating a read compare error).

The first line of the complement conditions listed above represents input data read from the discfile; the second line represents output data from the Data Control 136. Throughout the read compare command, RCT is complemented twice or not at all for each error-free bit comparison. Single complementing, which leaves RCT in the 1 state after the DCK 2 pulse, occurs only when an input data bit fails to match the corresponding output data bit; i. e. when there is a read compare error. No output parity bits are generated during RDC commands, and for read comparison purposes the incoming parity bits are ignored. (Note, however, that the incoming data from the discfile is subject to the normal parity test during read compare commands, just as during read commands. See PER above.)

The clock chain is started by incoming data pulses from the discfile. Each non-zero bit produces an RDO pulse. Every RDO except those that are initiated by input parity bits (and so are screened out by the 1 state

of the COH flip-flop) complements the RCT flip-flop from 0 to 1. This is the first complement operation (input data).

The second complement operation (output data) is clocked by the SDR pulse. The SDR pulse is produced by DCK 2 $\wedge$ COH(0). Every incoming data pulse starts the clock chain, and thus results in the generation of a DCK 2 pulse. Except for parity bits (again screened out by the 1 state of COH), every DCK 2 generates an SDR pulse. Provided that the output data bit in DA19 is a 1, the SDR pulse recomplements the RCT flip-flop at DCK 2 time.

The two sets of complement conditions work together to effect the bit-by-bit read comparison. If both the input and the output data are 0 bits, RCT will not be complemented at all, but will remain in its initial 0 state. If both bits are 1, the result is the same; although RCT is complemented twice, it is still left in the 0 state after the DCK 2 pulse. If, however, the input and output data bits are different, RCT is complemented only once, and is therefore left in the 1 state after DCK 2. The RCE flip-flop is then set by the DCK 3 pulse, thus saving the read compare error indication.

(74) RDC - Read Compare Command (-CM:A1)

This level indicates that the contents of CM0-1 specify the read compare command. If the RCT flip-flop ever remains in the 1 state at DCK 3 time while RDC is asserted, then the RCE flip-flop is set, indicating the detection of a read compare error.

There are certain other operations of the discfile control that are common to both the read compare command and the write command. These operations are enabled by the CM0(1) command-pair function rather than by the RDC level. Similarly, the CM1(1) command pair function is used instead of the RDC level to enable other operations that are common to the read compare command and the read command. (See CM0-1 above for a more detailed description of these two command-pair functions and the logical operations which they govern.)

Generated by:

CM0(1) ∧ CM1(1)

(75) RDO - Read Data Ones (-TR:D2)

During read commands, every RDO pulse except those initiated by data word parity bits loads a 1 into bit DA2. During both read and read compare commands, every RDO pulse except those initiated by data word parity bits complements the RCT flip-flop. For both read and read compare commands, every RDO pulse (including those produced by the parity bits) complements the PAR flip-flop.

Generated by:

EDO ∨ ODO . ∧:

RDS

For both the read and read compare commands, the RDS level enables each EDO or ODO input data pulse to initiate an RDO pulse. During both commands, the RDS level is asserted until the WCT7 flip-flop is set to 1;

this always occurs when the word count reaches 128. For the read command only, WCT7 can also be set to 1 by setting the END flip-flop. (See RDS below.)

(76) RDS - Receive Data Signal (-TR:C6)

The RDS level enables the generation of RDO and RDZ data pulses. When RDS is asserted, every EDO or ODO input data pulse from the discfile produces an RDO pulse, and every EDZ or ODZ produces an RDZ pulse.

The RDS level also determines what signals are allowed to start the DCK clock chain. The assertion of the RDS level prevents the WCE and WCO pulses from triggering DCK 1. However, the RDS level must be asserted in order for the input data pulses (EDO, EDZ, ODO, and ODZ) to trigger DCK 1.

Throughout the address transmission, the RDS level is not asserted (even if the command is read or read compare) because, when the file-control octoflop is in SNA state, SCS(1) is not asserted. Because RDS is negated, the WCE pulses initiated by the address clock are able to trigger DCK 1 and thus to initiate the DCK clock chain.

During the execution of the write command, CM1 is in the 0 state and therefore the RDS level is not asserted. Because RDS is negated, ~~both the WCE and WCO~~ the WRITE CLOCK EVEN and WRITE CLOCK ODD pulses initiated by the write clock control track in the discfile (inner-zone or outer-zone track depending upon the location of the sector to be written) are allowed to trigger DCK 1.

For both the read and the read compare commands, the assertion of the RDS level enables each input data pulse (EDO, EDZ, ODO, or ODZ) to trigger DCK 1 and thus initiate the DCK 1 clock chain.

Generated by:

CM1(1) ∧ SCS(1) ∧ WCT7(0)

The WCT7(0) condition provides the means of terminating the read and read compare commands. During both of these commands, the RDS level is asserted until the WCT7 flip-flop is set to 1; this always occurs when the word count reaches 128. For the read command only, WCT7 can also be set to 1 by setting the END flip-flop. When END contains 1, the first DCK 1 pulse to appear sets WCT7 and thereby ends RDS and turns off the DCK clock. With RDS no longer asserted, neither RDO nor RDZ pulses can be generated. The turn-off of the DCK clock initiates the remaining operations required to end the command and reset the system. The read command can thus be terminated at any point by merely setting the END flip-flop.

(77) RDY - Ready (-TR:C5)

Provided that ADE(0) is asserted (indicating that no address errors are present), the RDY pulse advances the file-control octoflop from ADT state to DFR state, and starts the 1-ms delay that measures the duration of DFR. If, at the termination of this delay, no command code is present in CM0-1 (NOP), the octoflop returns from DFR state to ADT state. However, if a command code is present (~NOP), then an ALP pulse is

sent to the discfile, and the octoflop is advanced to ALS state.

Generated by:

READY $\wedge$ CLR(0)

Except when the CLR flip-flop is set, each READY pulse from the discfile produces a RDY pulse within the discfile control. When CLR is set, END is also set, and it is not desirable to generate RDY pulses. While CLR is set, the discfile control cannot advance beyond ADT state.

When the read-next-sector bit is 0, the discfile sends the discfile control one READY pulse for each 52-ms revolution of the discs. The pulse is transmitted when the discfile senses the header of the sector immediately preceding the sector to be operated upon.

If the read-next-sector bit is 1, the discfile sends a READY at every sector; i.e. either four or seven times each revolution (depending upon whether the addressed track is in the inner or outer zone of the addressed disc).

Once an ALERT pulse has been accepted by the discfile, no further READY signals are emitted. The ALERT is always followed 15 μsec later by a READ or WRITE pulse. Since the READY pulse is sent at the header of the "get ready" sector preceding the sector to be operated upon, the ALP and the READ or WRITE command pulse arrive in ample time to govern the operations to be performed.

(78) RDZ - Read Data Zeros -(TR:D3)

During read commands, every RDZ pulse except those initiated by data word parity bits loads a 0 into bit DA2.

Generated by:

EDZ ∨ ODZ . :

RDS

For both the read and read compare commands, the RDS level enables each EDZ or ODZ input data pulse to initiate an RDZ pulse. During both commands, the RDS level is asserted until the WCT7 flip-flop is set to 1; this always occurs when the word count reaches 128. For the read command only, WCT7 can also be set to 1 by setting the END flip-flop. (See RDS above.)

(79) READ (-TR:A7)

The discfile control sends a READ pulse to the discfile through a twisted-pair transmission line to command the discfile to read a sector. If an additional READ signal is sent during the 8-μsec guard slot after the final data pulse of the sector, then the discfile increments its stored address by 1 and reads the next sector as well.

Generated by:

CSP ∧ CM1(1) . ∨:

ESP 2 ∧ CM1(1)

The CM1(1) command-pair function is asserted during the read and read compare commands. For both these commands, a READ pulse is

sent to the discfile at the CSP pulse that precedes the beginning of the first sector of the command. If the command code is still in the command register CM0-1 after the ESP 1 pulse marking the end of a given sector (indicating that the command is to continue to read or read compare additional sectors), then the closely following ESP 2 sends another READ pulse to the discfile. This process can continue throughout the 44 sequentially addressed sectors at a given disc position before the same data is read a second time. To read a file containing more than 44 sectors the programmer must give an additional DATAO 270 instruction that changes the disc and/or position fields.

(80) READY (-TR:A4)

When the discfile is ready to be alerted for a read or write operation it sends a READY pulse to the discfile control. The READY pulse is sent to the discfile control through the same twisted-pair line upon which the CLEAR pulse is received. The READY pulse marks the completion of the discfile's address seek position. It is generated after the positioner has settled on the address track and the read-write head has sensed the header of the "get ready" sector immediately preceding the sector in which the reading or writing is to be performed.

If the read-next-sector bit is 0, the discfile sends the discfile control only one READY pulse for each revolution of the discs. That pulse is transmitted when the discfile senses the header of the sector immediately preceding the sector to be operated upon. The header must agree completely with the entire address that has been sent to the discfile.

However, if the read-next-sector bit is 1, complete matching of the stored address to the header sensed is not required. It is sufficient to produce a READY if the header that is sensed agrees with the first sixteen bits of the address. Those sixteen bits comprise the disc and position fields, and the track portion of the record field. The header that is sensed need not, in order to generate a READY pulse, match the next three address bits, those which specify a single sector within the addressed track. (The addressed track may contain four or seven sectors depending upon whether it is located in the inner or outer zone of the addressed disc.) Once the discfile has found the addressed track, it sends the discfile control a READY at every sector; i. e. four or seven times each revolution. The READY pulses continue until an ALERT pulse has been received by the discfile; once the ALERT has been accepted, no further READY pulses are generated.

(81) RED - Read Command (-CM:A2)

This level indicates that the contents of CM0-1 specify the read command. When COH(0) is also asserted (indicating that the current RDO or RDZ pulse has been initiated by a data bit rather than by a parity bit) the RED level gates the RDO or RDZ data bit through into DA2.

Input data from the discfile is loaded into DA2 as it is read in serial fashion, one bit at a time, and is then shifted right until a full 18-bit data character is stored in DA2-19. At the conclusion of this loading process, the discfile control sends a Take a Character pulse to the Data Control 136,

causing the data control to strobe the character into its low order accumulator bits, and (if the character is the first of the two 18-bit characters making up the 36-bit data word) to shift it left to make room for the second character.

Although the Take a Character pulse is produced only during the execution of read commands, the RED level is not used to enable the generation of Take a Character. Instead of the RED level, the CM0(0) command-pair function (which is logically equivalent to NOP $\vee$ RED) is used for this purpose. The same CM0(0) level enables END(1) to set the WCT7 flip-flop during read commands only. The use of CM0(0) for these purposes rather than RED is, however, of no logical significance. Because the CM0(0) level is ANDed with SCS(1) (which can never be asserted during NOP), the resulting enable function is the exact logical equivalent of the RED level. (The CM0(0) level is used instead of the RED level only to ~~simplify wiring~~ reduce signal propagation time, and not from any logical necessity.)

There are certain other operations of the discfile control that are common to both the read command and the read compare command. These operations are enabled by the CM1(1) command-pair function rather than by the RED level. (See CM0-1 above for a more detailed description of this command-pair function and the logical operations which it governs.)

Generated by:

CM0(0) $\wedge$ CM1(1)

(82) SCE - Sector End (-FC:B7)

Eighth and final state of the file-control octoflop. The ESP 1 pulse advances the octoflop from SCS state to SCE state at the end of each sector. The octoflop remains in SCE state for 2 μsec. At the end of that interval, the octoflop is either reset to IDS state or returns to CMS state.

Although the SCE state lasts only for 2 μsec, the SEF flag saves SCE for sampling. The SCE(1) level enables the ESP 2 pulse to set SEF, which then remains set until being reset by an SCL pulse.

It is not permissible to change the command code in CM0-1 after the ALP has advanced the octoflop from DFR to ALS state. For this reason, the SCS(0) level is used as a necessary input condition for the generation of the CCM level.

Generated by:

ESP 1.

The discfile control senses the end of each sector of data by means of an integrating 5-μsec delay. When no DCK 1 clock pulses have been sensed for 5 μsec, this delay times out, providing the ESP 1 pulse and, 2 μsec later, the ESP 2 pulse. The octoflop is advanced from SCS state to SCE state by ESP 1. The ESP 2 pulse then performs one of two alternate functions. If there is no command code present in CM0-1 (NOP), no further operations are required, and the ESP 2 pulse resets the octoflop to IDS state. However, if a command code is present (~NOP), additional

sectors of data are to be operated upon. The ESP 2 pulse then sends another READ or WRITE pulse to the discfile, and simultaneously returns the octoflop from SCE state to CMS state.

(83) SCL - Status Clear (-SC:C6)

The SCL pulse clears the nine error-status flags: ADE, CME, DCE, DRL, FER, PER, RCE, SEF, and WLE.

Generated by:

IBR $\vee$.

PTA $\vee$.

PTC $\wedge$ IOB10(1)

An SCL pulse is generated by IBR when computer power is first turned on, when the IO reset key is operated, and when a programmed reset is executed. This ensures that the discfile control begins operations with the error-status flags cleared. Whenever a DATAO 270 instruction is used to send a new address to the discfile control, a PTA pulse is produced and the PTA in turn produces an SCL pulse to clear the discfile control.

The SCL pulse can also be controlled by the program. A CONO 270 instruction with bit 28 in the 1 state produces an SCL pulse at PTC time. (The IOB10 input corresponds to bit 28 of the CONO instruction word.) This permits the programmer to clear the error-status flags (usually after printing out the error status and executing other appropriate steps to service the error-induced interrupt).

(84) SCS - Sector Started (-FC:B6)

Seventh state of the file-control octoflop. After the octoflop has been in CMS state for 200 μsec, the NDP 1 pulse steps it to SCS state. The octoflop remains in SCS state until reaching the end of the sector; at that point the ESP 1 pulse advances the octoflop to its final SCE state.

Throughout the SCS state, the SCS(1) level enables each DCK 1 clock pulse to restart the 5-μsec integrating delay AB01. This delay permits the discfile control to sense the end of each sector of data. When no DCK 1 pulses have been applied to the delay for 5 μsec, the delay times out, producing the ESP 1 and ESP 2 pulses that mark the end of sector.

The SCS(1) level is applied to the input gating of BSC3. Since BSC3 has a weight of two in the bit-shift counter, setting it at the start of a count reduces the actual final count of the bit-shift counter by two. Although the counter always overflows when its contents are 20, if BSC3 is set before counting is begun only 18 counts are required to reach the overflow count of 20. For address transmission, SCS(1) is not asserted, BSC3 is not set, and the full 20 count is required to produce the overflow signal, BCO. However, for operations upon data (both output and input), SCS(1) is asserted, and therefore BSC3 is set at the end of each character. This ensures that the bit count of the following character starts at two so that overflow occurs at the DCK 1 clock corresponding to the eighteenth and final bit of each 18-bit character.

During read and read-compare commands SCS(1) causes RDS to be asserted, thereby enabling the data clock chain. Each incoming data bit

from the discfile then produces the following sequence of clock pulses: DCK 1, DCK 0, DCK 2, DCK 2.1, and DCK 3.

During the read command (data input from discfile to processor via Data Control 136), SCS(1) permits the application of a Take a Character pulse to the Data Control 136. The Take a Character pulse is sent to the Data Control at the same DCK 1 pulse that initiates the BCO. The Take a Character pulse causes the data control to strobe the 18-bit character into its low order accumulator bits, and (if required) to shift the accumulator left to make room for the second 18-bit character of the 36-bit data word.

The SCS(1) level also performs a second function during read commands. It permits the END(1) level to set the WCT7 flip-flop, thereby terminating the RDS level and turning off the clock. This permits terminating the read command at any point (even within a sector of data, or for that matter within a word within a sector).

During the output commands, write and read compare, assertion of SCS(1) causes the DCK 2.1 pulse that immediately follows the next BCO to initiate an SND pulse and a DTC pulse. The SND pulse gates an 18-bit output word from the Data Control 136 into the data accumulator of the discfile control. The DTC pulse sends the Data Control 136 a Give a Character pulse to shift the next output word into position for transfer into the discfile control.

It is not permissible to change the command code in CM0-1 after the ALP has advanced the octoflop from DFR to ALS state. For this reason, the SCS(0) level is used as a necessary input condition for the generation of the CCM level.

If an ERROR signal arrives from the discfile while the octoflop is in SCS state, the DCE flip-flop is set, indicating that the error is a data clock error. (See DCE above.) The CONI 270 samples SCS on IOB24.

Generated by:

NDP 1

An NDP 1 pulse is almost invariably produced 200 μsec after the octoflop is stepped to CMS state. (In the unlikely event that an IOB reset pulse should occur during the 200-μsec interval, no NDP 1 pulse would be generated. Instead of being advanced from CMS to SCS state, the octoflop would then be reset from CMS to IDS state.)

(85) SDR - Shift Data Right (-AD:B8 and -AC:C1)

The SDR pulse shifts the contents of DA2-18 one bit-position to the right. If ACE(1) is asserted, indicating that address transmission is in progress, the contents of DA19 are ring-shifted back into DA0. Since DA0-19 is directly accessible to the processor by means of the DATAI 270 instruction, this ring-shift feature permits a convenient check on the operation of the data accumulator shift logic. If ACE(0) is asserted, indicating that the current operation is a data operation, rather than address transmission, the contents of DA19 are not ring-shifted back into DA0; instead, at each SDR, a 0 is loaded into DA0.

Generated by:

DCK 2 ∧ COH(0)

An SDR pulse is generated at the DCK 2 pulse initiated by each address and data bit with the exception of the address and data parity bits. Because of the heavy load which SDR drives, it is generated by two independent pulse amplifiers with identical input conditions.

(86) SEF - Sector End Flag (-SC:A7)

This flag is set 2 μsec after the file-control octoflop is advanced to SCE state. The SEF remains set until being reset by an SCL pulse. The primary purpose of SEF is to save the SCE state for sampling. Even though SCE is no longer asserted, a 1 in SEF lets the programmer know that the end of the data sector has been passed. The CONI 270 samples SEF on IOB25.

Set by:

ESP 2 ∧ SCE(1)

Reset by:

SCL

The SCE state of the octoflop lasts only for the 2-μsec interval between ESP 1 and ESP 2; the SEF flag is set just as the octoflop is stepped out of SCE state. The SEF remains set until being reset by an SCL pulse. The SCL pulse that resets SEF can be produced in three ways: 1) by an IOB reset; 2) by the PTA pulse that accompanies a new DATAO 270 command; and 3) as a deliberately programmed SCL: the CONO 270 instruction produces an SCL at PTC time provided that bit 28 is coded to contain a 1.

(87) SEL - Select Discfile Control 270 (-SC:D2)

During the CONO instruction, this level gates the IOB CONO CLR and IOB CONO SET pulses from the processor to produce the CCB and PTC pulses. These two pulses clear the command buffer and set it to the new configuration specified by the CONO. For the CONI instruction, the SEL level allows the IOB STATUS pulse from the processor to produce the STP pulse; STP causes the processor to sample the status of various registers and flip-flops within the discfile control.

The SEL level also gates the processor signals sent to the discfile control during DATAO and DATAI instructions. For DATAO commands SEL is one of the conditions that must be present for the IOB DATA CLR and IOB DATA SET pulses to initiate CDA and PTA respectively. These two pulses clear the data accumulator, and transfer in a new address from IOB17-35. At a DATAI instruction, SEL allows IOB DATA I to initiate DTP. The DTP pulse then gates the contents of the data accumulator and the bit shift counter onto the IO bus.

Generated by:

Device selection code 270 applied to IOS3-9.

Not selected

(88) ~SEL 5 - ~~Select~~ Device 5 (-SC:B7)

If the ~SEL 5 level is asserted (at ground),

~~Unless the SEL 5 level is negatively asserted~~, the data request late flip-flop DRL is set by the NDP 2 pulse.

Generated by:

DC DEVICE SEL 5 (From Data Control 136)

When the internal device address in control-status register bits IC30-32 selects device 5 (the discfile control), the Data Control 136 sends a negative DC DEVICE SEL 5 level to the discfile control. If the Data Control 136 has not been selected to the discfile control, the DC DEVICE SEL 5 level is ground, thus asserting ~ SEL 5 at ground.

(89) SELECT (-TR:A6)

The discfile control sends a SELECT pulse to the discfile through a twisted-pair transmission line to prepare the discfile to receive and store an address. The SELECT pulse connects the discfile to the discfile control and clears the discfile of previously induced error conditions.

Generated by:

PTA

The DATAO 270 instruction IOB DATA SET initiates the PTA pulse that generates SELECT. Neither the PTA pulse nor the SELECT pulse can be produced unless the file-control octoflop is in IDS state or ADT state. If a SELECT pulse is sent to the discfile during ADT state (after the ATP pulse, but before RDY), the SELECT causes a seek interrupt at the discfile. The seek interrupt automatically interrupts the seek operation being executed by the discfile. (This would occur if the programmer were to send an address to the discfile and then were to send another new address before the discfile had found the first address.)

The time required to execute a seek interrupt depends upon how far the interrupted seek operation has progressed. If the new SELECT arrives

before the positioner has been set in motion, the discfile can begin to operate upon the new address immediately, so there is no increase in the required access time. If, however, the new SELECT pulse is received after the positioner has already been set in motion, there is an added delay before the new address can be operated upon. This delay is required to ensure that power is not removed from the positioner while it is in motion; its duration is dependent upon the time that the discfile has spent operating upon the old address, but shall not exceed a nominal 250 ms.

(90) SNA - Select New Address (-FC:B2)

Second state of the file-control octoflop. During this state, the address of the "get-ready" sector preceding the next data sector to be operated upon is transmitted to the discfile. A total of 21 bits are sent to the discfile. This includes 19 address bits, the read-next-sector bit, and a parity bit which permits the discfile to verify the accuracy of the address transmission. The SNA state is a necessary condition for the generation of the ATP pulse. The SNA(1) level also provides an input to the PAR flip-flop complement logic. When SNA(1) is asserted, the PAR flip-flop is complemented whenever DA19 contains a 1 at the time of the SDR pulse. In this way, the PAR flip-flop generates the address parity bit. The CONI 270 samples SNA on IOB19.

Generated by:

PTA plus 15-μsec delay $\wedge$.

IDS $\vee$ ADT

The PTA pulse starts a 15-μsec delay. The termination of this delay steps the octoflop from IDS or ADT state to SNA state and simultaneously generates an ADS pulse. (See ADS above.) Usually the DATAO 270 instruction that produces PTA is given during the IDS state of the octoflop. However, the programmer has the option of changing addresses at any time during ADT state.

After completion of the address transmission, an ATP pulse is sent to the discfile, and the octoflop advances from SNA to ADT state. If there is no command code stored in CM0-1, the octoflop then oscillates between ADT state and DFR state (it is held in DFR state during only one millisecond of each 52-millisecond disc-revolution). The programmer can change addresses whenever the octoflop is in ADT state or in IDS state, but not during DFR state.

(91) SND - Sample New Data (-AD:C8)

The SND pulse causes the contents of the 18 high order bits of the Data Control 136 data accumulator to be loaded into bits DA2-19 of the discfile control data accumulator.

Generated by:

DTC

The DTC pulse always produces both an SND pulse and a GIVE A CHARACTER pulse. During the two output commands, write and read compare, the DTC pulse is generated at the CSP pulse that precedes the beginning of each data sector, and at the DCK 2.1 pulse corresponding to

the 18th and final data bit of each data character throughout the 128-word sector. A programmed DTC can also be given at will from the processor. This DTC is generated at the IOB CONO SET time of any CONO 270 instruction that has a 1 in bit 19. (See DTC above for a more detailed description of the function and generating conditions of DTC, SND, and GIVE A CHARACTER.)

(92) STP - Status to Processor (-CM:C8)

This 2.5-μsec level gates the contents of the command buffer, the file-control octoflop, the nine error-status flags, and the relay-meter test switches onto the IO bus. The processor reads in this information from the bus at the 2-μsec point.

Generated by:

IOB STATUS ∧ SEL

The processor CONI instruction produces a 2.5-μsec negative IOB STATUS level. This command level is effective at the discfile control only when SEL is asserted, i.e. only when the CONI instruction device selection code is 270 octal.

(93) TAKE A CHARACTER (-SC:C7)

The TAKE A CHARACTER PULSE is sent to the Data Control 136 accumulator shift logic. The TAKE A CHARACTER PULSE signals the data control that the discfile control has presented an 18-bit input charac ter to the low order strobe data inputs of the data control data accumulator. This indicates to the data control that it may strobe the

character into the low order bits of the data control data accumulator and that unless the character is the last (second) character in the input data word, it should begin to shift the contents of its accumulator left by 18 bit-positions to make room for the next character to be read in. If the character is the last character of the input data word, the TAKE A CHARACTER pulse causes the data control to strobe the character into the low order accumulator bits and then to advance the entire 36-bit data word from its data accumulator DA into its data buffer DB.

Generated by:

DCK 1 $\wedge$ CM0(0) $\wedge$ SCS(1) $\wedge$ END(0) $\wedge$

BSC0(1) $\wedge$ BSC3(1) $\wedge$ BSC4(1) . $\vee$:

PTC $\wedge$ IOB2(1)

The CM0(0) command-pair function is asserted both for NOP and for the read command. However, CM0(0) and SCS(1) are asserted together only during the read command. If the END flip-flop is set, no further input characters may be strobed into the data control.

The DCK 1 pulse that generates the TAKE A CHARACTER pulse is initiated by the final data bit of each 18-bit data character. Note that the BSC enabling conditions represent a count of only 19; the 19 count is used rather than the usual overflow count of 20 because the TAKE A CHARACTER is generated early in the DCK chain; it is generated at the DCK 1 pulse, the same DCK pulse that increments the bit-shift counter.

The last line of the generating conditions listed above represents a programmed TAKE A CHARACTER pulse that can be given at will from

the processor. This TAKE A CHARACTER is generated at the IOB CONO SET time of any CONO 270 instruction that has a 1 in bit 20. (The IOB2 input corresponds to bit 20 of the CONO instruction word.)

(94) WCE - Write Clock Even (-TR:D7)

Address transmission to the discfile is synchronized to WCE pulses generated within the discfile control. (See CLK above.) Write operations, however, are synchronized to WCE and WCO pulses initiated by WRITE CLOCK EVEN and WRITE CLOCK ODD pulses generated at the discfile. These WRITE CLOCK pulses are transmitted from the discfile over two twisted-pair transmission lines to the discfile control. (See WRITE CLOCK EVEN below.)

Depending upon whether the address bit (during address transmission) or the data bit (during write operations) corresponding to a given WCE pulse is a 1 or a 0, the WCE pulse produces an EDO or an EDZ pulse. These pulses in turn are returned to the discfile over twisted pairs, and there clock in the address (data and parity) or cause the writing of 1s or or 0s (data and parity) as the case may be.

During address transmission, the WCE pulse starts the DCK clock chain, producing in sequence the following pulses: DCK 1, DCK 0, DCK 2, DCK 2.1, and DCK 3. For write operations, the WCE also starts the DCK clock chain (but only provided that the WCT counter has not yet reached its final count of 128).

Generated by:

CLK $\wedge$ ACS(1) . $\vee$:

WRITE CLOCK EVEN

The first condition specified above governs the generation of WCE pulses during address transmission. See CLK and ACS above.) The discfile control tracks furnish both WRITE CLOCK EVEN and WRITE CLOCK ODD pulses during the execution of write commands.

(95) WCO - Write Clock Odd (-TR:C6)

Write operations are synchronized to WCE and WCO pulses initiated by WRITE CLOCK EVEN and WRITE CLOCK ODD pulses generated at the discfile. These WRITE CLOCK pulses are transmitted from the discfile over two twisted-pair transmission lines to the discfile control. (See WRITE CLOCK EVEN below.)

the discfile control. Depending upon whether the data bit (or parity bit) corresponding to a given write clock odd is a 1 or a 0, the WCO pulses produce ODO or ODZ pulses; these pulses are returned to the discfile over twisted pairs and there cause the writing of 1s or 0s respectively.

Provided that the current command is a write command, and that the WCT counter has not yet reached its final count of 128, each write clock (even or odd) starts the DCK clock chain, resulting in the generation of DCK 1, DCK 0, DCK 2, DCK 2.1, and DCK 3.

Generated by:

Discfile control tracks

(See WCE above.)

(96) WCT - Word Counter (-WC:A3-7)

The word counter's function is to count the number of 37-bit data words that have been processed, and to end the current command when the count reaches 128 or, in the case of the read command only, when the END flip-flop is set. The only output from the word counter is the WCT7(0) level; this level is asserted so long as the contents of the word counter are less than 128. The WCT7(0) level must be asserted in order for RDS to be asserted. Neither the odd write clock pulses WCO nor the even write clock pulses WCE can initiate the DCK clock chain unless WCT7(0) is asserted.

Cleared by:

FCL

Incremented by:

DCK 2 ∧ BCO ∧ WDC(1)

WCT7 Set by (Adds 128 to WCT Contents):

DCK 1 ∧ CM0(0) ∧ SCS(1) ∧ END(1)

The FCL provides an initial clear; it clears the WCT before each new address is transferred out and at the beginning of each new data sector. The WCT is incremented once at the end of the address transmission. This is a "don't care" condition, since an FCL pulse clears out the 1 count at the beginning of the first data sector to be processed.

Throughout all data operations (read, write, read compare commands) the WCT is incremented at the 36th bit of each 37-bit data word (the final data bit before the parity bit). No incrementing occurs at the BCO corresponding to the final bit of the first character of each data word; this is because WDC(1) is asserted only during alternate (even-numbered) data characters. (See WDC below.)

The CM0(0) command-pair function is asserted both for NOP and for the read command. However, CM)(0) and SCS(1) are asserted together only during the read command. The read command can be terminated at any point by setting the END flip-flop. (This can easily be done by giving a CONO 270 instruction with a 1 in bit 23.) When END is set to 1, the first DCK 1 pulse to appear sets WCT7, and thereby ends the RDS level and turns off the DCK clock. With RDS no longer asserted, neither RDO nor RDZ pulses can be generated. The turn-off of the DCK clock initiates the remaining operations required to end the read command and reset the

system. Five μsec after the last DCK 1 pulse, the integrating delay AB01 times out, producing an ESP 1 pulse. That ESP 1 pulse clears the command register CM0-1, and 2 μsec later the ESP 2 pulse resets the octoflop to IDS state. The discfile control is then ready to receive a new command.

The read command differs from the output commands (write and read compare) in that it can be terminated in mid-sector. The output commands must always proceed to the end of the current 128-word sector even though the END flip-flop is set prior to the end of the sector. This is true even if the output data from the discfile is exhausted before the sector ends. If no further output data is available, 0's are written (or compared) for the rest of the 128-word sector.

(97) WDC - Word Control Flip-flop (-AD:C7)

The WDC flip-flop must be in the 1 state in order for the COH flip-flop to be set. The WDC flip-flop must also be in the 1 state for the WCT counter to be incremented.

Set by:

ADS

Reset by:

FCL

Complemented by:

DCK 2 ∧ BCO

The FCL provides an initial clear; it clears WDC before each new address is transferred out and at the beginning of each new data sector. The

ADS pulse that initiates the turn-on of the address clock sets WDC at the beginning of address transmission. The WDC flip-flop remains in the 1 state throughout the address transmission. The BCO level is asserted at the DCK 1 pulse corresponding to the 20th bit of the address (the last address bit before the parity bit). The single DCK 2 pulse that occurs while BCO is asserted sets the COH flip-flop and complements the WDC flip-flop (resetting it to the 0 state). The COH(1) level remains until the next DCK 2 pulse; it thus identifies the following address bit (the 21st and final bit of the address) as the address parity bit.

Throughout the data operations (the read, write, and read compare commands) the WDC flip-flop acts as a count-of-two counter. The FCL pulse ensures that the WDC flip-flop starts the data sector in the 0 state. The DCK 2 pulse then complements WDC at the BCO corresponding to the final data bit of each of the 18-bit data characters making up the sector. Provided that the WDC flip-flop already contains 1, the same DCK 2 pulse that complements the WDC flip-flop also sets the COH flip-flop to the 1 state and increments the WCT counter. As a result, only the DCK 2 pulse initiated by the 36th data bit of each data word sets the COH flip-flop and increments the WCT counter. The COH(1) level remains until the next DCK 2 pulse; it thus identifies the following data bit (the 37th and final bit of the current data word) as the data word parity bit. The WCT counter is incremented only at the even-numbered 18-bit data characters. This allows the use of the WCT to count the number of 37-bit data words that have been processed during the current sector.

(98) WLE - Write Lock Error Flip-flop (-SC:A3)

When this flip-flop is set, it indicates that a locked out disc has been addressed. The CONI 270 samples the contents of WLE on IOB28.

Set by:

WRITE LOCKOUT WARNING (from discfile)

Reset by:

SCL

Each discfile unit contains a panel of 16 lockout switches; each of these switches corresponds to one of the 16 data discs in the discfile. When a given switch is operated, it becomes impossible to write upon or erase the corresponding disc. The data on the locked out disc is preserved for reading only. The usefulness of this data-protection safeguard is augmented by the fact that the lockout switch panel is located behind a door which can be locked.

Should the processor address a locked out disc for either reading or writing, the discfile sends a WRITE LOCKOUT WARNING to the discfile control. This pulse arrives at the discfile control with or before the READY pulse and immediately sets both the FER flip-flop and the WLE flip-flop. Provided that the priority interrupt enable switch EFE (a flip-flop switch) is set to 1, FER(1) produces a PIE level. The resulting priority interrupt request can be used to warn the programmer that he has addressed a locked out disc.

If the command is a read or read compare, there is no difficulty in reading the data from the locked out discs, and no further error indications

result. However, if the discfile control should attempt to execute a write command in spite of the WRITE LOCKOUT WARNING, then the writing or erasure of data is prevented and a data check error is generated. The data check error produces a DC error indication at the discfile, and also sends an ERROR SIGNAL to the discfile control. At the discfile control, the ERROR SIGNAL sets the FER flip-flop and also the CME flip-flop.

(99) WRITE (-TR:A8)

The discfile control sends a WRITE pulse to the discfile through a twisted-pair transmission line to command the discfile to write out a sector of data. If an additional WRITE signal is sent during the 8-μsec guard-slot after the final data pulse of the sector, then the discfile increments its stored address by 1 and writes another sector.

Generated by:

CSP ∧ WRT . ∨:

ESP 2 ∧ WRT

During every write command, a WRITE pulse is sent to the discfile at the CSP pulse that precedes the beginning of the first sector of the command. If the command code is still in the command register CM0-1 after the ESP 1 pulse marking the end of a given sector (indicating that the command is to continue to write additional sectors), then the closely following ESP 2 pulse sends another WRITE pulse to the discfile. This process can continue throughout the 44 sequentially addressed sectors at a given disc position before any sector is written upon a second time. To

prevent inadvertent destruction of data, it is important that the programmer keep count of the number of records sequentially written at a single position. In order to write a file containing more than 44 sectors, the programmer should give additional DATAO 270 instructions that change the disc and/or position fields.

→ Insert page 112b here

(100) WRITE LOCKOUT WARNING (-SC:B1)

The discfile sends a WRITE LOCKOUT WARNING to the discfile control through a twisted-pair transmission line to indicate that the processor has addressed a locked out disc. This pulse arrives at the discfile control with or before the READY pulse and immediately sets both the FER flip-flop and the WLE flip-flop. (See WLE above.)

(101) WRT - Write Command (-CM:A1)

This level indicates that the contents of CM0-1 specify the write command. The WRT level enables both the CSP pulse and the ESP 2 pulse to transmit WRITE pulses to the discfile. The CSP pulse initiates the first WRITE pulse of a write command. If the command is to write more than one sector of output data, the ESP 2 pulse that follows the end of each sector sends the discfile another WRITE. Each WRITE pulse after the first increments the discfile address. Up to 44 successive sectors can be written at a single position without destroying any data previously written.

There are certain other operations of the discfile control that are common to both the write command and the read compare command.

(99A) WRITE CLOCK EVEN (-TR:D5)

The discfile control tracks furnish both WRITE CLOCK EVEN pulses and WRITE CLOCK ODD pulses during the execution of write commands. Within the discfile control these pulses are inverted and redesignated WCE and WCO. Note that during address transmission WCE pulses are also generated within the discfile control. (See WCE above.) One of the four control tracks on the top baffle disc of the discfile generates clock pulses for inner zone writing; a second control track generates control pulses for outer zone writing. The write clock frequency depends upon whether the addressed sector (at which the writing is to be done) is located in the inner or outer zone. The control track for the outer zone contains approximately 35,000 ~~34,840~~ bits; the control track for the inner zone contains only about 21,000 ~~20,904~~ bits; both tracks are recorded with a one-bit maximum anomaly at the splice point.

Write clock pulses from both control tracks are sent to the discfile control on two twisted pairs (odd and even). The first, third, and succeeding odd pulses are transmitted on the odd twisted pair; the even pulses are transmitted on the even pair. Since the nominal period for a single disc revolution is 52 ms, the interval between successive clock pulses is about 1.4 ~~1.5~~ μsec for the outer-zone control track, and about 2.4 ~~2.5~~ μsec for the inner-zone control track. The intervals between successive even write clocks (or successive odd write clocks) are twice the intervals stated.

(99B) WRITE CLOCK ODD (-TR:D5)

See WRITE CLOCK EVEN above.

These operations are enabled by the CM0(1) command-pair function rather than by the WRT level. (See CM0-1 above for a more detailed description of this command-pair function and the logical operations which it governs.)

Generated by:

$CM0(1) \wedge CM1(0)$

CHAPTER 9

ENGINEERING DRAWINGS

This chapter contains reduced copies of engineering logic diagrams, flow diagrams, circuit schematics, and other engineering drawings necessary for understanding and maintaining the Discfile Control 270. Only those drawings which are essential and are not available in the pertinent reference documents are included. For a list of all drawings in this manual refer to the table of contents.

A complete set of engineering drawings is supplied with the equipment. Should any discrepancy exist between the drawings in this chapter and those supplied, the assumption is that the latter drawings are correct.

9.1 DRAWING NUMBERS

Engineering drawing numbers contain five pieces of information, separated by hyphens. This information consists of a 2-letter code specifying the type of drawing; a 1-letter code specifying the size of the drawing; and variable-length codes specifying the type number of the equipment, the manufacturing series of the equipment, and a serial number for the drawing. The drawing type codes are:

BS, block schematic or logic diagram

CD, cable diagram

CS, circuit schematic

FD, flow diagram

ID, interconnection drawing

PW, power wiring

RS, replacement schematic

SD, system diagram

TD, timing diagram

TFD, timing and flow diagram

UML, utilization module list

WD, wiring diagram

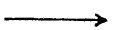
9.2 CIRCUIT SYMBOLS

The block schematics of Digital equipment are multipurpose drawings that combine signal flow, logical function, circuit type and location, wiring, and other pertinent information. Individual circuits are shown in block or semiblock form, using special symbols that define the circuit operation. These symbols are similar to those in the Digital System Modules Catalog and the FLIP CHIP Modules Catalog but are often simplified. Figure 9-1 illustrates most of the symbols used in Digital engineering drawings.

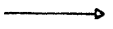
9.3 LOGIC SIGNAL SYMBOLS

A Digital logic signal symbol is shown at the input of almost all circuit symbols to specify the logic conditions that must be asserted to produce the output shown. These logic signal symbols are summarized in Figure 9-1.

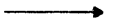
All logic signals are either standard Digital logic levels or standard Digital pulses. A standard Digital logic level is either a ground (0 to -0.3v)



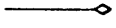
NON-STANDARD SIGNAL



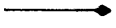
POSITIVE PULSE



NEGATIVE PULSE

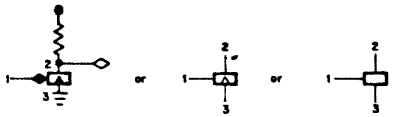


POSITIVE LEVEL

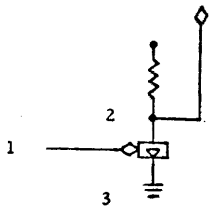


NEGATIVE LEVEL

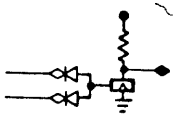
CLAMPED LOAD



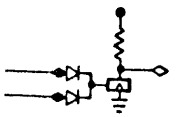
PNP TRANSISTOR INVERTER
1. BASE
2. COLLECTOR
3. EMITTER



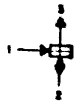
NPN TRANSISTOR EMITTER-FOLLOWER
1. BASE
2. EMITTER
3. COLLECTOR



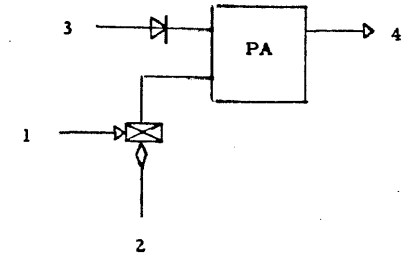
POSITIVE NAND, NEGATIVE NOR DIODE GATE



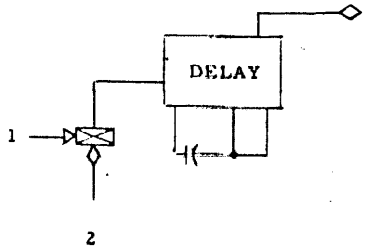
POSITIVE NOR, NEGATIVE NAND DIODE GATE



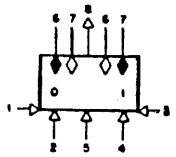
CAPACITOR-DIODE GATE, POSITIVE OR NEGATIVE INDICATED BY POLARITY OF THE INPUTS
1. PULSE INPUT
2. CONDITIONING LEVEL INPUT
3. PULSE OUTPUT



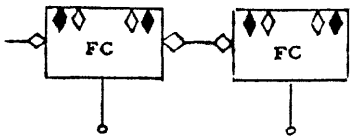
PULSE AMPLIFIER
WITH DCD GATE INPUT
1. PULSE INPUT
2. CONDITIONING LEVEL INPUT
3. DIRECT INPUT
4. PULSE OUTPUT



DELAY (ONE-SHOT MULTIVIBRATOR)
WITH DCD GATE INPUT
1. PULSE INPUT
2. CONDITIONING LEVEL INPUT
3. OUTPUT LEVEL, -3V DURING DELAY



FLIP-FLOP (MOST FLIP-FLOPS HAVE ONLY SOME OF THE FOLLOWING):
1. DIRECT-CLEAR INPUT
2. GATED-CLEAR INPUT
3. DIRECT-SET INPUT
4. GATED-SET INPUT
5. COMPLEMENT INPUT
6. OUTPUT LEVEL, -3 V IF 0, 0 V IF 1
7. OUTPUT LEVEL, 0 V IF 0, -3 V IF 1
8. CARRY PULSE OUTPUT



TWO OF THE EIGHT STAGES OF THE FILE-CONTROL OCTOFLOP.

NOTE - ALTHOUGH DRAWN WITH FLIP-FLOP SYMBOLS, THE OCTOFLOP IS ACTUALLY COMPOSED OF TWO R-284 QUADRAFLOP CIRCUITS.

FIGURE 9-1 DEC LOGIC SYMBOLS

or -3v (-2.5 to -3.5v). Logic signals are generally given mnemonic names (e.g. RED, WRT, END) which indicate the condition represented by assertion of the signal. An open diamond (——◇) indicates that the signal is a level and that ground represents assertion; a solid diamond (——◆) indicates that the signal is a level and that -3v represents assertion.

The standard Digital negative pulse is indicated by a solid triangle (——▶) and goes from ground to -3v. The standard Digital positive pulse, indicated by an open triangle (——▷) goes from -3v to ground. The width of the standard pulses used in this equipment is either 70, 100, 400 nsec or 1.0 μsec, depending on the module and application.

9.4 COORDINATE SYSTEM

Each engineering logic diagram is divided into 32 zones (4 horizontal, and 8 vertical) by marginal map coordinates. Figure references in the text are usually followed by a letter and a digit specifying the zone in which the referenced circuit is located.

Example:

Figure -FC:B4

zone B4

in Figure BS-D-270-FC

(To avoid needless repetition of the full drawing number most in-text references include only the difference modifier portion of the number.)

9.5 MODULE IDENTIFICATION

Two numbers appear in or near each circuit symbol or inside the dotted line surrounding multiple circuit symbols. The upper number designates the module type and is usually composed of a letter followed by three digits. Standard modules are identified by this number in the FLIP CHIP Modules Catalog. Nonstandard FLIP CHIP modules are prefixed by the letter G. The only nonstandard module included in the Discfile Control 270 is the G980 pulse amplifier. A circuit schematic of the G980 pulse amplifier is included in this chapter.

The lower number is the module location code. This number consists of a letter followed by two digits. The letter identifies the two-row mounting panel and row within the panel and the number identifies the module location within the row. The mounting panels are lettered alphabetically with panel A-B in the upper location. Within each row, the modules are numbered consecutively 1 to 32 from left to right.

Module connector terminals are identified by letters next to the circuit symbol. To identify any particular terminal, the terminal letter is added to the module location as a suffix. These letters run in alphabetical order, with the letters G, I, O and Q omitted. See Figure 9-2 for examples.

Example:

Figure 9-2 is a portion of engineering logic diagram BS-D-270-0-FC repeated here to illustrate Digital symbols and nomenclature.

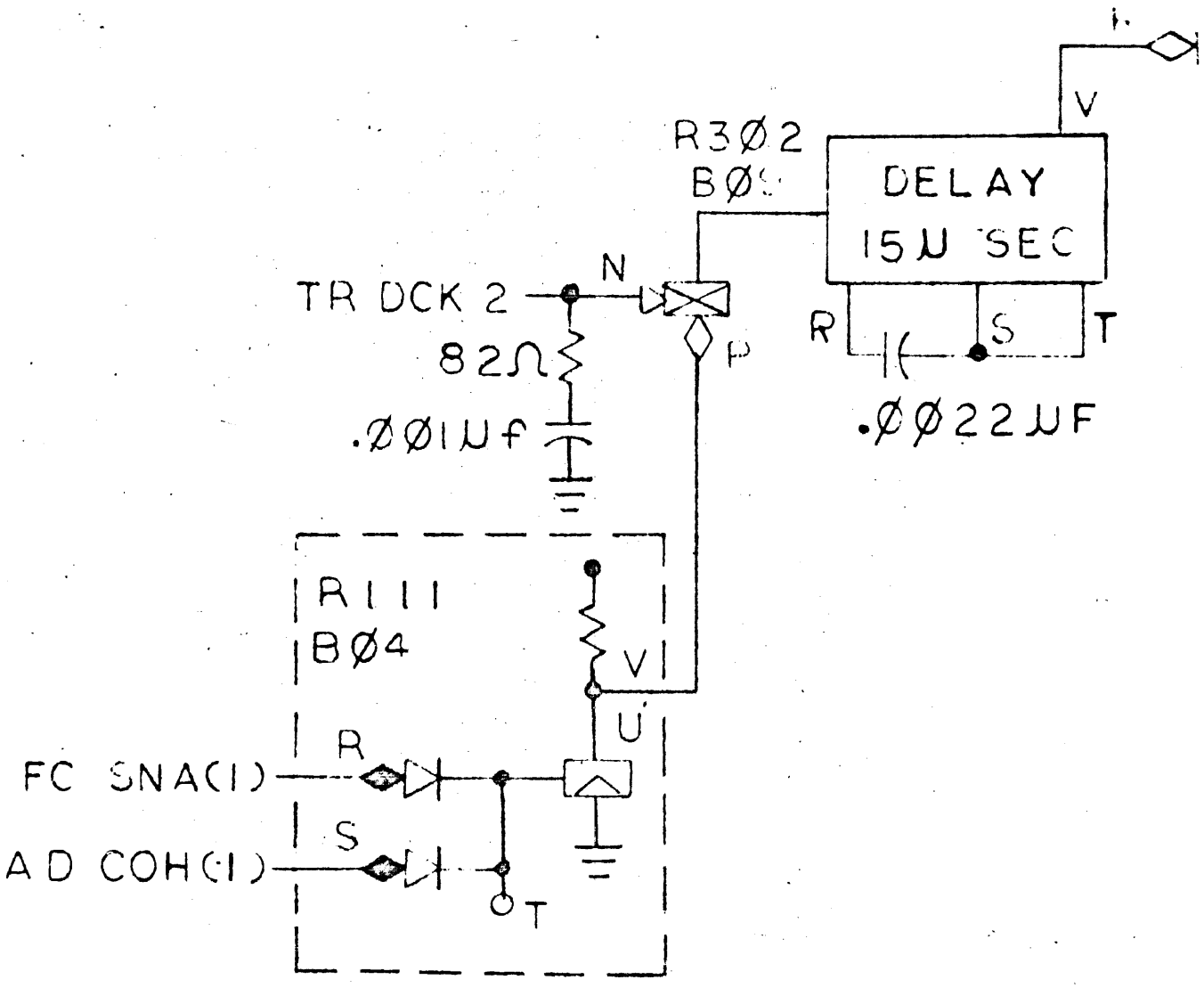


Figure 9-2

Typical DEC Engineering Logic Diagram

Portions of two separate modules are shown. (To determine what other circuits may also be contained in these two modules, consult the utilization module list drawing UML-D-270-0-11-sheet 1.) Both modules are located in Row B, the second row of modules in the top (A-B) mounting panel of the cabinet. The R302 one-shot delay is located in position B09 (ninth module from the left end of the row). The R111 diode gate is located in position B04 (fourth module from the left end of the row).

The two inputs to diode gate B04 are FC SNA(1) and AD COH(1). The two-letter prefixes FC and AD indicate that these signals originate on engineering logic diagrams BS-D-270-0-FC and BS-D-270-0-AD, respectively. As shown by the solid diamond, the SNA(1) level is asserted negative (-3 vdc) whenever the SNA flip-flop contains 1. Similarly, the COH(1) level is asserted at -3 vdc whenever the COH flip-flop contains 1.

If both the SNA(1) and the COH(1) input levels are asserted negative, then the R111 diode gate generates a ground-assertion output level. This level (shown by a hollow diamond) is applied to terminal P of the DCD gate at the input of the delay B09. When the level is asserted (ground), it enables the DCD gate to trigger the delay at the arrival of a positive DCK 2 pulse (shown by a hollow triangle). As indicated by the TR prefix, the TR DCK 2 pulse originates on engineering logic diagram BS-D-270-0-TR.

The output of the R302 delay is normally at ground. When the delay is triggered, the output drops from ground to -3 vdc, and remains at that negative level for the duration of the delay. When the delay period terminates, the output level returns to ground. (As shown in Figure 9-2, the

delay is calibrated to 15 μsec, but its duration can be varied by adjusting a potentiometer or changing the capacitor.)

digital EQUIPMENT CORPORATION
MAYNARD, MASSACHUSETTS

MASTER DRAWING LIST

CODE	DWG. NO.	REV. LET.	NO. OF SHEETS	TITLE
BS	D-270-0-AC	A	1	ADDRESS SHIFT COUNTER & CLOCK
BS	D-270-0-AD	B	1	ADDRESS HOLD & DATA ACCUMULATOR
BS	D-270-0-CM	B	1	COMMAND & CONTROL
BS	D-270-0-FC	B	1	FILE CONTROL
BS	D-270-0-SC	A	1	STATUS & CONTROL
BS	D-270-0-TR	B	1	TRANSMITTERS & RECEIVERS
BS	D-270-0-WC	A	1	WORD COUNTER
FPL	D-270-0-17		1	FRONT PANEL LAYOUT
CP	D-270-0-19		3	270 DISC FILE CHECKOUT PROCEDURE
FD	D-270-0-FD	A	3	FLOW DIAGRAM
FD	D-270-0-8		1	CONTROL STATES
UML	D-270-0-11	B	2	UTILIZATION MODULE LIST
WL	B-270-0-10	B	1	DISC FILE CONTROL WIRING LIST
PW	D-270-0-18		1	[illegible] DC POWER WIRING
WD	D-270-0-13	B	1	BUS SCHEDULE FOR 270 CONTROL
ID	A-270-0-14		7	270 CABLING COMPONENTS
CL	A-270-0-15	A	2	COMPONENTS LIST
RAD	C-270-0-9		1	RELAY TEST SWITCH PLATE
MD	B-270-0-16		1	DISC FILE NAME PLATE
MD	C-270-0-6		1	MASTER CLEAR SWITCH PLATE
SP	A-270-0-20		6	SYSTEM ACCEPTANCE TEST & PROCEDURE

ISSUED
APR 14 1966

REV.	ECO	ENG	DATE
A	166	S.A.L.	7/29/65
B	232	S.A.L.	11/22/65
C	255	AAY	2/17/66

MADE BY C. LaCorti 5/11/65 | CHECKER [illegible] | ENG [illegible]

TITLE 270 CONTROL

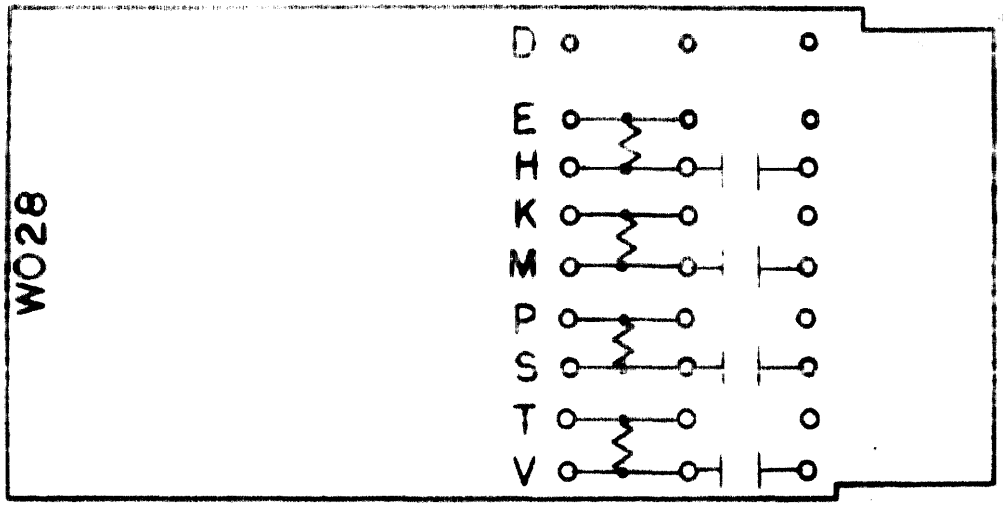
FOR THIS SCHEMATIC IS FURNISHED ONLY FOR TEST & MAINTAINANCE PURPOSES. THE CIRCUITS ARE PROPRIETARY IN NATURE & SHOULD BE TREATED ACCORDINGLY.

SHEET 1 OF 1 | CODE MDL | DWG. NO. A-270-0 | REV LET C

A-270-0-14 | CODE ID

W28	ELCO
D	
E	D4
H	E5
K	D14
M	E15
P	E1
S	D2
T	E11
V	D12

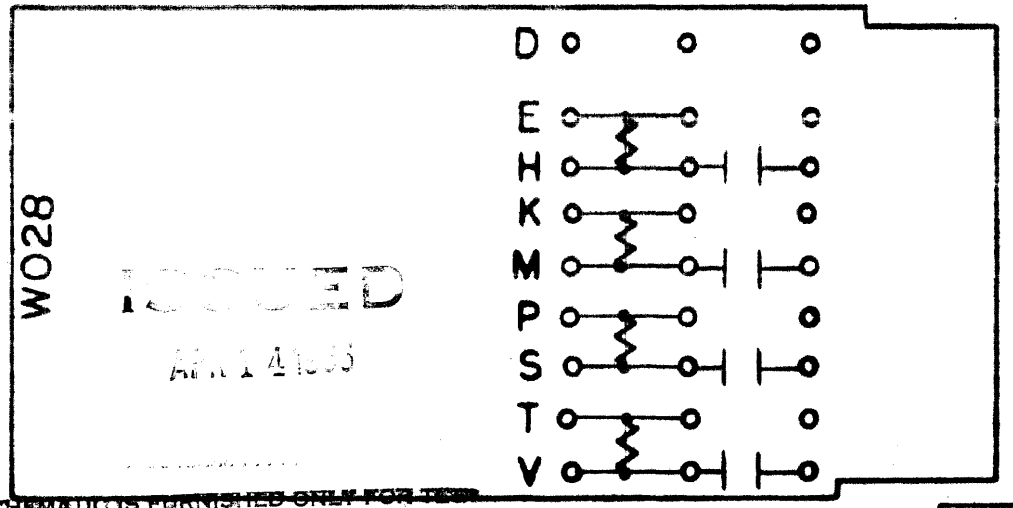
270
E31



CABLE TO DISCFILE MODEL 5022

D	
E	F2
H	F4
K	F6
M	F8
P	F10
S	F12
T	H2
V	H4

270
E32



ISSUED
APR 1 4 1983

Elco Connector #00-8017-100-000-012
Resistors are 100 OHM 1/4w
Capacitors are .001uf

CODE ID

SHEET 1 OF 7

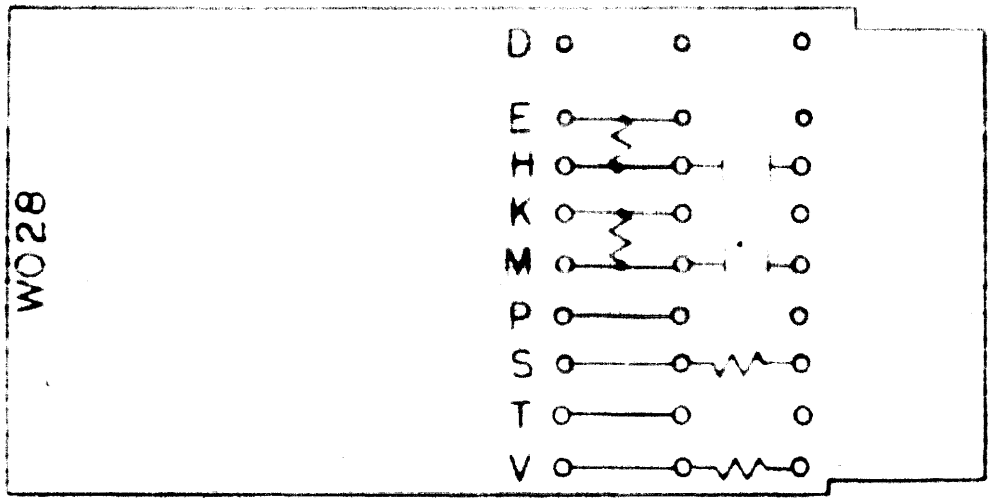
270 CABLING COMPONENTS

digital [redacted] A-270-0-14

A-270-0-14

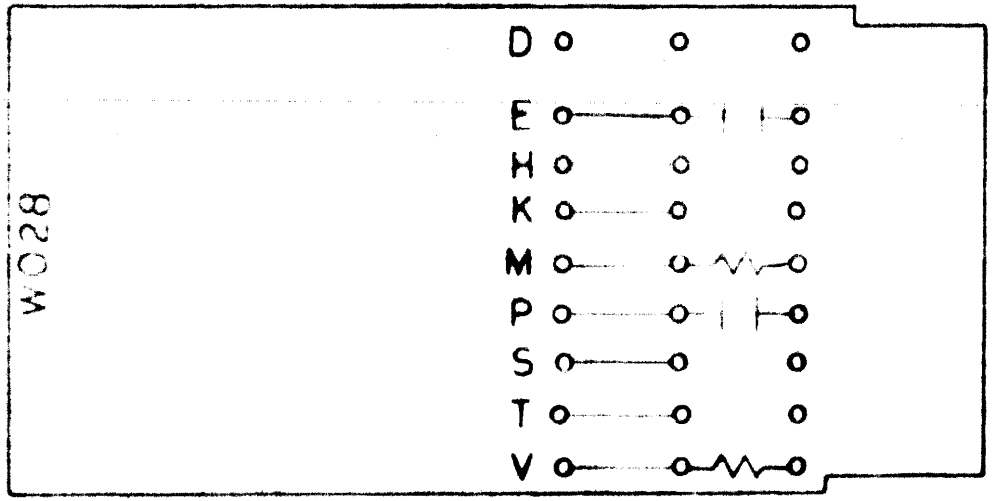
CODE 10

W028	ELCO
D	
E	B4
H	B2
K	C1
M	C3
P	C5
S	C7
T	C9
V	C11



CABLE TO DISCFILE MODEL 3022

W028	ELCO
D	
E	L1
H	L3
K	M2
M	M4
P	S3
S	S1
T	U9
V	U11



Elco Connector #00-8017-1[illegible]

Resistors are 100 OHM 1/4w

Capacitors are .001uf

CODE ID

SHEET 2 OF 7

APR 14 1975

drawn B. Lynch

checked

engineer

production

other

270 CABLING COMPONENTS

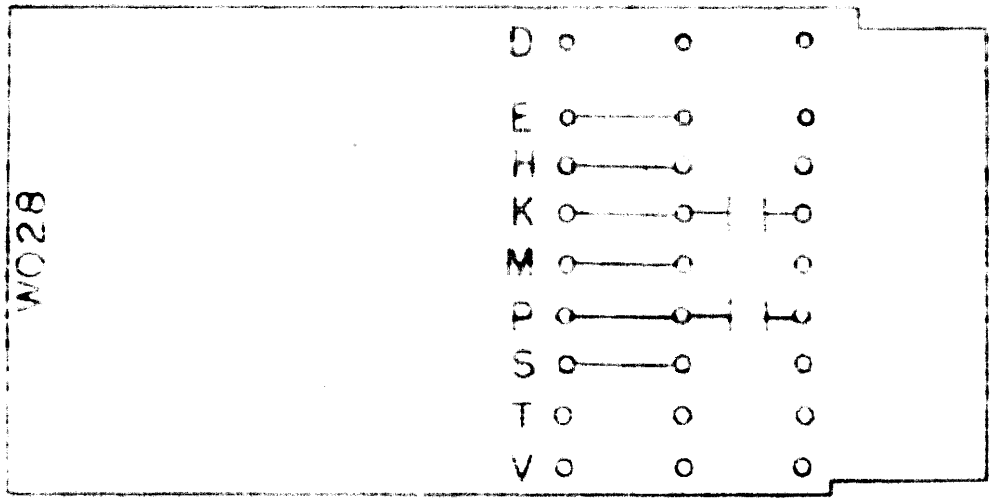
digital

A-270-0-14

A-270-0-14

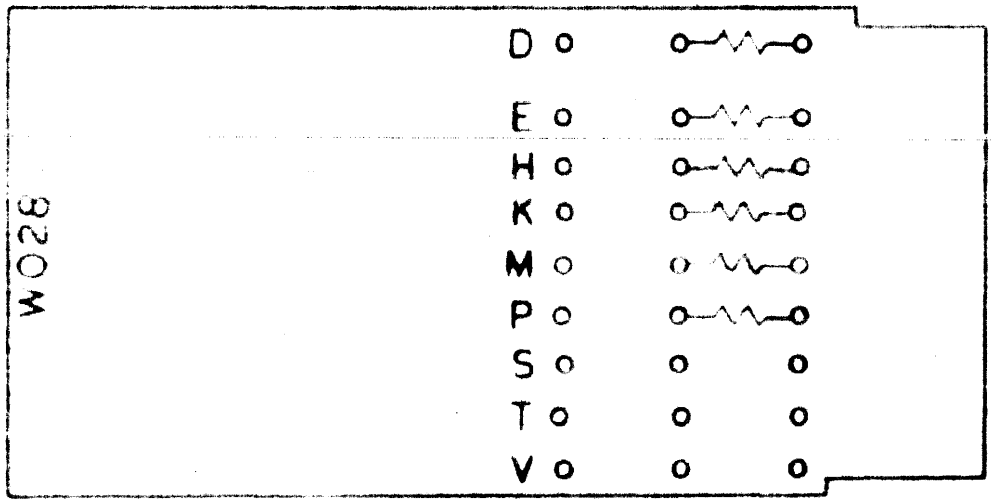
CODE
ID

W028	ELCO
D	
E	[illegible]
H	[illegible]
K	[illegible]
M	[illegible]
P	[illegible]
S	[illegible]



CABLE TO DISCRETE MODEL [illegible]

TERMINATED CABLE



Elco Connector #[illegible]
Resistors are 47 OHM [illegible]
Capacitors are .001uf

SHEET 3 OF 7

CODE
ID

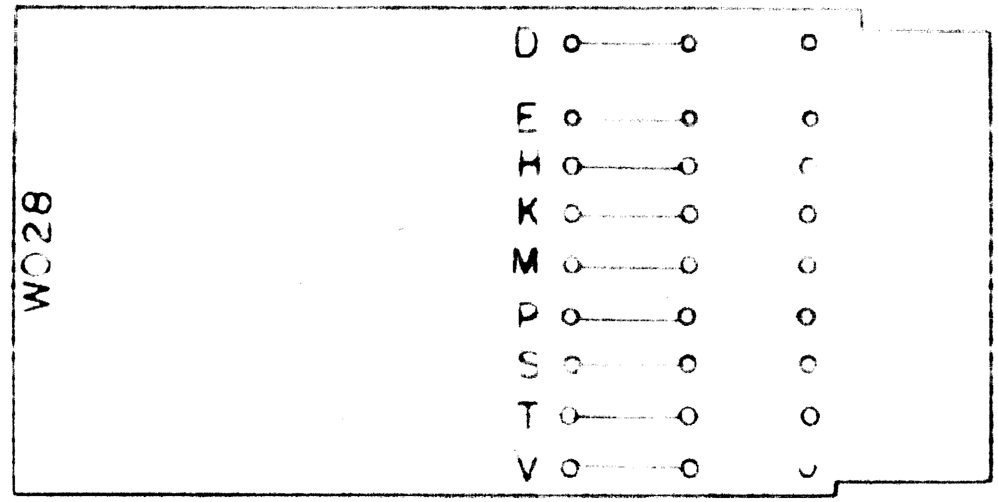
			exper.	drawn	270 CABLING COMPONENTS
			production	checked	
			other	engineer	digital

A-270-0-14

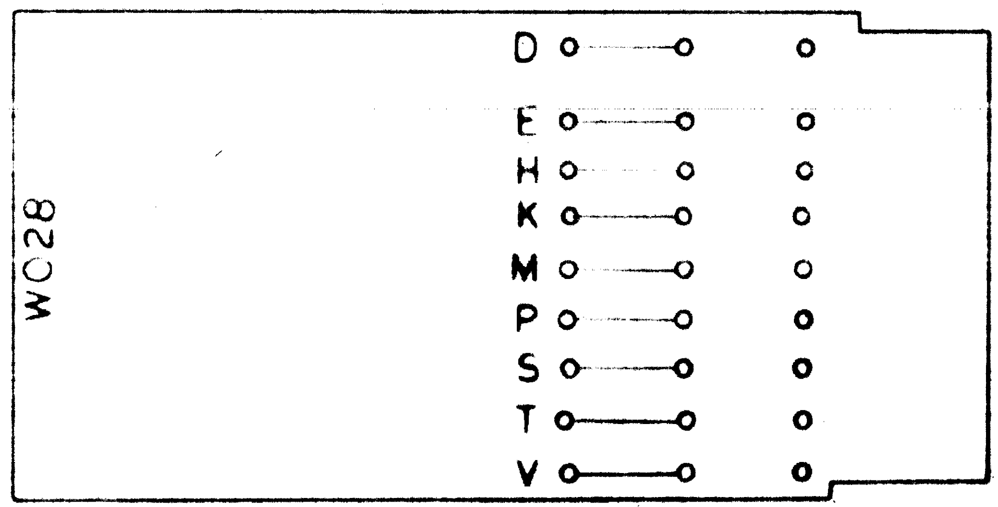
A-270 0 14

CODE ID

W028	1032A	E04
		E05
D	B	E06
E	C	E07
H	D	E08
K	E	E09
M	F	E10
P	H	E11
S	K	
T	L	
V	M	
D	N	
E	P	
H	R	27[illegible]
K	T	
M	U	F05
P	V	F06
S	W	F07
T	X	F08
V	Y	F09
		F10
		F11



[illegible]OB CABLE



SHEET 4 OF 7

CODE ID

ISSUED APR [illegible]

chg no. | date | app. | changes | engineer. | production | other

drawn B. [illegible] | checked [illegible] | engineer

270 CABLING COMPONENTS

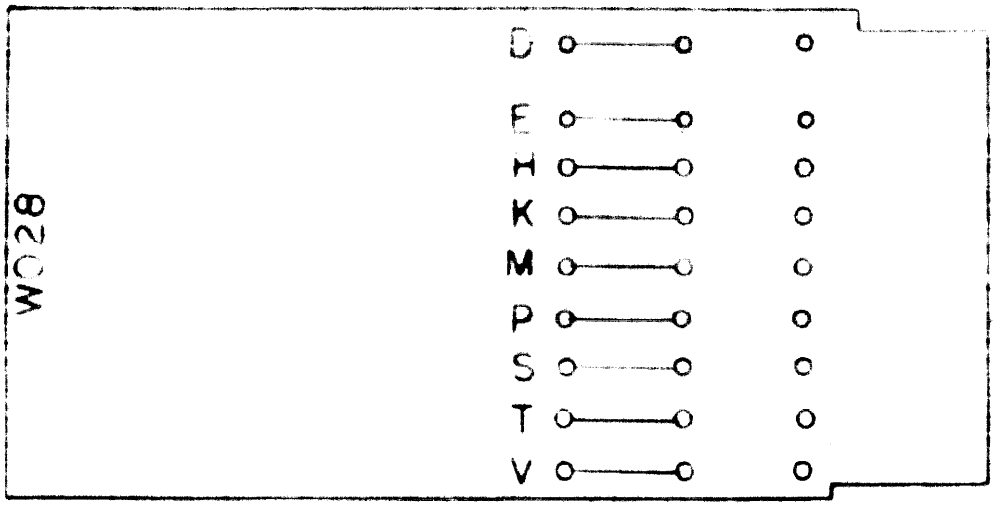
digital

A-270-0-14

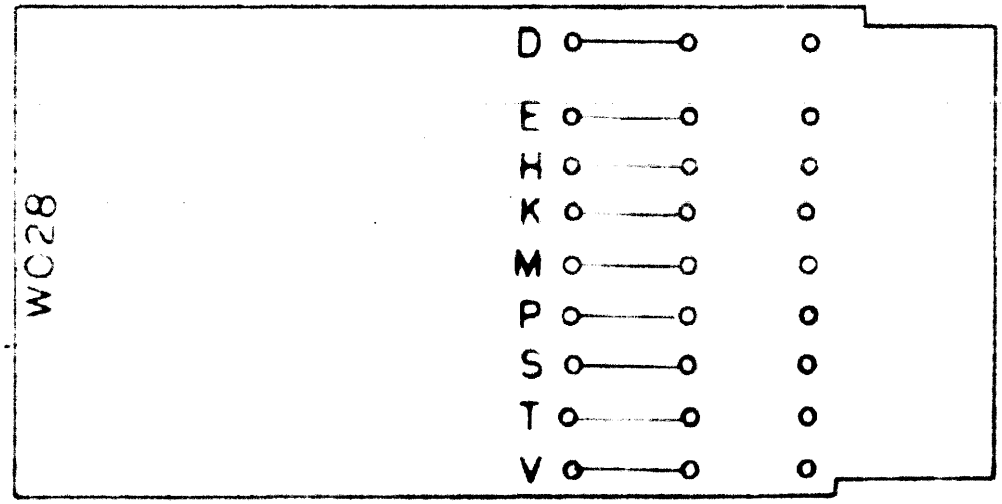
A-270-0-14

CODE ID

WØ28	1032A
D	B
E	C
H	D
K	E
M	F
P	H
S	K
T	L
V	M
D	N
E	P
H	R
K	T
M	U
P	V
S	W
T	X
V	Y



CABLE TO 136 DATA CONTROL



ISSUED

SHEET 5 OF 7

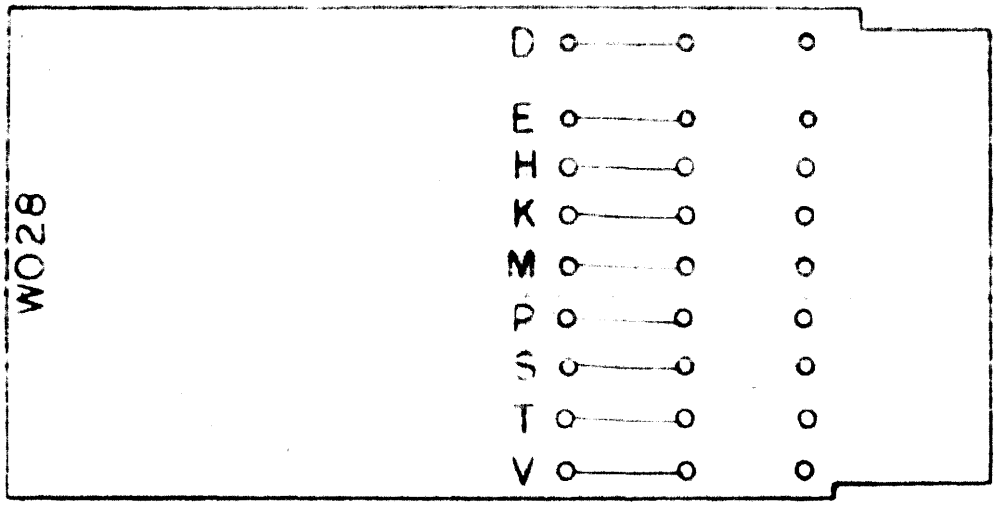
CODE ID

APR 14 1983	eco.	Changes	exper.	drawn B. Simpkin	270 CABLING COMPONENTS
	ECO		production	checked J. Whalen	digital
			other	engineer	A-270-0-14

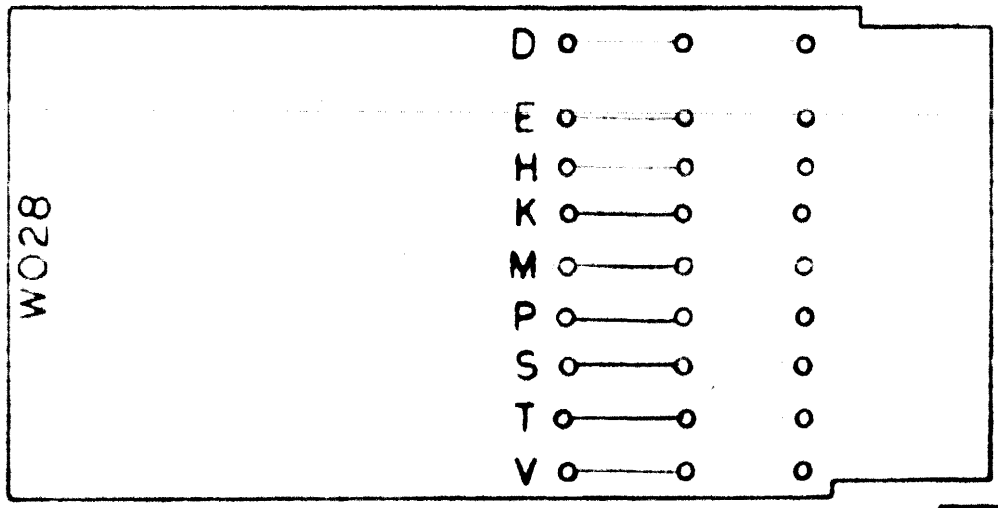
A-270-0-14

CODE
ID

W028	1032A		
D	B	136	270
E	C	1D17	F27
H	D		
K	E		
M	F		
P	H		
S	K		
T	L		
V	M		
D	N		
E	P		
H	R		
K	T		
M	U		
P	V	136	270
S	W	1D17	F27
T	X		
V	Y		



CABLE TO 13[illegible] DATA CONTROL



ISSUED APR

CODE
ID

SHEET 6 OF 7

ch. no.	Changes	exper.	drawn B. Lumpkin
app.		production	checked
		other	engineer

270 CABLING COMPONENTS

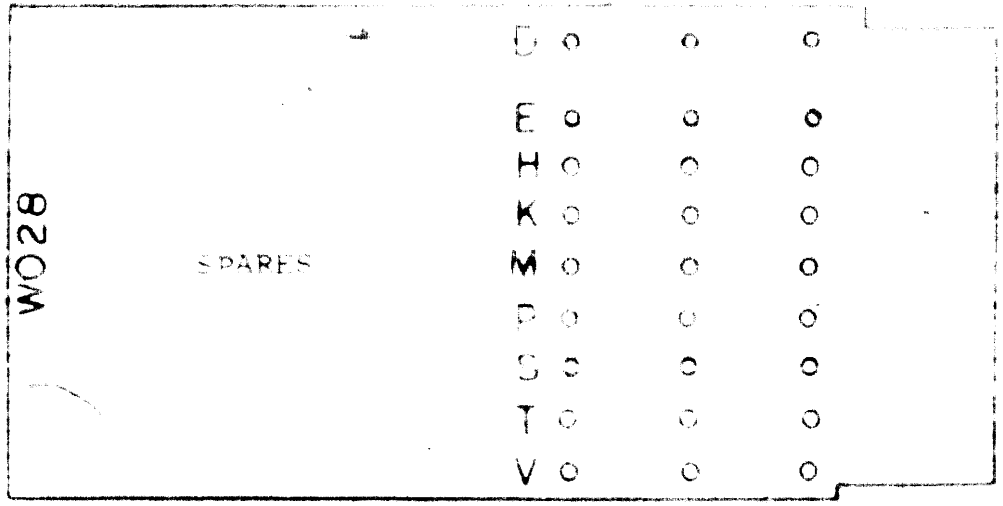
digital

A-270-0-14

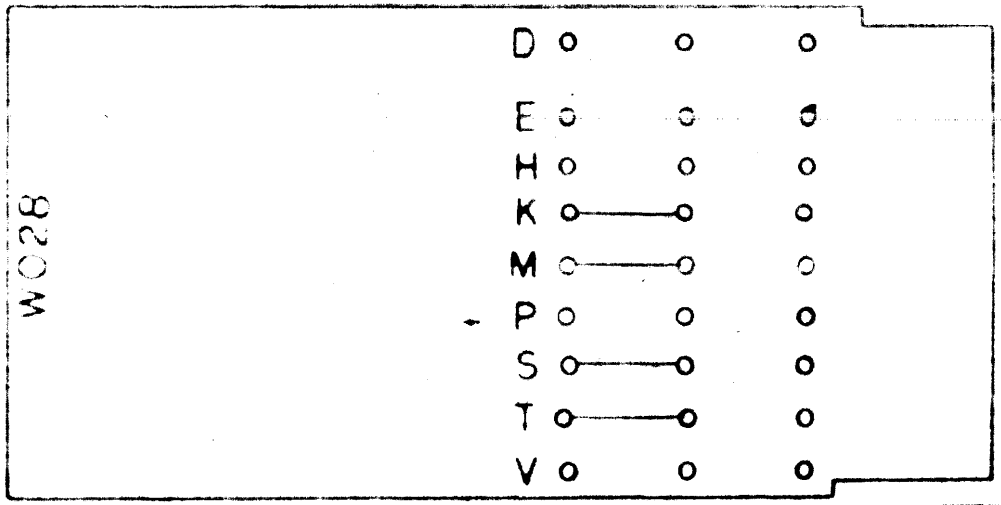
A-270-0-14 | CODE ID

W028 1032A

D	B	136	270
E	C	1D21	521
H	D		
K	E		
M	F		
P	H		
S	K		
T	L		
V	M		
D	N		
E	P		
H	R		
K	T		
M	U		
P	V		
S	W	136	270
T	X	1D20	F28
V	Y		



CABLE TO 136 DATA CONTROL



CODE ID

SHEET 7 OF 7

ISSUED APR 14

chg. no.	Changes	exper.	drawn B. Lumpkin	270 CABLING COMPONENTS	
app.		production	checked		
		other	engineer	digital	A-270-0-14

DIGITAL EQUIPMENT CORPORATION

MAYNARD, MASSACHUSETTS

ENGINEERING SPECIFICATION

DATE 11/22/65

TITLE 270 DISC FILE CHECKOUT PROCEDURE

REVISIONS

REV	DESCRIPTION	CHG NO	ORIG	DATE	APPD BY	DATE
-	ORIGNATED BY ECO 232		S.A.L.	11/22/65	S.A.L.	11/22/65

ISSUED

APR 14 1966

ENG	APPD	CODE	SIZE	NUMBER	REV
Steve Lambert	Steve Lambert	CP	A	270-0-19	

TITLE 270 DISC FILE CHECKOUT PROCEDURE

EQUIPMENT REQUIRED: PDP-6-16K COMPUTER
136 DATA CONTROL
270 DISC FILE CONTROL
5022 DATA PRODUCTS DISC FILE

NOTE: Remove all modules before starting checkout.
Remove all logic voltages when setting up logic for checkout.

1. Inspect for all obvious errors, such as broken wires, components, and other things of that nature. Record all defects, and rework in log book provided for that purpose.

2. Connect up IO Buss cables and logic voltages.

3. Place all M.C. switches to down position and turn on logic voltages via the PDP-6. Check for +10A and -15V B on logic panels A to F.

4. Place +10V A M.C. switches up. Vary +10 M.C. on PDP-6 and observe similar variations on logic.

5. Place +10V A M.C. switches down and -15V B M.C. switches up. Vary -15 M.C. on PDP-6 and observe similar variation on all logic panels A to F.

6. Place -15 M.C. switches down and turn off logic voltages. Install all modules except delays and clock in their respective positions as designated on UML-D-270-0-11. The W300 delay should be installed at this time.

7. Using a bench setup, adjust the delays and clock as indicated on the Block schematics or flow diagram before installing them in the 270 logic system. This can be accomplished by using a low rep-rate clock at the input to the delays and a scope for observing the duration. Any external components wired on the logic must also be used on the bench setup. Install the delays and clock after they have been adjusted.

8. Turn on power to the PDP-6 and 270 logic. Read in Maindec program #670 (Magnetic Disc Test) and follow the instructions set forth in the program write-up. The first test used in the checkout of the 270 is called DISK DPST. During this test, error typeouts will give an indicatioon as to which area of the logic is failing. The program will loop on the error so that a scope can be connected to the failing logic. When an error is found, be sure to turn power off before taking corrective action.

9. When an error condition has been corrected, turn power on and restart the DISK DFST program. If another error typeout appears, follow the procedure indicated in paragraph 8 and above until the DISK DFST program completes without any error typeouts.

10. The Magnetic Disc Test has many data transferring programs contained in one package. The first test to be tried after DISK DFST is to transfer* to the Disk and back "all ones." Once this has been accomplished "all zeros" should be transferred. If both these conditions work, then the worst case condition of random numbers (DISK DRAND) should be tried. If DISK DRAND works it can be assumed that the 270 logic is fully checkout.

11. Once DISK DRAND works, the next step is to take margin both +10V and -15V. These margins are to be entered on the data sheet provided with the 270 logic.

*Transfer "All ones" - "All zeros"

```
START   3500/ JSR INT          3510/ JSR INT
        3501/ GEN ONES         3511/ GEN ZEROS
        3502/ DISKW 0          3512/ DISKW 0
        3503/ GEN ZEROS        3513/ GEN ONES
        3504/ DISKR 0          3514/ DISKR 0
        3505/ CHECK ONES       3515/ CHECK ZEROS
        3506/ JRST DDT         3516/ JRST DDT
```

CONNECTION MEDIA	POL	PIN	PIN	NAME
		C24R 82Ω RES TO	.001uf CAP TO GND	AC CLK
		D18S		AC SDR
		D17F		AD CDA
		C09T		AD DTP
		E22N		AD FCL
		E22P		AD SDR
		F [illegible]K		CM CCB
		D25P		CM PTC
		E12T		CM STP
		C10K		
		B05R		
		F29P		FC ALP
		F20N		FC ESP 1
		D32E		FC [illegible]SP 2
		C21J		FC NDP 1
		E20N		FC NDP 2
		D17M		AD PTA
		F25P		SC IBR
		F25D		TR, DCK 1
		B09N 82Ω RES TO	.001uf CAP TO GND	TR DCK 2

DRAWN Maureen Mariano 5/25/65

CHECKED [signature] 5-27-65

ENG [signature] 5-28-65

digital EQUIPMENT CORPORATION MAYNARD, MASSACHUSETTS

EXTERNAL COMPONENTS LIST

TITLE
TYPE 270
COMPONENTS LIST

ISSUED
APR 14 1965

APPV SAL

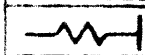
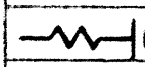
ECO. NO. REV. LTR. A-166

DWG NO A-270-0-15

REV. LTR. A

SHEET 1 OF 2

CODE CL

CONNECTION MEDIA	POL	PIN	PIN	NAME
		D21R 82Ω RES TO .001 uf CAP TO GND		TR DCK 3
		DØ2S 82Ω RES TO .001 uf CAP TO GND		TR RD 0
		D16R 82Ω RES TO .001 uf CAP TO GND		AD SMD
		BØ6R TO 0.15 uf CAP TO BØ6S		DELAY 0.2 MSEC
		BØ7H TO 0.15 uf CAP TO BØ7J		DELAY 1.0 MSEC
		BO7R TO .0022 uf CAP TO BØ7S		DELAY 15 USEC
		BØ9H TO .0022 uf CAP TO BØ9J		DELAY 15 USEC
		BØ9R TO .0022 uf CAP TO BØ9S		DELAY 15 USEC

DRAWN Maureen Mariano 5/25/65
CHECKED RH Watts 5-25-65
ENG

digital
EQUIPMENT CORPORATION
MAYNARD, MASSACHUSETTS

EXTERNAL COMPONENTS LIST

TITLE
TYPE 270
COMPONENTS LIST

APP'V
ECO. NO. REV. LTR. A-166

DWG NO A-270-0-15
REV. LTR. A

SHEET 2 OF 2
CODE CL

DIGITAL EQUIPMENT CORPORATION

MAYNARD, MASSACHUSETTS

ENGINEERING SPECIFICATION

DATE 1/12/66

TITLE TYPE 270 DISC FILE SYSTEM ACCEPTANCE TEST AND PROCEDURE

REVISIONS

REV	DESCRIPTION	CHG NO	ORIG	DATE	APPD BY	DATE

ISSUED

APR 14 1966

ENG	APPD	CODE	SIZE	NUMBER	REV
			A	270-0-20	

PURPOSE OF TEST

The purpose of the test described herein is to assure that the performance of the system meets the criteria for acceptance set forth. The testing procedure includes individual component tests as well as complete system tests. When all conditions of the tests have been satisfied, Digital Equipment Corporation considers the system acceptable. Customer acceptance of the Type 270 Disc File System is to be based on the performance of the system as indicated by the test programs supplied by DEC personnel.

TESTS PERFORMED

DISK DFST (Disc File Status Test)

Individual tests are performed on all parts of the Type 270 System including the portion of the 136 Data Control connected to the Disk Control. Error typeouts will appear if any logic component fails. This test requires approximately 10 minutes run time.

DISK DADDR (Disc Address Test)

The Disc Address test is performed on every Disc in the following manner. Each position on the specified positioner is Written, Read and Checked. When all portions have been written, the program will re-read and check all the data on the positioner. The pattern used is such, that each word of a sector will contain the Disc, Position and Address of that sector. This test insures that all locations exist. There are 3072 address operations performed, 2.1×10^8 bits* written, 4.3×10^8 bits * read, during this test. The program requires 39 minutes operation time.

DISK DRAND (Random Disc Address)

This program selects at random a Disc Position, and sector whereupon, random numbers are Written, Read-Compared, Read and Checked for accuracy over one revolution or 44 sectors. 1×10^3 operations are performed in random sequence. 2.1×10^8 bits are written and 4.2×10^8 bits are read. This test consumes 30 minutes of machine time.

*Bits read or written include parity bits.

DISK DFZO (Floating Zero/One Test)

This test is performed using a pattern of floating zeros and ones on alternate positions throughout the 16 Discs. A floating zero consists of a 36 bit word of all ones except one bit which is zero. This bit changes position on each successive word. A floating one is the complement of a floating zero. The program will Write, Read and Check all words at a single position. The position is then decremented and the procedure is repeated using the complement pattern. When all positions on a positioner have been written, the data is Read, Checked and the position is incremented. When checking is complete, the above process is repeated using the complement of the above pattern. The total number of operations executed is 1024. There are 4.3×10^8 bits written and 8.6×10^8 bits read during this test. Program run time is 1 hour 4 minutes.

DISK DOXIDE (Surface Oxide Test)

This test is identical to the DISK DFZO test except the pattern is alternate ones and zeros 5252 etc, 2525 etc. The program run time is 1 hour. The bits transferred are the same as DFZO.

Acceptance Program - Maindec(---):

The program is initiated by starting at address 3500. The sequence with run time and bits transferred is:

Comment	Instruction	Run Time	Bits Read	Bits Written
Start	JSR INT	None	None	None
	DISK DFST	10 min.	None	None
	DISK DADDR	39 min.	4.3×10^8	2.1×10^8
Repeat 10X	DISK DRAND	5 hrs.	4.2×10^9	2.1×10^9
Repeat 2X	DISK DFZO	2 hrs. 8 min.	1.72×10^9	8.6×10^8
	DISK DFZO	2 hrs. 8 min.	1.72×10^9	8.6×10^8
	DISK DOXIDE	2 hrs. 0 min.	1.72×10^9	8.6×10^8
	DISK DOXIDE	2 hrs. 0 min.	1.72×10^9	8.6×10^8
	TOTAL	14 hrs. 5 min.	1.151×10^{10}	5.75×10^9

1.9×10^4 operations during test.

Equipment Life

The file has a design life of at least three (3) years before major overhaul is required, assuming that proper scheduled maintenance is performed.

Error Warranty

There should be no errors due to failure of the magnetic recording medium (bad spots) for the warrantable life of the equipment assuming that recommended maintenance procedures are followed. In addition, the equipment shall not exhibit any magnetic degradation with time of pre-written data sufficient to cause errors while reading.

Acceptance Error Rates

The following paragraphs define the acceptance error rates of the DISCFILE System.

Malfuntion Definitions

Reading Error

A data error is detected and three repeated reading operations initiated under program control, are error-free.

Writing Error

A data error is detected as the result of a write check* and three repeated writing operations, followed by write checks initiated under program control are error-free.

Incomplete Operation Error

A malfunction of the equipment occurs during which there are no data error. Under program control, the operation is repeated and is completed successfully.

Failure

A condition which causes errors or other malfunctions which can only be corrected by unscheduled maintenance.

Acceptance Test Error Rates

Following are the acceptance test criteria for the final acceptance test performed by Digital Equipment Corporation personnel:

Incomplete Operation Error Rate

Not more than 1 in 2×10^4 operations.

*The first operation of reading and checking data written on a previous revolution of the disc.

Read Error Rate

Not more than 1 in 2×10^9 data bits transferred.

Write Error Rate

Not more than 1 in 10^{10} data bits transferred.

Failure Rate

Four maximum within 250 hours of cumulative file operation.

Storage Capacity

Words per sector	128
Sectors per position	x 44
Words per position	5,632
Bits per word (including parity)	x 37
Bits per position	208,384
Positions per Disc	x 64
Bits per Disc	13,336,576
Discs per file	x 16
Bits per file	213,385,216

Selection Time

The time to switch from one Disc and Position to another and receive a ready signal from the newly selected sector, should not exceed 351 milliseconds (worst case).

While the DISK DRAND test is operating, set bit 34 of the Console Data Switches to a one. All selection times will be printed on the Type 646 Line Printer (if one is available) in the following format:

Previous Address		Present Address	Switching Time
1550	>	1551	122.4 M
Disc Position		Disc Position	milli-seconds

The acceptance test program must be run from beginning to end (14 hours 5 minutes consecutively) with no more than five Read or Read Check, one Write and one Incomplete Operation error typeout.

Signatures by a DEC Field Engineer and the Customer constitutes acceptance of the Type 270 Disc File System.

Signed _____

DEC Field Engineer

Signed _____

Customer

DRWG NO
B-27Ø-Ø-1Ø

REV LTR
B

REV LTR	ECO NO	DATE	ENG
A	166	12-29-65	S.A.L
B	232	1/-22-66	S.A.L

REVISIONS

THIS SCHEMATIC IS FURNISHED ONLY FOR TEST, MAINTAINANCE PURPOSES. THE CIRCUITS ARE PROPRIETARY IN NATURE & SHOULD BE TREATED ACCORDINGLY.

ISSUED
APR 14 1966

DRAWN	DATE
CHECKED R H WATSON	DATE 5-28-65
ENG S. A. Lambert	DATE 5-27-65
PROJ ENG S. A. Lambert	DATE 5-27-65
PROD S. A. Lambert	DATE 5-27-65

digital
EQUIPMENT
CORPORATION
MAYNARD, MASSACHUSETTS

TITLE
DISC FILE CONTROL WIRING LIST

TAPE # FILE #

ASSY NO	CODE
	WL

SHEET 1 OF 1

DRWG NO	REV. LTR
B-27Ø-Ø-1Ø	B

DIGITAL EQUIPMENT CORPORATION · MAYNARD, MASSACHUSETTS

5541 PRINTED IN U.S.A. .1-4/66

MAGNETIC TAPE CONTROL 516 INSTRUCTION MANUAL VOLUME 1

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

COPY NO.

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

CONTENTS (continued)

CONTENTS (continued)

CONTENTS (continued)

CONTENTS (continued)

ILLUSTRATIONS

TABLES

CHAPTER 1

INTRODUCTION

PURPOSE AND SCOPE

The purpose of this instruction manual is to aid personnel in the installation, operation, and maintenance of the DEC Magnetic Tape Control 516 and the associated Interface Unit Type 520, 521, or 522A (Figure 1-1).

MANUAL ORGANIZATION

Volume I presents a brief description of the system application and specifications in Chapter 1, Introduction. Chapter 2, Tape Control Inputs and Outputs, explains the principal signals exchanged between the tape control and the adjacent units of the PDP-6 system. The next four chapters, Chapters 3 through 6, contain the theory of operation of the tape control logic itself. Chapters 7 through 9 describe the theory of operation for the three types of interface unit that may be used with the tape control, the 520, 521, and 522A, respectively. Chapter 10, Maintenance, contains information useful for inspection, troubleshooting, and repair. Chapter 11, Installation, explains the procedures to be followed when first unpacking and installing the tape control and interface.

Volume II presents complete signal glossary lists in Chapter 12 for the Type 516 Tape Control, and Types 520, 521, and 522A Tape Control Interface Units. Chapter 13 contains introductory information on DEC drawing conventions and all pertinent engineering drawings for this manual.

Figures

This manual includes four general classes of figures: engineering logic drawings; flow diagrams; circuit schematics; and miscellaneous figures such as photographs, block diagrams, and logic rack layout diagrams. The complete system logic for the tape control is shown in the engineering logic diagrams referenced in Chapter 3 and contained in Volume II. The interface logic is shown in the engineering logic diagrams referenced in Chapters 7, 8, and 9. Flow

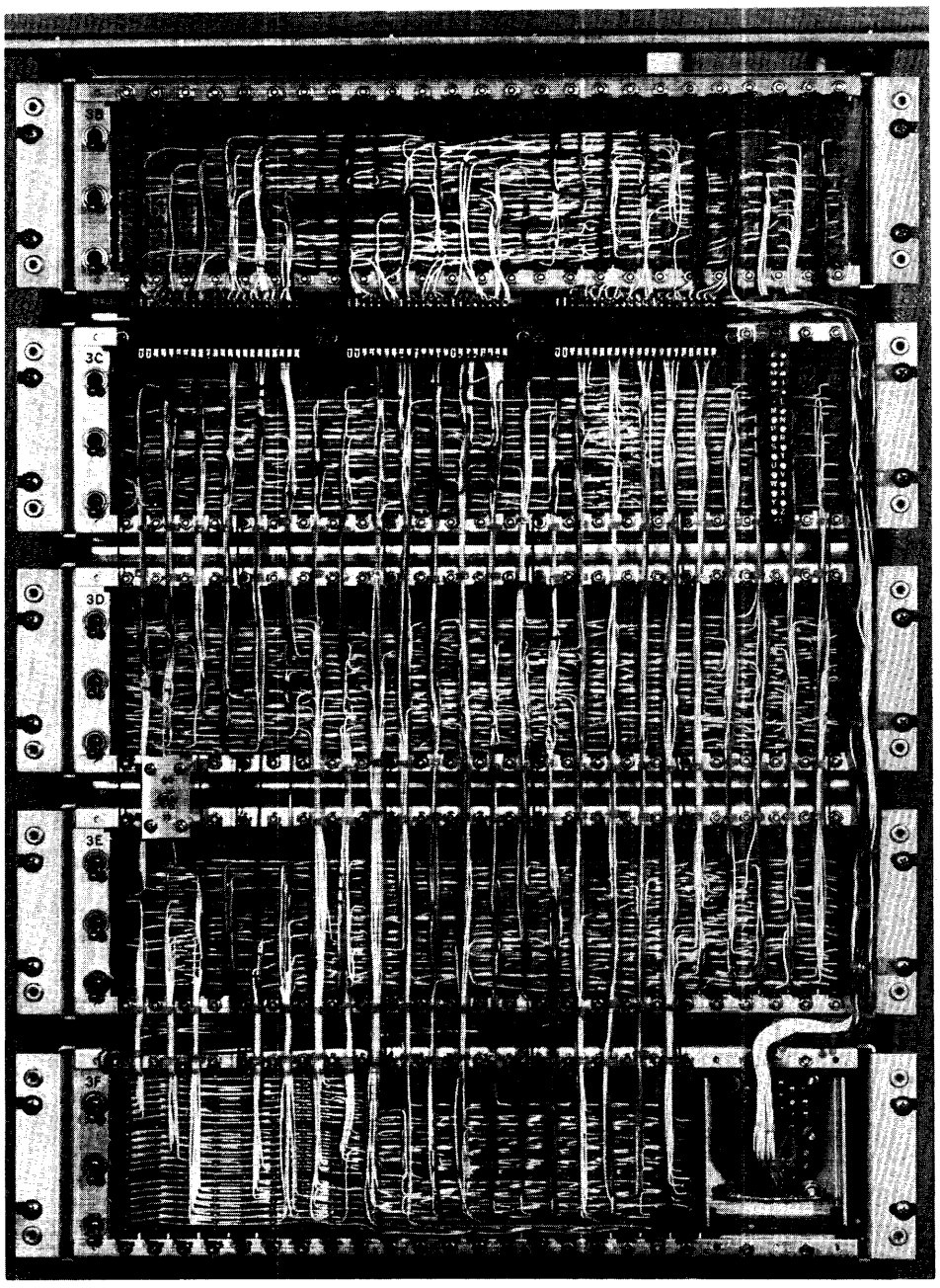


Figure 1-1 Magnetic Tape Control Type 516
and Tape Control Interface Type 521

diagrams of the various system operations are presented in text with Chapters 3 through 6. Circuit schematics and logic rack layout diagrams are contained in Chapter 13.

At the date of issue of this manual, up-to-date engineering drawings are included in Chapter 13. However, in time engineering changes do occur, thus if any discrepency exists between the engineering drawings of this manual and a current set of engineering drawings, the current engineering drawing set is correct.

All engineering logic drawings in Chapter 13 are arranged numerically for quick reference. Furthermore, all circuit schematics of Chapter 13 are arranged in order of circuit type designation (e.g., Negative Diode Module 4114 precedes Diode Module 4115).

USE OF MANUAL

This manual is intended to serve two purposes: instruction and reference. Chapters 2 through 9 on theory of operation are intended primarily for instruction. These chapters cover the major logical networks and operating sequences of the system in a systematic and detailed manner, but they are not organized for rapid reference. After being read, they rarely need be referred to again.

Once they have been thoroughly understood, the engineering logic drawings and the flow diagrams provide the fastest and most extensive source of reference data. After a reasonable learning period a competent maintenance engineer can orient logic function with particular drawings and even the approximate area of the drawing which shows given sections of the logic.

The flow diagrams add a time dimension to the logic drawings by providing sequential summaries of the specific operations performed during each type of command. Further reference data is provided in Chapter 12, which contains signal glossaries for the tape control and the three interface units.

SYSTEM APPLICATION

The magnetic tape system provides computer input-output at much faster data rates than most other peripheral devices. It is frequently used as high-capacity storage to augment core memory. Figure 1-2 is a block diagram showing typical tape system configurations. A given system

uses one of three types of interface units; the choice depends on the type of tape transport to be driven. Up to eight transports can be used by each interface-control combination. During both output and input operations, the Magnetic Tape Control 516 receives control information from the processor and generates appropriate signals to cause the interface and selected transport to correctly execute the programmed command.

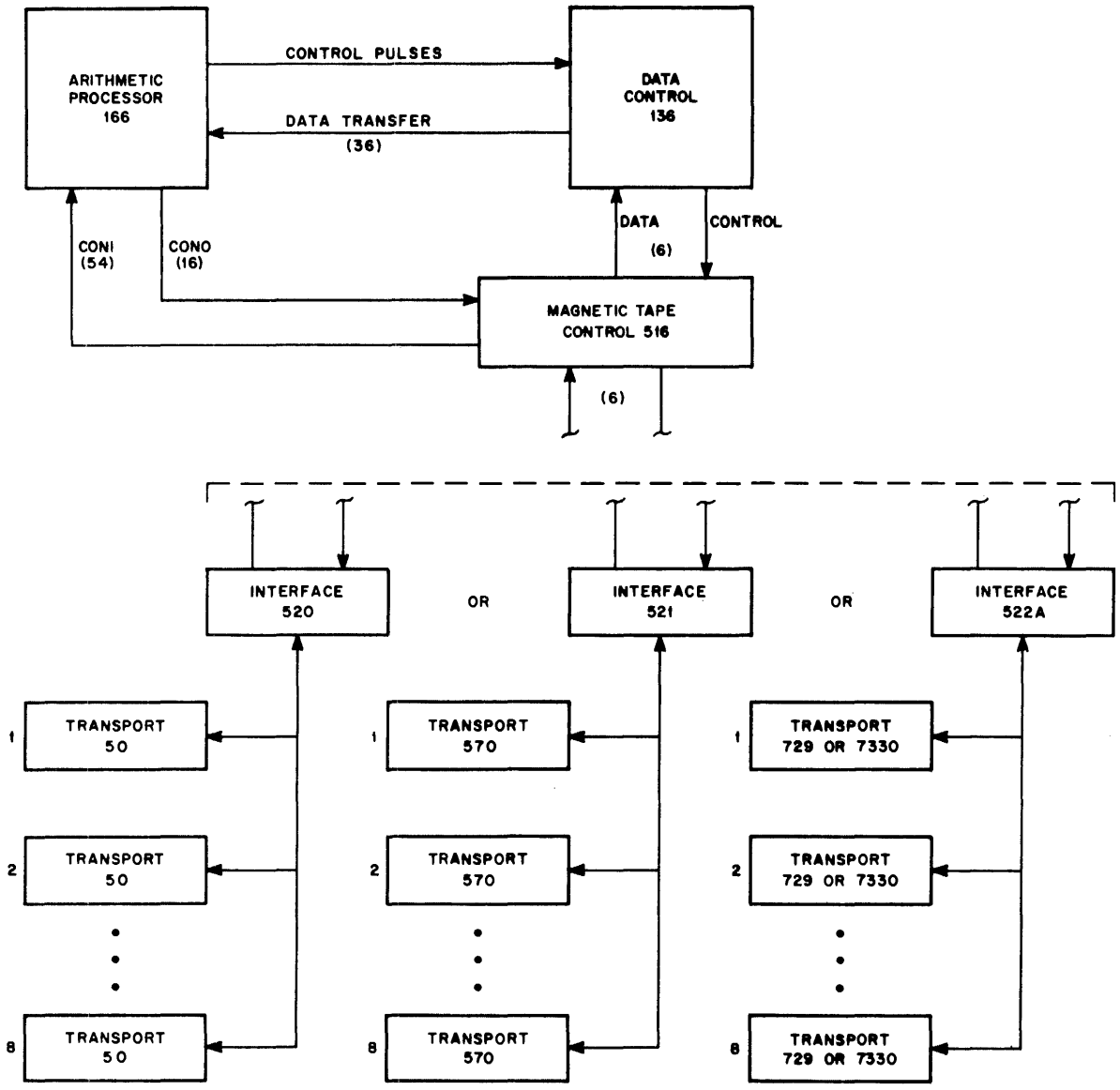


Figure 1-2 Tape Systems Configurations

For output, 36-bit computer words are transferred from the PDP-6 processor to the Data Control 136. The data control, through six sequential shifts out of the high-order end of its

accumulator, sends the six 6-bit characters (converted from each computer word) to the tape control to be written on tape. The tape control applies the output data characters to the write logic of the selected tape transport via the transport interface.

For input, the sequence is reversed. Data read from the tape in 6-bit characters is sent to the data control via the interface and the tape control. The data control shifts the data into its accumulator until six characters have been assembled into a 36-bit computer word (the end of the accumulator chosen and the direction of the shift depend upon the direction of tape motion, i.e., upon whether the command is read or backward read). The data control then transfers the full 36-bit word into the processor.

PROGRAMMED OPERATIONS

Both CONI and CONO tape control instructions may be addressed to the Magnetic Tape Control 516.

CONI Instruction

The processor samples 54 tape control status bits by means of CONI instructions. Three separate device selection addresses are provided; each samples 18 status bits. This permits tape control and transport status information to be sampled directly from the processor control console.

CONO Instruction

The processor sends 16 bits of control information to the tape control at each CONO 220 tape control instruction. (The tape control also receives CONO 224 and CONO 230 instructions for more specialized functions described later in this chapter under Parity and Priority Interrupt.) The functions controlled by this information include:

1. the choice of the command to be executed (read, write, write file, etc.)

2. the selection of the tape transport that is to carry out the command

3. whether or not the selected transport is to be kept reserved after completing the command

4. the parity mode (binary or BCD)

5. the density at which data is to be written on tape (200, 556, or 800 bits per inch)

6. the slice level at which data is to be read from tape

7. whether or not end of record gaps are to be ignored

8. the priority interrupt channel to be assigned to the tape control.

The functions determined by the 12 more significant CONO 220 bits are listed in Table 3-1 (CB Register Bit Assignment). The remaining four CONO bits are never transferred into the CB register, but they make up two of the control fields listed above. Bit 32 is the maintenance bit; when programmed to be a 1, it causes the tape control to ignore the record gap and treat the entire tape as a single record. Bits 33-35 assign the tape control priority interrupt channel.

Command Repertoire

By suitably coded CONO 220 instructions, the processor can cause the Magnetic Tape Control 516 to execute any of the following ten commands.

Rewind	The selected transport rewinds the tape to load point and stops.
Rewind Unload	Rewinds the tape off the takeup reel of the selected transport.
Write	Writes N words of six characters each (from consecutive or nonconsecutive memory locations) into one tape record. A lateral parity bit (binary or BCD) is generated for each character written. All characters are read and checked for parity after writing. At the end of the record, an EOR gap is spaced out, and an EOR mark (longitudinal parity check character) is written.

Write End of File	Writes an end-of-file record consisting of an end-of-file character (17 octal, BCD) followed by an EOR mark. The tape control does this automatically without use of the Data Control 136.
Write Blank Tape	Writes about 3 inches of blank tape and stops. The Data Control 136 is not used.
Read Compare	Reads a full record from tape and compares it character by character with output data from the processor memory. Any discrepancy sets the RCE flag. (The EOR mark that follows the tape record is not compared.)
Read	Reads one complete record from tape. Each character is shifted into the low-order end of the data accumulator in the Data Control 136 for assembly into 36-bit computer words and transferred into the PDP-6 processor. Although the tape control always reads a full record, the processor can be programmed to read in any lesser amount of data.
Read Backward	Identical to read except for the direction of tape motion. Because data is shifted into the high-order end of the data control accumulator, the ordering of characters within each 6-character computer word is the same as for read. The word order is, of course, still inverted.
Space Forward	All spacing is done without the use of the Data Control 136. To space forward one record, the tape control reads the record, but does not apply the resulting data to the data control. If a 1 is programmed in bit 24, the space forward command is iterated until a file mark record is reached.

Space Backward	To space backward one record, the tape control executes a backward read but does not apply the resulting data to the data control. If a 1 is programmed in bit 24, the space backward command is iterated until a file mark record is reached.

Parity

Both lateral and longitudinal parity are checked during all commands except rewind and read compare. Separate error flags are provided for the two types of parity error. The PER flag indicates lateral parity errors, and the LPE flag indicates longitudinal parity errors. Longitudinal parity is always even; lateral parity can be programmed to be odd or even (binary or BCD).

Priority Interrupt

Seven priority interrupt channels are available. See Chapter 2, Priority Interrupt, for a description of interrupt conditions.

TAPE FORMAT

All transports driven by the Magnetic Tape Control 516 use 1/2-inch tape containing seven information channels. The tape format is shown in Figure 1-3. The left portion of the figure shows the tape in relation to the read and write heads. The tape moves by the heads vertically – forward direction being downward. The tape is composed of a mylar base coated on one side with an iron oxide composition. The oxide or dull side of the tape faces the heads, with the left edge toward the transport drive plate. The recording density may be either 200, 556, or 800 characters per inch. Tape speed varies depending upon the type of transport used.

The method of recording used is nonreturn-to-zero (NRZI). Although the tape has two basic states of remnant magnetization, the remaining magnetic state of the tape at a given bit position does not determine the value of that bit. A logical 1 is represented by a change from one state of magnetization to the other, in either direction. A logical 0 is represented by a constant state of magnetization. Therefore, to write a series of characters containing all 0s

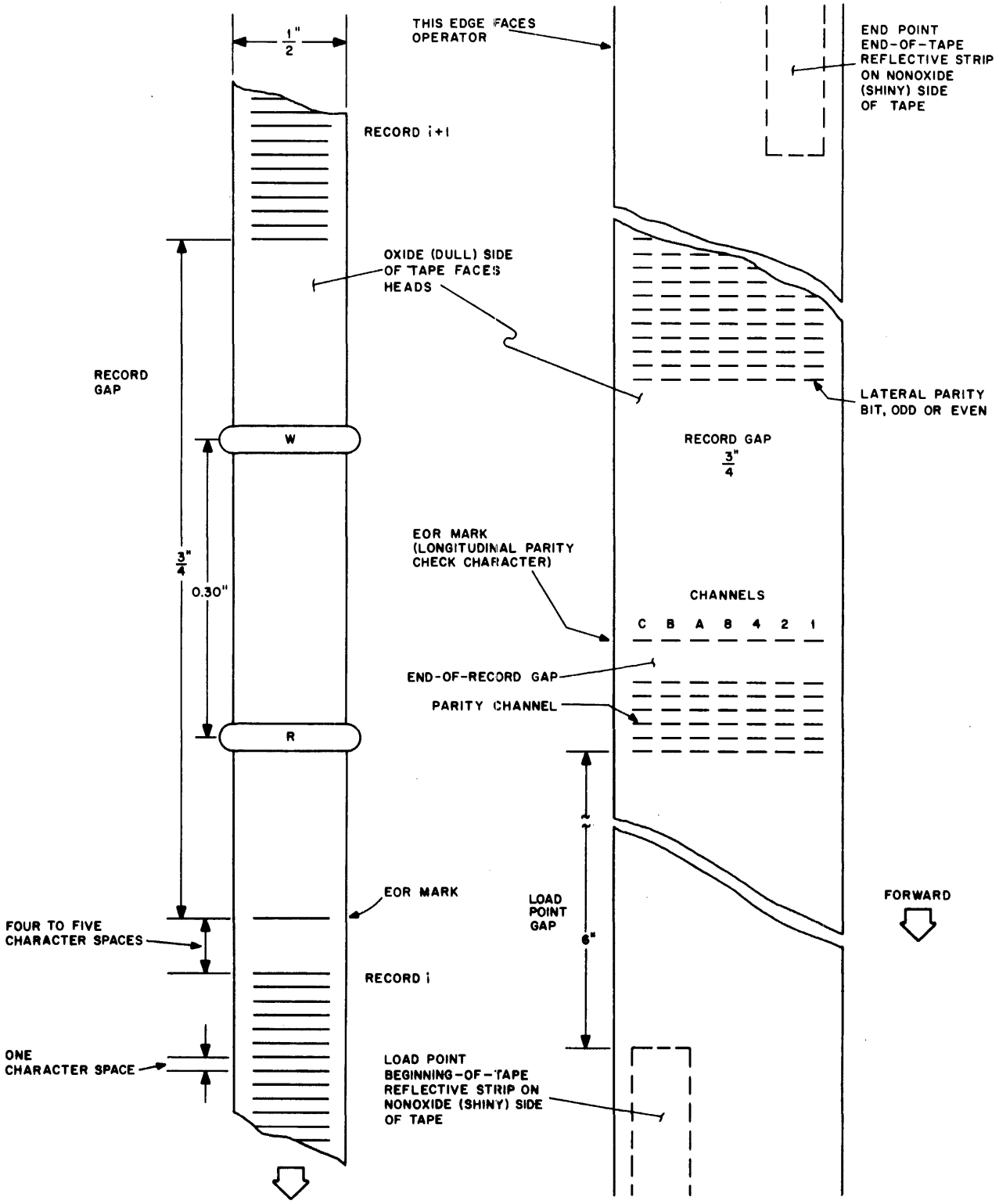


Figure 1-3 Tape Format

is equivalent to writing a section of blank tape. Each time a character is transferred into the write buffer, the NRZI writers produce an equivalent character on the tape. However, because of NRZI method of recording, a transfer into the write buffer is not a normal 1 transfer. Instead, whenever a 1 bit is to be written in a given tape channel, the corresponding flip-flop of the write buffer is complemented, producing a change in tape magnetization. When a 0 is to be written, the corresponding bit of the write buffer remains in its initial state, and there is no change in tape magnetization.

The structure and relative spacing of the individual tape characters is shown in the right portion of Figure 1-3. Each 36-bit computer word is divided into six 6-bit characters. However, the write buffer contains seven flip-flops WB0-6 corresponding to the seven tape channels. The seventh channel (WB6) is the parity channel. A lateral parity bit is generated for each 6-bit data character and written into the parity channel. WB0 writes the most significant data bit; WB5, the least significant. The parity of the character may be either odd (binary) or even (BCD) as specified by the program. In reading the tape, only 1s are detected.

The smallest unit of information that can be written on the tape is a record. Since each computer word contains six 6-bit characters, a record normally contains 6N data characters, where N is the number of words that the processor BLKO (or DATAO) instructions transfer out to the Data Control 136. After the last data character of the record is written, the tape control writes slightly over four character spaces of blank tape (the EOR gap), and clears the write buffer to produce an end-of-record character, the EOR mark. The bit configuration of the EOR mark produced by the write buffer clear leaves an even number of 1 bits in each of the seven channels of the tape (all bits of the write buffer start in the 0 state, and to end in the 0 state they must undergo an even number of transitions). For this reason, the EOR mark is sometimes referred to as the longitudinal parity check character.

The smallest unit of information that can be read by the tape control is also an entire record. This does not mean, however, that the entire record must be deposited in computer memory. Although the tape control always reads the entire record and always sends every data character to the data control accumulator (unless the Data Control 136 is disconnected), the processor need not read in the full record. The processor accepts 6N data characters, where N is the number of words that the processor BLKI (or DATAI) instructions request from the Data Control 136.

After the processor has accepted the last word of input data from the data control data buffer, the data control advances one more word from the data accumulator to the data buffer. All additional characters are then shifted into the data accumulator. These characters, with the exception of the final six characters of the record, are shifted out the end of the accumulator and lost.

The distance that the EOR mark moves beyond the read head, before the tape is stopped, added to the acceleration distance which the tape moves before writing is allowed, produces a section of blank tape between records. This section of blank tape is called a record gap; it is approximately 3/4 inch long. The EOR character always moves slightly further beyond the heads in writing than in reading. This is an added precaution to prevent the generation of inter-record "gap-trash" which might otherwise be produced by failing to erase the entire record gap when writing on previously recorded tape. As long as the erase heads are functioning properly, the precaution is unnecessary. (Since the direction of tape magnetization changes only when 1s are written on tape, the absence of write output data during a write command erases the tape, that is, writes blank tape with a uniform state of magnetization.)

Besides detecting changes in magnetization through the read heads, the tape transports also include a photoelectric system for sensing the beginning and end of the tape (TLP — tape load point, and TEP — tape end point).

The load point and end point of the tape are marked by reflective strips mounted on the side of the tape away from the heads. These strips are detected by photodiodes which sense light reflected from them. In writing on a newly mounted or rewound tape, a gap of about 6 inches is left from the load point before writing can begin.

TAPE WRITE AND READ SIGNAL FLOW

Figure 1-4 is a simplified block diagram of the tape system write and read paths for a single channel. The actual circuit modules used, and the distribution of these modules between the tape transport and the transport interface, varies depending on the choice of interface and transport. (Refer to the appropriate interface chapter for a description of the specific circuits used with a given transport-interface combination.)

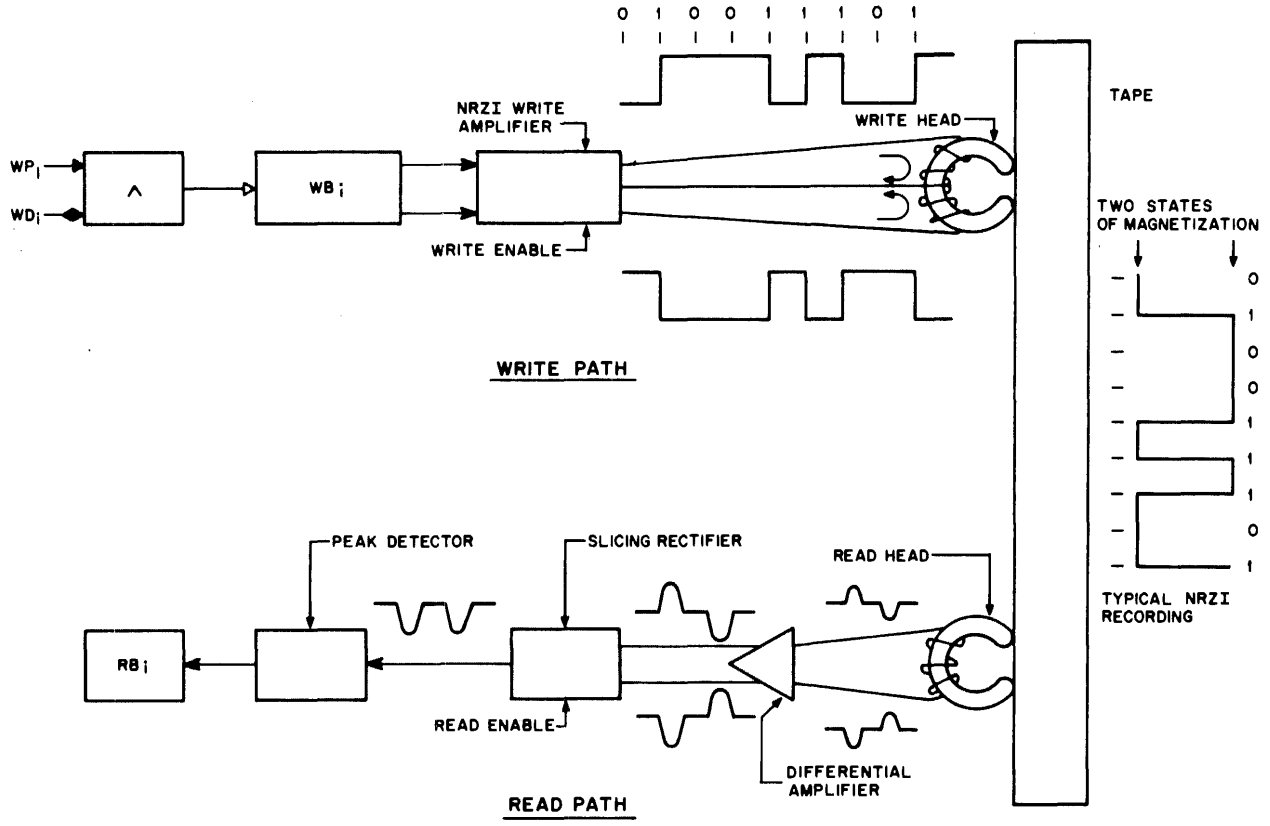


Figure 1-4 Tape Write and Read Signal Flow

Write Path

The write path is shown at the top of the figure. The WB_i flip-flop is complemented at each WP pulse that occurs while the WD_i^1 level is asserted. Both outputs of the WB_i flip-flop are connected (through various drivers not shown in the figure) to an NRZI write amplifier. When gated by a write enable signal, this write amplifier drives one of two oppositely wound coils at the write head. The choice depends on the state of the WB_i flip-flop. If the write enable level is not present, no current flows through either coil.

Whenever a 1 is to be written on tape, WD_i^1 causes the WB_i flip-flop to be complemented at the WP pulse. The transition of WB_i terminates the current through one coil and starts it in the other, changing the direction of tape magnetization and writing a 1 on the tape. As long as WB_i remains in the same state, current continues to flow through one side of the coil, and 0s are written on the tape. (The tape is then magnetized in the same direction over a series of character spaces.)

Read Path

The read path is shown at the bottom of Figure 1-4. The tape reaches the read head shortly after traversing the write head. As long as the direction of tape magnetization remains constant, no current flows through the read head coil. Each change in the direction of tape magnetization induces a current in the read head. The read current produced by two consecutive tape 1s is shown in the waveforms near the read head.

These signals are applied to a differential read amplifier that provides considerable amplification for difference signals but only fractional amplification for common mode signals. The output of the read amplifier is then sent through a slicing rectifier. This circuit can produce an output only when the read enable level is present. This prevents read signals from being sent to the tape control when tape is in motion but not being read; for example, during the rewind operation. The rectifier produces an output pulse (of one polarity) for every input pulse of either polarity from the read amplifier. However, the rectifier input circuit, to prevent a low-level noise input from generating an unwarranted output pulse, requires a minimum threshold level.

The slicing rectifier output is applied to a peak detector. The peak detector produces a logic pulse output at the peak of the input pulse which, therefore, sets the read buffer flip-flop whenever a 1 is read from the tape.

SYSTEM DATA

System specifications and physical characteristics are listed in Tables 1-1 and 1-2, respectively.

TABLE 1-1 SYSTEM OPERATING SPECIFICATIONS

System Parameters	
Number of transports per tape control	1-8
Recording method	NRZI
Characters	7 bits: 6 data, 1 parity
Parity	
Lateral	Odd or even
Longitudinal	Even (data channels only)
Density	200, 556, or 800 characters/inch

TABLE 1-1 SYSTEM OPERATING SPECIFICATIONS (continued)

Maximum transfer rate (in characters/second)	200	556	800
Interface 520, Transport	15 kc	41.6 kc	----
Interface 521, Transport 570 or Interface 522A, Transport 729			
(at 112.5 ips)	22.5 kc	62.5 kc	90 kc
(at 75 ips)	15 kc	41.6 kc	60 kc
Interface 522A, Transport 7330	7.2 kc	20 kc	----

Load point gap	6 inches
Inter-record gap	3/4 inch
EOR gap (longitudinal parity character spacing)	4-5 character spaces

Error flags	Function
PER	lateral parity error
LPE	longitudinal parity error
RCE	read comparison error
ILC	illegal command
Priority interrupt; 7 channels available	Interrupt occurs at end of each record and at illegal commands
Status bits	54 status bits available for sampling at PDP-6 console

Transport Parameters

Read-write heads	7 channel
Tape	
Material	1.5 mil polyester
Width	1/2 inch
Length	2400 feet
Reel diameter	10-1/2 inches
Data protection	Write enable lockout ring

TABLE 1-1 SYSTEM OPERATING SPECIFICATIONS (continued)

Tape Position Indications				
Load point and end point detection	Photoelectric			
Full reel and low reel	Mechanical (Used only for Transport Type 50.)			
Meaning of Position Indications				
Load point	10 feet from physical beginning of tape			
End point	14 feet from physical end of tape			
Full reel	Less than 100 feet on takeup reel			
Low reel	Less than 100 feet on supply reel			
Transport Type	50	570	729	7330
Multicontrol operation	no	yes	no	no
Slice levels available	1	2	2	2
Tape speed (inches/second)	75	75+ 112.5	75+ 112.5	36
Maximum rewind speed (inches/second)	225	450	550	220
Rewind time full reel (seconds)	400	<90	<90	1330

TABLE 1-2 PHYSICAL CHARACTERISTICS

Construction
One standard, all-steel construction DEC cabinet.

Logic Racks	
Magnetic Tape Control 516	Four racks
Interface 520	One rack
Interface 521	One rack
Interface 522A	Two racks

Modules
Standard DEC system plug-in units, series 100 and 4000.

TABLE 1-2 PHYSICAL CHARACTERISTICS (continued)

Power Equipment	
For	Use
Mangetic Tape Control 516, and any interface	Power Control Type 834-836 and two 728 Power Supplies
Magnetic Tape Control 516, Data Control 136, and Interface 520 or 521	Power Control Type 834-836 and two 728 Power Supplies
Magnetic Tape Control 516, Data Control 136, and Interface 522A	Power Control Type 834-836 and two 728 Power Supplies

Logic
Solid state. Transistors and crystal diodes utilizing static logic levels (0 vdc and −3 vdc).

Dimensions	
Height	69-1/2 inches
Width	22-1/4 inches
Depth	27-1/8 inches
Clearance, front doors	8-3/4 inches
Clearance, rear doors	14-7/8 inches

Weight (lbs)	
Magnetic Tape Control 516	120
Interface 520 or 521	20
Interface 522A	40
Cabinet	160

TABLE 1-3 POWER REQUIREMENTS

Line Voltage Input
105 to 125 volts, 60 cycle, single phase
Power plug - Hubble Twist-Loc 3-prong, 30 ampere, 250 volt

TABLE 1-3 POWER REQUIREMENTS (continued)

	Current Consumption	
	Surge Amperes	Normal Amperes
Magnetic Tape Control 516	6	4
Interface 520 or 521	1.5	1
Interface 522A	3	2

	Power Consumption
	Watts
Magnetic Tape Control 516	460
Interface 520 or 521	115
Interface 522A	230

	Heat Dissipation
	Btu/Hour
Magnetic Tape Control 516	1560
Interface 520 or 521	392
Interface 522A	785

REFERENCE CONVENTIONS

The Digital Equipment Corporation engineering drawing conventions and instruction manual referencing should be understood at this point. A study of the material contained in Chapter 13 and the following paragraphs before proceeding with detailed descriptions will save considerable reference time and preserve thought continuity when reading the text that follows.

Any reference to an illustration by a chapter-oriented figure number indicates that the figure is to be found in text following the reference. Any reference to an engineering drawing number indicates that the drawing is to be found in a special drawing section or chapter. All engineering drawings are referenced first by the full drawing number.

Example: BS-D-516-0-CM

To locate a specific signal or function on a drawing, a system of coordinates is used. As shown on the drawings of Chapter 13, coordinates are designated by a number and letter. Thus, in any drawing reference, coordinate location appears immediately after the number separated by a colon.

Example: BS-D-516-0-CM:D1

To avoid needless repetition of the full drawing number, in-text references can use a short designation form that includes only the difference modifier(s) of the drawing designation plus the coordinates.

Example: -CM:D1

One last text reference convention must be noted. Occasionally it is desirable to indicate the condition of a circuit within a logic description. As shown on the drawings of Chapter 13, circuit locations are identified. For reference in text, this designation is noted; for example, CB24. If the condition of the circuit is to be stated the reference becomes either CB24(1) or CB24(0).

The signal glossaries of Chapter 12 also are an important adjunct to both the text and drawings. These glossaries can be used in a cross indexing manner so that any signal and the conditions that generate it can be easily and completely referenced.

REFERENCE DOCUMENTS

Systems Modules Catalog, C-100	by Digital Equipment Corp.
FLIP CHIP Modules Catalog, C-105	by Digital Equipment Corp.

CHAPTER 2

TAPE CONTROL INPUTS AND OUTPUTS

PROCESSOR

Signals between the processor and the Magnetic Tape Control 516 fall into four major categories: clear signals; signals required for the CONO command; signals required for the CONI command; and priority interrupt request signals.

Clear Signals

The IOB RESET pulse (see BS-D-516-0-CM:D1) is an ungated clear pulse which is applied to all I/O devices and control units when computer power goes on, or when the operator presses the I/O reset key on the computer console. The program may also generate this reset. The IOB RESET is amplified and sent on to the transport interfaces as POWER CLEAR.

Within the tape control, the IOB RESET clears the command hold buffer, and resets the command sync quad-flop and the tape status quint-flop. It also generates two additional clear pulses MCL and CCB which complete the tape control clear operation. These two clear pulses are also applied to the transport interface (with the new designations STATUS CLEAR and 0 → CM). The tape control engineering drawings show one additional clear signal called INT CLR. This pulse is not used at present.

CONO Signals

The processor CONO (conditions-out) instruction produces an IOB CONO CLR pulse followed 1 μsec later by an IOB CONO SET pulse (BS-D-516-0-CM:A1). These two command pulses are effective only at the I/O device or control unit specified by the CONO instruction device selection code. Although the tape control may occasionally receive CONO 224 and CONO 230 instructions for certain specialized functions (described in Chapter 10), these instructions are exceptions; nearly all tape control CONO instructions use device selection code 220.

When gated by the SEL (Select Tape Control 516) level, which is only asserted when IOS3-9 represent 220 octal (BS-D-516-0-CM:C1), the IOB CONO CLR pulse generates a CLR pulse which clears the command hold buffer CHB20-35. The IOB CONO SET pulse, when gated by the SEL level, transfers the CONO control information into the command hold buffer CHB20-35.

CONI Signals

The processor CONI (conditions-in) instruction produces an IOB STATUS level. This level is effective only at the I/O device or control unit specified by the CONI instruction device selection code. The IOB STATUS level places status information from the selected device or control unit on the I/O bus for 2.5 μsec. The processor reads the information from the bus into its arithmetic register at the 2-μsec point.

Each of the three device addresses assigned to the tape control, SEL (220 octal), SLT (224 octal), and SCS (230 octal) gates 18 status bits onto the bus (see BS-D-516-0-ST for the specific gating control of the 54 status bits). For maintenance purposes, the most relevant tape control and tape transport status information can be monitored directly at the processor console.

Priority Interrupt

The processor can assign a priority-interrupt request channel to the tape control by means of a CONO instruction which sets CHB33-35 to some octal number from 1 to 7. If all three bits are 0, no priority interrupt channel is assigned. By asserting the PIE level (BS-D-516-0-CM:A8), the tape control can request a priority interrupt break on the assigned channel. The PI request is sent to the processor by grounding the appropriate PIR (priority interrupt request) line. The lower the channel number, the higher the priority. Breaks on channel 1 take precedence over all other breaks. Breaks on channel 2 take precedence over all breaks except those on channel 1, etc.

The conditions under which the PIE level is asserted are themselves subject to program control. There are four flip-flop switches which determine which of four possible priority interrupt conditions is to be allowed to assert the PIE level (see BS-D-516-0-MD:C6-7). Every CONO 224 instruction sets the four priority interrupt enable switches EFE, LIE, ICE and XNE to the states determined by bits 33, 20, 35, and 21, respectively, of the CONO 224 instruction.

When a given switch flip-flop is set to 1, the corresponding interrupt condition is gated through to assert PIE; when the switch is left in the 0 state, the input condition is not allowed to produce a PIE.

The four priority interrupt conditions that can be enabled to generate PIE are ERF(1), LIF(1), ICR, and XNC. The ERF(1) condition is asserted when the end of record flag is set; this is perhaps the most frequently used interrupt condition. The LIF(1) condition is asserted when the load point interrupt flag is set; this interrupt condition is used primarily during rewind commands. The ICR condition indicates that the interface and control are ready. The XNC condition represents the transfer new command state of the command sync quad-flop (see Chapter 3, Command Sync Quad-Flop). This condition indicates that the tape control is free to accept another CONO 220 command. The XNC interrupt condition is particularly useful during space-to-end-of-file commands when use of the ERE interrupt would be inconvenient. The four interrupt conditions that produce the PIE level are summarized in Table 2-1.

TABLE 2-1 PRIORITY INTERRUPT CONDITIONS

CONO 224 Bit No.	Sets Switch	Permitting Interrupt Condition to Assert PIE
33	EFE	End of Record Interrupt ERF(1)
20	LIE	Load Point Interrupt LIF(1)
35	ICE	Interface and Control Ready ICR
21	XNE	Transfer New Command Interrupt XNC

DATA CONTROL 136

The signals between the Data Control 136 and the Mangetic Tape Control 516 pertain to either output (write) operations, or input (read) operations. For any operation in which the data control selects the tape control, the SEL 3 level causes the DCS flip-flop (BS-D-516-0-MD:B7) to be set at BRP.

Output Signals

During RDC (read compare) and WRT (write) commands, the data control applies the contents of its six high-order accumulator bits DA0-5 to the tape control via the output levels DA0(1)-DA5(1). Negative levels represent 1s in the corresponding accumulator bit positions. Although contents of the six high-order accumulator bits are always present at the tape control input terminals (BS-D-516-0-RD:C1-4), the tape control only samples these outputs during the WRT and RDC commands.

The tape control sends TK/GV LT (take or give character shift left) command pulses to the Data Control 136 during RDC and WRT commands (and also during the input command RED). These command pulses synchronize data control operations to the timing requirements of the tape control and the transports. At the receipt of each TK/GV LT pulse, the data control shifts the contents of its accumulator six bit positions left. This shift advances the next 6-bit output character into position to be sampled by the tape control.

Each character requires a single TK/GV LT pulse; no pulses are required for EOR marks. For all three commands during which the tape control sends TK/GV LT command pulses to the data control, the pulse produces certain common effects in the data control logic. At the arrival of the TK/GV LT, the data accumulator is shifted left by six bit positions, and the data control character counter is incremented by 1.

Input Signals

During RED, SPF, BER, SPB, and RDC commands (see signal glossary), the transport interface applies the contents of its read buffer register RB to the tape control via the input data levels RBB0(1)-RBB5(1) (BS-D-516-0-RD:B2-5). Negative levels represent 1s in the corresponding read buffer bit positions. The RBB input data is read into the Data Control 136 only during the two commands RED (read) and BER (backward read). If the RED or BER command is in BCD (even) parity mode, the Hollerith code 0 character (12 octal) is converted to 00 octal before being applied to the data control (BS-D-516-0-RD:C1-3).

For the RED command only, the tape control strobes each input data character into the six low-order bits of the data control accumulator with a TK/GV LT pulse. On receiving each TK/GV LT,

the data control shifts its accumulator left six bits to make room for the input character and simultaneously strobes the character into the six low-order accumulator bits DA30-35. As in the case of the RDC and WRT output commands, the data control increments its character counter at each TK/GV LT.

For the BER command only, the tape control strobes each input character into the six high-order bits of the data control accumulator with a TK/GV RT pulse. On receiving each TK/GV RT, the data control shifts its previous accumulator contents right six bits to make room for the input character and simultaneously strobes the character into the six high-order accumulator bits DA0-5. The data control also increments its character counter at each TK/GV RT.

The BER command involves one complication not present with the RED command. When executing a RED command, the EOR gap provides a convenient means of sensing the end of the data record, and the EOR mark is not read as data. However, for the BER command, the EOR mark is read first, then the EOR gap, and finally the data record. Therefore, the EOR mark initially looks like data and it is strobed into the data control by the associated TK/GV RT pulse. The tape control must prevent the data control from treating the EOR mark as a valid data character. It does this by following the EOR mark TK/GV RT with a 0 $\longrightarrow$ CCT pulse. This pulse clears the data control character counter and causes the EOR mark to be shifted out and eliminated.

TRANSPORT INTERFACE

For ready reference, signals from the transport interface are listed in Table 2-2, and signals to the transport interface are listed in Table 2-3. The signals between the tape control and transport interface can be better understood if learned in context as encountered in the subsequent chapters of the manual.

TABLE 2-2 SIGNALS FROM TRANSPORT INTERFACE

Signal Name	Function
START 2	Steps 5-flop from TCR to UPS or MOS. (Called STM within the tape control.)
BCT	Selected transport reserved by other tape control. (Interface 521 only.)
RDY	Transport ready to receive new command.
TEP	Tape at end point.
TLP	Tape at load point.
TWL	Tape write enable lockout ring out. A data-protection condition that disables write circuits.
RBCP	Read buffer character present (RB0-5 $\neq$ 0).
RBEF	Read buffer end of file. Senses end-of-file character (RB0-5 = 17 octal).
TRPE	Tape read parity error. Lateral parity error detected for character just read.

TABLE 2-3 SIGNALS TO TRANSPORT INTERFACE

Output Signal Name	Signal Name Inside Tape Control 516	Function
		Clear Pulses
POWER CLEAR	IOB RESET	I/O bus reset (initial ungated clear pulse).
STATUS CLEAR	MCL	Motion clear; occurs at beginning of each command.
0 $\longrightarrow$ CM	CCB	Command buffer clear; occurs at termination of command levels.

TABLE 2-3 SIGNALS TO TRANSPORT INTERFACE (continued)

Output Signal Name	Signal Name Inside Tape Control 516	Function
	Command Levels	
WRITE	WWE	Enables write circuits.
FOW	-	Forward command.
BAC	-	Backward command.
REW	-	Rewind command.
WEF	-	WRF or WBT command.
NOP	-	No operation.
	Control Pulses	
IOT 7104	CTT	Start pulse.
AC1 ⟶ UN	JNU	Unit selected.
MP	MDF	Motion pulse.
WP	WCP	Write pulse.
START 1	EOR 2	Write EOR mark.
0 ⟶ RB	-	Clear read buffer.
1 ⟶ EFF	-	Set end of file flag.
0 ⟶ EFF	-	Reset end of file flag.

CHAPTER 3

COMMAND LOGIC

The command logic accepts control information from the PDP-6 processor, decodes the information, and applies the resulting command signals to the transport interface. The command logic is composed of two buffer registers, a 4-state control device (the command sync quad-flop), and a set of decoding networks which process the control information in the buffer register and generate appropriate command levels for transmission to the selected tape transport.

During the CONO tape control instruction, the 16 low-order bits of the CONO effective address are transferred out from the processor to the command hold buffer CHB (BS-D-516-0-CM). Four of these control bits remain in CHB. The remaining twelve control bits are not decoded in CHB; before being decoded these bits are advanced into the command buffer CB. The advance of these twelve bits from CHB into CB takes place in two successive stages. First four selection bits are transferred to CB to select (and, if necessary, reserve) a transport; second, when the selected transport indicates that it is ready for the command, the other eight control bits are also transferred into CB. From CB, the control bits are applied to the decoding networks. The command decoding networks convert the processor control fields into the command levels required at the transport interface.

The primary function of the command sync quad-flop is to control the timing of these control information transfers, and at their completion, to start the selected tape transport.

COMMAND SYNC QUAD-FLOP

The command sync quad-flop (BS-D-516-0-CS:C6-8) controls the advance of information from CHB to CB by initiating the JNU pulse and the JNC pulse. The JNU pulse causes the selection of a single transport; JNC applies the command levels of the current instruction to that transport via the transport interface. After the command levels have been applied to the transport, the quad-flop causes the generation of a CTT pulse (−CS:B5) and the selected transport to begin the command.

Quad-Flop Structure

The command sync quad-flop is composed of four 3-input negative diode NOR gates (3E19, 20, 25). The inputs to each of these four gates are taken from the outputs of the remaining three gates. The result of this arrangement is that the quad-flop functions as a 4-state flip-flop. It remains stable in any one of its four states; the active output is at ground, and the three quiescent outputs are negative logic levels.

Basic Quad-Flop Cycle

The basic cycle of the command sync quad-flop is shown in Figure 3-1. The basic cycle operations are also shown in steps 1 through 6 of the more detailed command and status logic flow diagram, Figure 3-2. Steps 11 through 13 of the figure show variations from the basic quad-flop cycle which arise during the execution of successive commands in continue mode (CON = 1). These variations are described under Quad-Flop Continue Cycle later in this chapter.

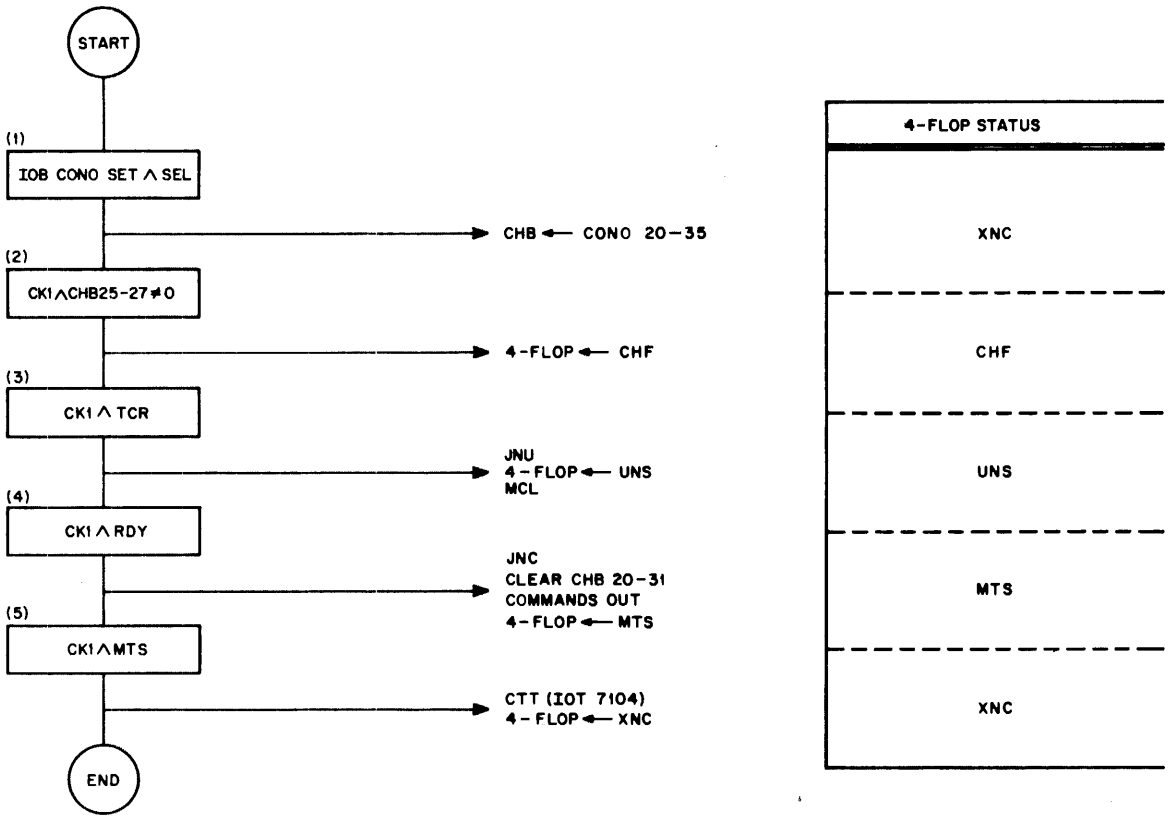
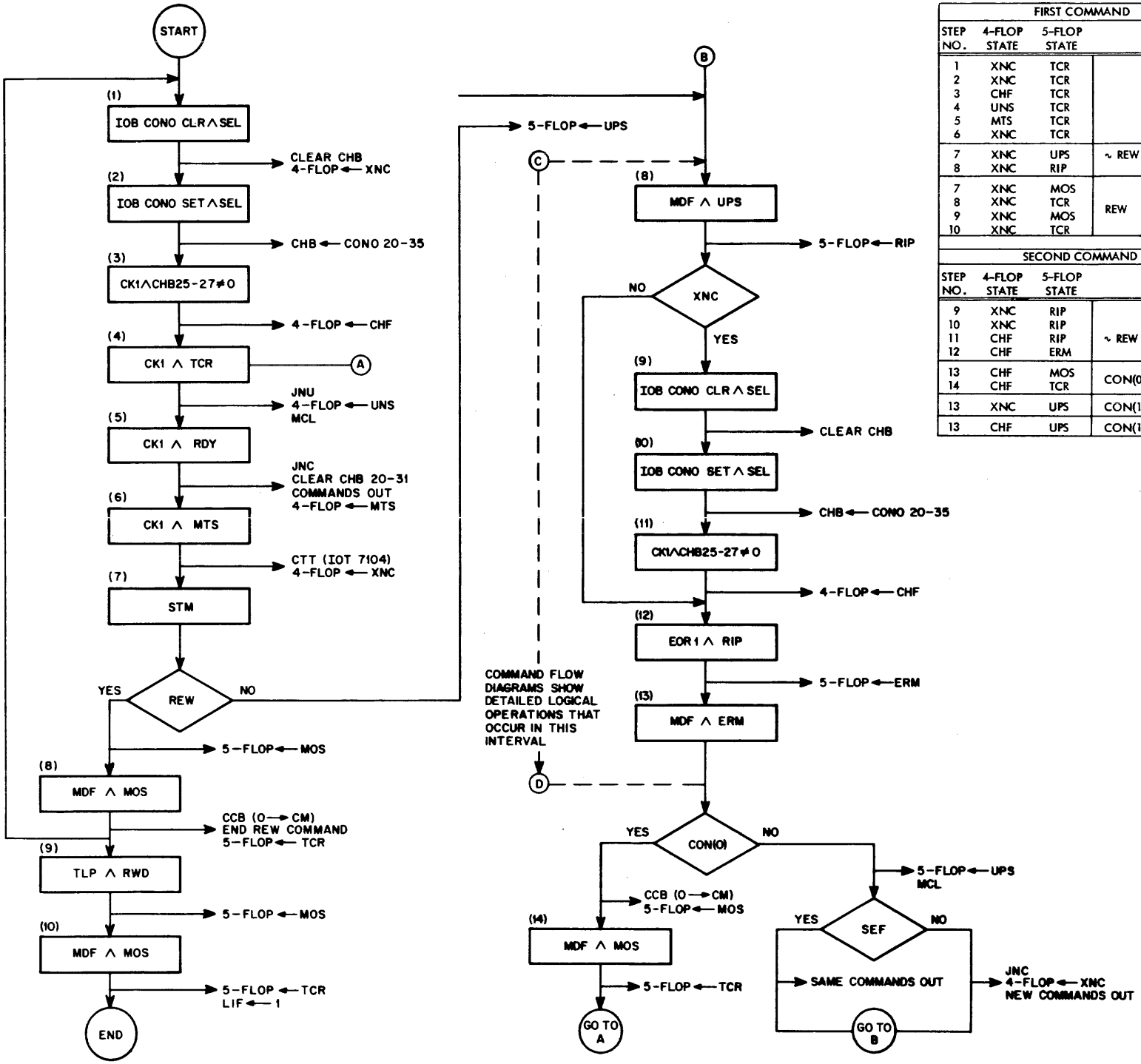


Figure 3-1 Command Sync Quad-Flop Basic Cycle



FIRST COMMAND			
STEP NO.	4-FLOP STATE	5-FLOP STATE	
1	XNC	TCR	
2	XNC	TCR	
3	CHF	TCR	
4	UNS	TCR	
5	MTS	TCR	
6	XNC	TCR	
7	XNC	UPS	~ REW
8	XNC	RIP	
7	XNC	MOS	
8	XNC	TCR	REW
9	XNC	MOS	
10	XNC	TCR	

SECOND COMMAND			
STEP NO.	4-FLOP STATE	5-FLOP STATE	
9	XNC	RIP	
10	XNC	RIP	
11	CHF	RIP	~ REW
12	CHF	ERM	
13	CHF	MOS	CON(0)
14	CHF	TCR	
13	XNC	UPS	CON(1) ∧ ~ SEF
13	CHF	UPS	CON(1) ∧ SEF

Figure 3-2 Command and Status Logic Flow Diagram

XNC State

Because the quad-flop is reset to XNC state (Transfer New Command) by IOB RESET and IOB CONO CLR, the quad-flop begins each command in XNC. The quad-flop must be in XNC to advance to its second state CHF (Command Hold Full).

CHF State

The IOB CONO SET pulse (Figure 3-1, step 1) loads the effective address of the current CONO instruction into the command hold buffer. Every tape command code has a 1 in at least one of the three bits 25-27, so CHB25-27 $\neq 0$ after the IOB CONO SET. Therefore, the first CK1 to occur after IOB CONO SET steps the quad-flop to the CHF state (Figure 3-1, step 2). The CHF state indicates that the command hold buffer is loaded with control information. The quad-flop must be in CHF to advance to its third state UNS (Unit Selected).

UNS State

As soon as the tape control is ready to initiate a command (indicated by the TCR — Tape Control Ready — state of the tape status quint-flop and CNR(0); refer to Chapter 4, Tape Status Quint-Flop), the next CK1 pulse generates a JNU pulse, and steps the quad-flop from CHF to UNS (Figure 3-1, step 3). The quad-flop must be in UNS state to advance to its fourth state MTS (Magnetic Tape Start). The JNU pulse produces an MCL pulse which clears most of the tape control flip-flops. The JNU pulse also advances the device selection information as stated earlier from CHB28-31 to CB28-31. The contents of CB29-31 specify which of the available transports is to receive the current command. The CB28 bit determines whether that transport is to be kept in a reserved status after the completion of the command or is to be returned to the pool. The JNU pulse is sent to the 521 Interface units as AC1 $\longrightarrow$ UN.

MTS State

When the selected transport is ready to execute the current command, it sends an RDY level to the tape control. This level enables the next CK1 pulse to generate a JNC pulse (Figure 3-1, step 4). The JNC pulse advances the contents of CHB20-27 into CB20-27. These eight control bits are decoded, and the resulting command levels are applied to the selected transport.

The JNC pulse clears CHB20-31 (once the control information is advanced to CB, it is no longer needed in CHB). The JNC pulse also steps the command sync quad-flop to its fourth and final state MTS.

Reset to XNC

The CK1 pulse immediately following the CK1 that produced JNC (and applied the command levels to the selected transport) generates the CTT pulse (BS-D-516-0-CS:B8). This pulse resets the command sync quad-flop from MTS to XNC. The same pulse is sent to the transport interface as IOT 7104 and returns from the interface (sometimes delayed, depending on the type of interface used) as START 2. After it reenters the Magnetic Tape Control 516, it is called STM (BS-D-516-0-CS:B3).

The IOT 7104 pulse causes the transport to start tape motion. The STM pulse sets the tape status quint-flop to the UPS acceleration state (see Chapter 4, Tape Status Quint-Flop).

Quad-Flop Continue Cycle

At any time after the command sync quad-flop is reset to XNC and the tape-status quint-flop has been stepped to UPS, a second CONO instruction can be sent to the tape control. This second instruction transfers new control information into CHB (which has been empty since the JNC pulse of the first instruction cycle). The next CK1 pulse can then advance the quad-flop to the CHF state (see Figure 3-2, step 11).

The first command continues to completion while the second command remains stored in CHB. The MDF pulse at the end of the ERM state of the first command (Figure 3-2, step 13) represents a branch point in the tape control cycle. At that MDF pulse, the tape control enters the continue mode if the CON flip-flop is in the 1 state. (Refer to Continue Logic, later in this chapter, for a more detailed description of the significance of the state of CON.)

If CON = 0, the second instruction does not call for continue mode, and the command sync quad-flop follows the same basic cycle as the first instruction. From CHF, it proceeds to UNS, MTS, and is reset to XNC. This sequence is represented by the left branch of the Figure 3-2 flow diagram (including the jump from step 14 to point A).

However, if CON = 1, indicating that the second instruction does call for continuous mode operation, the quad-flop cycle never advances to UNS or MTS. Instead, the quad-flop is reset directly from CHF to XNC (by the JNC pulse). This continue mode operating sequence is represented by the extreme right branch of the Figure 3-2 flow diagram. The basic quad-flop cycle is cut short during continue mode operations because: 1) no JNU pulse is required during continue mode — the second instruction always selects the same transport interface as the first instruction; and 2) no IOT 7104 pulse is sent to the selected transport — the transport is already moving in the required direction. Therefore, since both the JNU and IOT 7104 pulses are omitted during continue mode instructions, there is no need for the quad-flop to step through the UNS and MTS states; instead it can be reset directly from CHF to XNC (Figure 3-2, step 13, right branch). In continue mode, the MDF pulse that ends the tape-status quint-flop ERM state (Figure 3-2, step 13) also produces a JNC pulse. This JNC pulse resets the quad-flop from CHF to XNC state, clears CHB, and advances the new control information in CHB20-27 into CB20-27, thereby applying the new command levels to the transport interface.

Because certain transitions of the command sync quad-flop are conditioned upon the state of the tape-status quint-flop (the quint-flop TCR state is one of the input conditions governing the generation of JNU and JNC, see BS-D-516-0-CS:A3 and A6) the quad-flop cycle should not be considered in isolation; rather the quad-flop and the quint-flop are to be regarded as two interdependent elements of a single control system. The operation of the complete system is shown in Figure 3-2, and described in Chapter 4, Tape Status Quint-Flop.

COMMAND HOLD BUFFER

The CHB (BS-D-516-0-CM) is a 16-bit buffer register which receives CONO control information from the I/O bus. During the CONO tape control instruction, the 16 low-order bits of the CONO effective address are loaded into CHB20-35. (Physically, the transfer occurs over conductors IOB2-17. This wiring method is chosen to equalize the loading on the two I/O bus cables. The control information entering CHB20-35 is taken from bits 20-35 of the CONO address; so the use of IOB2-17 instead of IOB20-35 has no logical significance.)

The entire CHB is cleared by the CLR pulse which occurs at IOB RESET, INT CLR, and IOB CONO CLR $\wedge$ SEL. The JNC pulse clears CHB20-21; i.e., all of the twelve CHB bits that

JNC advances into CB. The remaining four CHB bits CHB32-35 are not cleared by JNC, but only by CLR. Since IOB CONO CLR precedes IOB CONO SET by 1 μsec, the CHB is always ready for new control information at the arrival of the IOB CONO SET. That pulse loads the effective address of the current CONO instruction into CHB by means of a 1 transfer. The IOB CONO SET is gated through to the CHB capacitor-diode inputs only when the device selection code of the current CONO instruction specifies the tape control (220 octal). Decoder 3E3 (BS-D-516-0-CM:C1) then produces a SEL level. The IOB CONO CLR pulse is also gated by the SEL level.

The JNU pulse advances the contents of CHB28-31 into CB28-31. These four bits specify which of the available transports is to receive the current command and whether the selected transport is to be kept in a reserved status after the completion of the instruction or is to be returned to pool.

When the selected transport is ready to execute the current instruction, a JNC pulse is generated. This pulse advances the eight high-order CHB bits from CHB20-27 to CB20-27. Once in CB, these eight bits are decoded to produce the command levels required by the current instruction. The JNC pulse also clears CHB20-31.

The four low-order bits CHB32-35 are never advanced into CB. These four bits are used while they are in CHB. Bit CHB32 is the disable EOR 1 maintenance bit. Setting this bit to a 1 with a CONO instruction inhibits the ERE level and prevents the generation of EOR 1 pulses. When the EOR 1 pulses are disabled, the tape control treats the tape essentially as a single record. This permits the entire tape to be traversed with a single command. Bits CHB33-35 assign a priority interrupt channel to the tape control; for a description of the priority interrupt logic refer to Chapter 2, Priority Interrupt. Device address 220 samples CHB20-35 on IOB20-35.

COMMAND BUFFER

The CB (BS-D-516-0-CM) is a 12-bit buffer register which receives CONO control information from the CHB at the JNU and JNC pulses. This information is decoded only from CB; it has no effect on the transport interface as long as it remains in CHB.

The CCB pulse (clear command buffer) clears CB. This pulse occurs at IOB RESET, INT CLR, and whenever the ILC flip-flop is set (indicating that the current command is illegal). For all noncontinuous commands except rewind, the CCB also occurs at the end of the ERM state of the tape status quint-flop (Figure 3-2, step 13, left branch). This occurrence of CCB terminates the application of the command levels to the transport interface. During the rewind command, the CCB terminates the command levels at the end of the quint-flop MOS state (step 8, REW). The UNU $\wedge$ SNU condition for generating CCB is relevant only at the transport interface.

The contents of CHB28-31 are first advanced into CB28-31 by JNU, and (at least one CK1 interval later) the JNC pulse advances CHB20-27 into CB20-27. The CB contains six data fields. The significance of these fields is indicated in Table 3-1.

TABLE 3-1 CB REGISTER BIT ASSIGNMENT

Bits	Function
CB20	Governs slice level. Set to 1 for old tapes where more signal is required. (Not applicable to Interface 520.)
CB21	Parity mode. Set to 1 for binary (odd) parity; to 0 for BCD (even) parity.
CB22-23	Density in bits per inch. 00 = D200 01 = D556 10 = D800 11 = D556
CB24-27	Command codes.
CB28	When 1, transport is to be kept reserved after completion of command. When 0, transport is to be returned to pool. (Interface 521 only.)
CB29-31	Device selection code. (Can select up to eight tape transports.)

COMMAND DECODING

The four command bits CB24-27 are decoded to specify twelve different tape control operations (or no operation). The decoding is done in two stages. Bits CB25-27 are applied to binary-to-octal decoder 3E8 (BS-D-516-0-CM:A4). Three of the eight resulting levels, NOP, WRT,

and RDC, specify unique tape commands regardless of the state of CB24. The remaining five levels each produce two different commands, the choice depending upon the state of CB24. All commands are summarized in Table 3-2.

For each of the commands except rewind-unload, the command abbreviation represents a signal level actually occurring within the tape control. There is no rewind-unload level generated in the tape control. The REW level is sent directly to the transport interface; but when CB24 contains 1, the CON input to the interface causes the execution of a rewind-unload instead of a normal rewind.

TABLE 3-2 COMMAND DECODING

CB25-27	Decoder Output Level	CB24	Command to be Executed
000	NOP	X	NOP (No Operation)
001	REW	0	REW (Rewind)
		1	Rewind-unload (Not used with Interface 520.)
010	WRT	X	WRT (Write)
011	WEF	0	WRF (Write file)
		1	WBT (Write blank tape)
100	RDC	X	RDC (Read compare)
101	RDE	0	RED (Read)
		1	BER (Backward read)
110	SPF	0	SPF (Space forward)
		1	SPF $\wedge$ SEF (Space forward and space to end of file)

TABLE 3-2 COMMAND DECODING (continued)

CB25-27	Decoder Output Level	CB24	Command to be Executed
111	SPB	0	SPB (Space backward)
		1	SPB ∧ SEF (Space backward and space to end of file)

OUTPUT COMMAND LEVELS

There are four basic command levels sent to the transport interface units: WRITE, BAC, FOW, and REW (write, backward, forward, and rewind BS-D-516-0-CS:A2); at least one of these levels is used with every command. (Other levels and pulses of more specialized application are also sent from the tape control to the interface units; these levels and pulses are described in the context of their generating functions and in the appropriate interface chapters.)

The WRITE level is applied to the transport interface during both WRT and WEF commands. Within the tape control, WRITE is designated WWE. The WRITE level enables the write logic of the selected tape transport.

The BAC level is applied to the transport interface during both BER and SPB commands. Within the tape control, BAC is designated RSB. The BAC level causes the selected tape transport to operate in the backward direction. The BAC level is not sent to the 521 Interface. The FOW level is applied to the transport interface during those commands calling for forward tape motion. It is generated whenever the current command is neither a NOP, a REW, nor an RSB command.

The REW level is applied to the transport interface units during the REW and rewind-unload commands. It causes the selected tape transport to enter the rewind mode (reverse tape motion at about double the normal reading speed). The CON level is applied directly from CB24(1) to the transport interface. When both CON and the REW level are asserted, CON causes the selected transport to execute a rewind-unload.

ILLEGAL COMMANDS

It is illegal for any command to be given to a transport that is already selected by another tape control (BCT). A read backward or space backward command cannot be given when the tape is at load point (TLP). Neither a write nor a write end-of-file command can be given when the tape write enable lockout ring is out (TWL).

The ILS level (BS-D-516-0-CS:A3) is asserted whenever an illegal command is given, indicating that the tape should not be accelerated. The ILS signal resets CON (and holds it reset regardless of the state of PRO at MDP). The ILS level also prevents the tape status quint-flop from being stepped to the UPS state (Chapter 4, Basic Quint-Flop-First Command), and causes the STM pulse to set the ILC flip-flop. The ILC set produces a CCB pulse that clears the CB register. The CCB pulse is sent to the transport interface as 0 $\longrightarrow$ CM. The effect of the CB clear is to terminate the command levels to the interface and thereby to prevent an illegal command from starting tape motion.

The state of ILC can be sampled on IOB18 by device address 224.

CONTINUE LOGIC

The state of CON (the continue flip-flop; BS--D-516-0-CS:D3) at the MDF pulse marking the end of ERM controls the stepping sequence of the tape status quint-flop and determines the operating cycle of the entire tape control (refer to Chapter 4, ERM state). At the MDP pulse preceding the end of ERM, if ILS is not asserted, the state of CON is determined by the PRO level (proceed; –CM:C7) and the assertion or nonassertion of PRO is, in turn, determined by whether or not the second command is sufficiently similar to the current command to require continue-mode operation.

To establish this similarity, the second command in CHB is compared with the current command in CB. If both commands select the same tape transport and if both require a similar response from that tape transport (motion in the same direction combined with writing — or not writing — for both commands), the PRO level is asserted. The CON flip-flop is then set, causing the tape control to enter continue mode.

Generation of PRO is accomplished by combining a number of subsidiary logic functions, including the CNA, CNB, CNC, RBC, and SEF functions (any one of which indicates that the requisite similarity of commands exists). The full logical conditions for generating PRO are listed and explained in the signal glossary, Chapter 12. Table 3-3 summarizes those command sequences which require continue-mode operations and the associated logic function which generates PRO during each such sequence.

TABLE 3-3 COMMAND SEQUENCES THAT REQUIRE CONTINUE MODE

Second Command	Current Command	During ERM State the PRO Level is Generated by:
WRT WEF	WRT WEF	CNA - continue A; (write after write)
RED SPF RDC	RED SPF RDC	CNB - continue B; (read forward continue)
SPB	SPB BER	CNC - continue C; (space backward continue)
BER	SPB BER	RBC - read backward continue
EFF(0)	SPF $\wedge$ CB24(1)	SEF - space to end of file
EFF(0)	SPB $\wedge$ CB24(1)	SEF - space to end of file

CHAPTER 4

TAPE STATUS LOGIC

The tape status logic consists of a 5-state control device (the tape status quint-flop) and a programmable delay network which furnishes the basic time delays required to accelerate, move, and decelerate the tape.

TAPE STATUS QUINT-FLOP

The tape status quint-flop (−CS:C1-6) continuously reflects the current status of the selected tape transport. The initial quiescent state of the quint-flop is called TCR (tape control ready). The second state UPS (up to speed) represents the acceleration period during which the tape is brought up to the required speed for forward or backward commands (but not for rewind; see REW Command Cycle, later in this chapter). During RIP (record in progress), the third state of the quint-flop, the tape is moving at the correct speed for reading or writing. The fourth ERM state (end of record motion) represents the tape runout interval between sensing the EOR mark and beginning to stop the tape. The final MOS (motion stop) state is the deceleration state during which tape motion is stopped.

Quint-Flop Structure

The tape status quint-flop is composed of five 4-input negative diode NOR gates (3E18, 19, 20). The inputs to each of these five gates are taken from the outputs of the remaining four gates. The result of this arrangement is that the quint-flop functions as a 4-state flip-flop. It remains stable in any one of its five states; the active output is at ground, and the four quiescent outputs are negative logic levels.

Basic Quint-Flop Cycle — First Command

The basic cycle of the quint-flop is shown in simplified form in Figure 4-1. The same basic cycle operations are shown in greater detail in Figure 3-2. The latter figure not only shows

the quint-flop operations but also shows the concurrent operations of the command sync quad-flop. In addition to the basic cycle operations, Figure 3-2 also includes certain variations of the basic quint-flop cycle which arise during the rewind command (see REW Command Cycle), and during the execution of successive commands in continue mode (Second Command - CON(1)Λ ∿ SEF and Space to End of File Command - CON(1)ΛSEF in this chapter).

TCR State

Because the quint-flop is reset to TCR state (Tape Control Ready) by IOB RESET, the quint-flop is always in TCR when the tape control receives its first command. The JNU pulse can be generated and the command sync quad-flop can be stepped from CHF to UNS only when TCR is asserted and the CNR flip-flop contains 0. Except in continue mode, JNC also can be generated only when the quint-flop is in TCR state.

UPS State

After the transport interface receives the signal to start tape motion, an STM pulse is returned to the tape control (Chapter 3, Reset to XNC). This STM pulse signals the beginning of tape acceleration. Provided that the command is not illegal and not a REW command, the STM advances the tape status quint-flop to the UPS acceleration state (up to speed). The transition from TCR to UPS is shown in Figure 4-1, step 1, and in Figure 3-2, step 7, ∿REW. The quint-flop must be in UPS to advance to its third state RIP (record in progress).

RIP State

After a predetermined acceleration delay (which varies depending on the nature of the current command), the motion delay logic (see Motion Delay Logic, later in this chapter) produces the MDF pulse that steps the quint-flop from UPS state to RIP state. In the RIP state, the tape is moving at the correct speed for reading or writing. All reading and writing must occur in RIP. (The RIP level is a necessary input condition for generating the RCP pulse, the ERC level, the EMC level, and the ERD delay; these functions are described in Chapters 5 and 6). The transition from UPS to RIP is shown in Figure 4-1, step 2, and in Figure 3-2, step 8, ∿REW. The tape status quint-flop must be in RIP to advance to its fourth state ERM (end record motion).

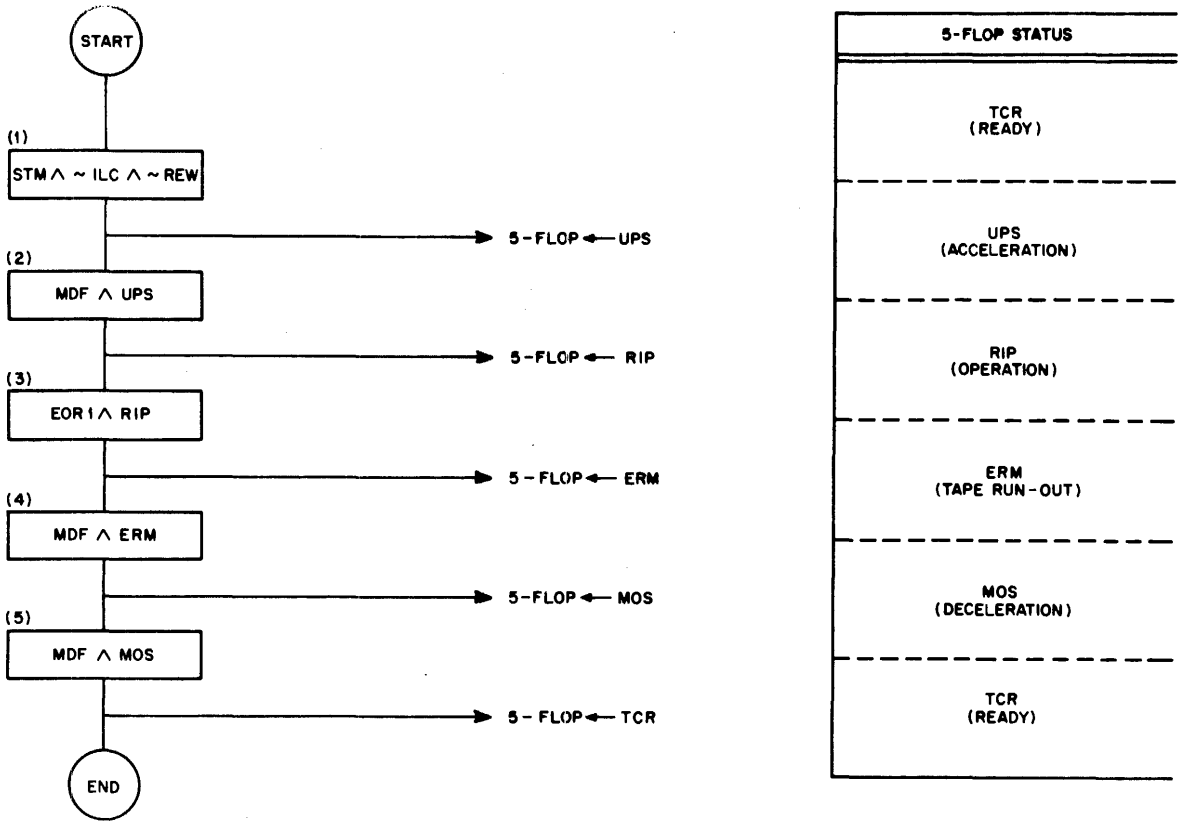


Figure 4-1 Tape Write and Read Signal Flow

ERM State

When a command reaches the end of record, the EOR gap causes the generation of an EOR 1 pulse. That pulse steps the tape status quint-flop from RIP state to ERM state. The ERM state represents the tape runout interval between sensing the EOR mark and beginning to stop the tape. The ERM level is a necessary input condition for generating the TZA pulse which sets the LPE and RCE flip-flops (see Chapter 6, Read Comparison and Longitudinal Parity). The transition from RIP to ERM is shown in Figure 4-1, step 3, and in Figure 3-2, step 12.

For noncontinuous commands (CON = 0), the quint-flop advances from ERM state to its fifth and final state MOS (motion stop). During continuous-mode commands (CON = 1), the quint-flop is reset directly from ERM to UPS; continue mode operations are described in greater detail under Second Command - CON(1)∧ ~ SEF and Space to End of File Command - CON(1)∧SEF in this chapter.

MOS State

After a predetermined runout delay (which varies depending on the nature of the current command), the motion delay logic (-MD) produces the MDF pulse that steps the tape status quint-flop from ERM state to its final MOS state. The delay selected determines the correct amount of tape which is allowed to run past the tape heads before starting to brake the transport and decelerate the tape. The same MDF pulse that signals the expiration of the ERM runout delay also produces a CCB pulse which clears CB and terminates the application of command levels to the transport interface. The MOS deceleration state represents the time interval required to actually bring the moving tape to a complete stop. The transition from ERM to MOS is shown in Figure 4-1, step 4, and in Figure 3-2, step 13, CON(0). During REW commands, the MOS state is used for an entirely different purpose; the duration of MOS then controls the period during which the REW command level is applied to the transport interface; this alternate function of MOS is described under REW Command Cycle below.

Reset to TCR

After a predetermined deceleration delay (which varies depending on the nature of the current command), the motion delay logic produces the MDF pulse that resets the tape status quint-flop from MOS state to TCR. This quint-flop reset is shown in Figure 4-1, step 5, and in Figure 3-2, step 14, CON(0). Once the tape status quint-flop is again in the TCR state, the JNU pulse can be generated, advancing the device selection portion of the next (second) command from CHB28-31 to CB28-31, and stepping the command sync quad-flop from CHF to UNS.

The generation of JNU and the CHF to UNS transition of the quad-flop assumes that the quad-flop was in CHF by the time that the tape status quint-flop is reset to TCR. If no new command has been given by the time that the quint-flop reset occurs, CHB remains empty, and the command sync quad-flop remains in XNC until the next CONO tape control command arrives from the processor.

REW Command Cycle

During the execution of the REW commands, rewind and rewind-unload, the tape status quint-flop follows the sequence of operations shown in Figure 3-2, steps 7-10, REW.

Command Level Timing

The REW command levels are first applied to the transport interface at the JNC pulse (Figure 3-2, step 5). One CK1 interval later, the CTT pulse is applied to the transport interface, causing an STM pulse to be returned to the tape control, and thereby advancing the quint-flop directly from the TCR state to MOS (Figure 3-2, step 7, REW). The duration of the MOS state determines the period during which the REW command level is applied to the transport interface.

The TCR to MOS transition of the tape status quint-flop is implemented in the following manner. The REW output of the command decoder (Chapter 3, Command Decoding) is inverted (–CM:C4) and applied to input K of the negative diode 3D25 (–CS:D1) preventing the TCR to UPS transition of the tape status quint-flop that normally occurs at the STM pulse during non-REW commands. The same inverted REW level is also applied to input V of capacitor-diode 3E17 (–CS:D5), enabling the STM pulse to step the quint-flop directly from TCR to MOS (as indicated in Figure 3-2, step 7, REW).

After a predetermined delay (which varies depending upon the logical requirements of the type of interface and transport used), the motion delay logic produces the MDF pulse that returns the quint-flop from MOS state to TCR state (step 8, REW). This same MDF pulse produces the CCB pulse that terminates the REW command levels. The CCB pulse is sent to the transport interface as 0 $\longrightarrow$ CM.

Deceleration Interval

If the rewinding transport is still selected when it reaches TLP, the tape status quint-flop is once more advanced from TCR to MOS (Figure 3-2, step 9, REW). After a predetermined deceleration delay (unrelated to the command-timing delay described above), the motion delay logic produces another MDF pulse which once more resets the quint-flop from MOS to TCR (step 10, REW). This second reset from MOS to TCR indicates that enough deceleration delay has elapsed to stop tape motion. The same MDF pulse that resets the quint-flop from MOS to TCR is gated by TLP (tape at load point) to set the load point interrupt flag LIF (step 10, REW). If LIE contains 1, the LIF set produces a PIE signal (priority interrupt enable) which notifies the programmer that the selected transport has reached load point. The LIF can also be sampled directly by a CONI 224 instruction.

The MOS deceleration state occurs only if the rewinding transport remains selected until it reaches TLP. (Otherwise, there is no TLP pulse to step the quint-flop to MOS.) This use of the MOS deceleration state during REW command serves as a safeguard to avoid giving a new forward command while the transport is in the process of decelerating. The programmer can take advantage of this safeguard, and can utilize the LIF flag, only by keeping the rewinding transport selected (thus keeping the quint-flop in MOS state) until deceleration is complete.

Although the tape transports have a built-in stop at TLP condition, there is still the hazard of the programmer inadvertently calling for a forward tape command on a given nonselected transport after that transport has reached TLP, but while it still retains some backward momentum. This could result in the quint-flop progressing to RIP state before the tape can come up to full forward speed.

If the programmer is certain that he will not use the rewinding transport again until after it has completed the rewind and come to a full stop, the potential difficulty no longer exists, and there is no need to keep the tape transport selected. A new command to another tape transport can be given at any time after the first time the quint-flop is reset to TCR. This programming procedure is represented by the return path from step 8, REW back to step 1.

Second Command - CON(0)

A second CONO tape control command can be programmed at any time after the first command advances the tape status quint-flop to the RIP state. In Figure 3-2, the second command is shown as steps 9 and 10, ∿REW, but note that the second command need not be given at that specific point of the first command cycle; the second command can be programmed at any point after step 8, ∿ REW.

Provided that CON = 0, the second command is executed in exactly the same manner as the first command, except that the transfer of control information from CHB to CB is delayed until CB is cleared and the first command is finished. If the second command is given before the completion of the first command (as in the sequence shown in Figure 3-2), the command sync quad-flop advances from XNC to CHF (step 11), but must wait in CHF state until the first command is completed. As described under Basic Quint-Flop Cycle - First Command above,

the tape status quint-flop completes the first command by advancing from RIP to ERM, then from ERM to MOS, and finally from MOS to TCR; see steps 12-14, CON(0).

Once the tape status quint-flop has been reset to the TCR state, the next CK1 pulse produces a JNU pulse, advancing the contents of CHB28-31 to CB28-31 and selecting a tape transport for the second command. The JNU, in turn, produces an MCL which clears the tape control in preparation for the second command. The same JNU pulse advances the command sync quad-flop from CHF to UNS. The second command then proceeds without further interruption in exactly the same manner as the first command. The completion of the first command is shown in Figure 3-2 as step 14, CON(0), and the generation of JNU is indicated as immediately following that step (connector A at step 4 of the diagram).

Second Command - CON(1)$\Lambda \sim$ SEF

Continue mode operation requires a sufficient degree of similarity between the current command and the next (second) command so that there is no need to stop the tape at the end of the current command, (refer to Chapter 3, Continue Logic). Instead, tape motion is allowed to continue. After the runout of sufficient tape for the correct record gap, the second command is executed immediately without any pause to stop and restart the tape. This change from normal non-continuous operation requires modifications in the operating cycles of both the command sync quad-flop and the tape status quint-flop.

Decision to Enter Continue Mode

At the MDF pulse that concludes the ERM phase of the first command, the state of the CON flip-flop is sampled to determine whether or not the second command is to be executed in continue mode. The state of CON at MDF is, in turn, dependent upon the state of the PRO level at the preceding MDP pulse, which occurred 1 μsec before the MDF pulse. (For a detailed description of the logical conditions which govern the state of the PRO level and determine whether or not the tape control is to enter continue mode at the end of ERM, refer to Chapter 3, Continue Logic.)

Summary of Quint-Flop and Quad-Flop Operation - CON(0)

If CON(0) is asserted at the MDF that terminates ERM, no continue mode operation is called for, and the tape control follows the CON(0) sequence described previously under Second Command - CONO(0). This sequence is represented by the left branch of the flow diagram; Figure 3-2, steps 13-14, CON(0). The first command is completed in the normal manner. The tape status quint-flop steps through its five states in regular rotation: TCR, UPS, RIP, ERM, MOS, and then is reset to TCR in preparation for the second command. The command sync quad-flop also follows the CON(0) sequence described under Second Command - CONO(0). During the execution of the second command, the quad-flop proceeds through the four states of its cycle in regular rotation: XNC, CHF, then (after waiting for the completion of the first command and the reset of the quint-flop to TCR), UNS, and MTS, and finally a reset to XNC in preparation for still further commands.

Tape Status Quint-Flop Operation - CON(1)

When CON(1) is asserted at the MDF that terminates ERM (indicating that the second command is to be executed in continue mode), the quint-flop and quad-flop operating cycles differ considerably from those described above. When CON = 1, the tape status quint-flop skips the final MOS state. It proceeds through the first four states as usual, but from ERM it is returned directly to the UPS state instead of being advanced to MOS. The MDF pulse at the end of ERM causes this return to UPS (see Figure 3-2, step 13, CON(1)). Unless the current command is a SEF command, this same MDF pulse also produces a JNC pulse, advancing the control information of the second command from CHB20-27 to CB20-27 and applying the command levels of second command to the transport interface. (If the current command is a SEF command, the JNC pulse is omitted (refer to Space to End of File Command - CON(1)ΛSEF below.)

Command Sync Quad-Flop Operation - CON(1)

The command sync quad-flop cycle is also altered during continue mode operations. Although the quad-flop is stepped from XNC to CHF after the second command is loaded into CHB, regardless of whether or not the second command requires continue mode operation (Figure 3-2, step 11), the quad-flop is never advanced to UNS or MTS when CON contains 1. Instead the same MDF pulse that terminates the ERM state of the tape status quint-flop generates a JNC

pulse; this JNC resets the command sync quad-flop directly from CHF to XNC (see Step 13, CON(1)). Because the continue mode requires that the same tape transport be selected for successive commands, no JNU pulses are needed. Since the transport is not stopped between commands, the start pulse is also unnecessary; no IOT 7104 pulse is sent to the transport interface, and no STM pulse is returned to the tape control. Because both the JNU pulses and the start pulses are omitted during a series of continue mode commands, the quad-flop cycle can be shortened by deleting both the UNS and MTS states.

Flow Diagram - CON(1)$\wedge \sim$ SEF

For those continue mode operations in which the current command is not a SEF command, the NDF pulse that terminates ERM produces the sequence of operations represented by the right subbranch of the right branch of the Figure 3-2 flow diagram (step 13, CON(1)$\wedge \sim$ SEF). The MDF pulse steps the tape status quint-flop from ERM state to UPS. The same MDF pulse produces an MCL pulse that clears all tape control flip-flops except the four interrupt enable switches, CON, and LPH in preparation for the next command. The MDF also generates a JNC pulse that steps the command sync quad-flop from CHF state to XNC, advances the control information of the second command into CB, and clears CHB20-21.

After the normal acceleration delay for the type of command represented by the second command, the motion delay logic produces another MDF pulse which steps the tape status quint-flop from UPS state to RIP state. The UPS interval is not used in this case for acceleration, but rather as a tape runout interval to allow for the correct record gap prior to the commencement of the second command RIP state. As soon as the quint-flop is advanced to RIP, the second command is executed in the normal manner.

From connector B at the bottom of the Figure 3-2 flow diagram, the tape control operating sequence returns to the top of the diagram (see step 8, $\sim$REW). At any time after the MDF pulse that causes the UPS to RIP transition of the quint-flop, a third command can be given. Because the command sync quad-flop is in the XNC state (at step 8, $\sim$REW), there is no need to bypass steps 9-11. These steps are omitted during the SEF command sequence described in the next paragraph.

Space to End of File Command - CON(1)∧SEF

If the current command is an SPF (space forward) or SPB (space backward) with CB24(1), the selected transport executes an SEF (space to end of file). This is done by repeating the SPF or SPB command until the file mark record is sensed. The SPF or SPB commands making up the SEF sequence follow the left subbranch of the right branch of the flow diagram (see Figure 3-2, CON(1)∧SEF).

Command Iteration

The entire SEF sequence is controlled by the single SPF or SBP command in the CB register. That command is executed over and over again until the tape reaches the end-of-file record. During each iteration of the command that occurs while the tape control remains in the SEF loop, the sequence of operations returns from the bottom of Figure 3-2 to the top of the diagram (connector B). Each time the tape status quint-flop advances from ERM state to UPS state (step 13, CON(1)∧SEF), another iteration of the SPF or SBP command begins.

No JNC pulse is produced between successive space commands (see —CS:B3). Therefore, CB is not cleared; and if a second command is programmed during the execution of the SEF, the control information for that command remains in CHB until the SEF is completed. Until CHB is free, no further commands can be sent to the tape control without destroying the second command.

Quad-Flop States During SEF Loop

The flow diagram shows the programming of a second command during the execution of the SEF loop (Figure 3-2, steps 9-11). This second command is stored in CHB while the SEF is in progress; the presence of the command code in CHB steps the command sync quad-flop from XNC state to CHF (step 11). Since no JNC pulses are produced during the iterated SEF commands, the quad-flop remains in CHF state throughout the SEF loop, and may not be reset to XNC until the SEF is completed. The CHF state of the quad-flop warns the programmer that the CHB is full and that the tape control is not ready to receive a third command. In the flow diagram, this restriction is indicated by the path bypassing steps 9-11 when XNC is not true.

Escape from SEF Loop

When the end-of-file record is sensed, the EOF level sets the EFF flip-flop and ends the SEF level. The termination of the SEF level permits the tape control to escape from the SEF loop. The MDF pulse that marks the end of the ERM state of the final SPF or SPB commands (the commands that read the file mark record) then causes the tape control to follow either the left branch of the flow diagram, CON(0), or alternatively, the right subbranch of the right branch of the diagram (CON(1)∧SEF). The choice depends on whether or not a continue-mode command follows the SEF.

If a continue-mode command follows the SEF (e.g., a BER command following a backward SEF), the MDF that terminates the ERM phase of the final SPB iteration produces a JNC pulse. That JNC pulse advances the new command from CHB to CB, and resets the command sync quad-flop from CHF to XNC. The control then starts to execute the new command, and because XNC is true, still another command can be sent into CHB (Figure 3-2, steps 9-11).

If the command that follows the SEF sequence is not a continue mode command (e.g., a BER command following a forward SEF), the MDF that terminates the ERM phase of the final SPF iteration cannot produce an immediate JNC pulse. Instead, the tape control follows the sequence indicated by the left branch of the flow diagram, CON(0). The tape status quint-flop then progresses from ERM state to MOS (step 13), and from MOS to TCR (step 14). A JNU can then be generated (step 4), advancing the command sync quad-flop to the UNS state. From UNS, the quad-flop advances to MTS and then to XNC. (As soon as the quad-flop is reset to XNC state, CHB is free to accept another command.)

MOTION DELAY LOGIC

The motion delay logic (BS-D-516-0-MD) is composed of a 4303 Integrating Single Shot module MOD, a set of logically gated delay resistors, a triggering network, and a pair of output pulse amplifiers. Any one of 12 motion delay resistors can be gated into the MOD single shot circuit. Each resistor produces a different delay period. A major advantage of this design is the efficient use of the single delay circuit which provides twelve separate programmed delays. Similar delay networks are used in the write and read logic described in Chapters 5 and 6.

Triggering

The MOD single shot is triggered by a positive-going level change at input terminal S while a ground level is applied to input terminal T. The rising trailing edge of the SMD output pulse from pulse amplifier 3D19 (–MD:B5) provides the level change. The ground level gate is derived from the terminal K output of negative diode 3D18. The delay period is started by setting MOD to the 1 state. At the expiration of the delay, the MOD resets itself to the 0 state.

Output Pulses

When the MOD resets itself to the 0 state, a 1-μsec MDP pulse is generated at output terminal J of pulse amplifier 3D19 (–MD:A5). One μsec later, a 1-μsec MDF pulse is generated at output terminal V of pulse amplifier 3D16. In addition to being used within the tape control, the MDF pulse is also sent to the transport interface as MP.

Delays

Table 4-1 summarizes the operation of the motion delay logic. Although the logical gating of the various delays is the same for each of the five types of transport used with the 516 Control, the actual time constant of the delays used varies in accordance with transport requirements.

The six UPS delays (D1, D2, D11, D6, D4, and D5) are listed at the top of Table 4-1. One of these six UPS delays is used to time each UPS acceleration period. The choice depends upon the nature of the current command and whether or not the tape is at load point.

The next three entries in Table 4-1 are the three ERM delays (D10, D3, and D7). One of these three ERM delays is used to time each ERM tape runout interval. The choice between these three delays depends only upon the nature of the current command.

The next two entries are the MOS delays (D8 and D9). Delay D8 (selected by the REW level) is used to time the duration of the REW command level. Delay D9 (selected by the NOP level) is used to time all tape deceleration periods. The NOP level is asserted from the time that the CCB pulse clears the CB register. During all noncontinuous commands, NOP is asserted when MDF terminates the ERM state of the tape status quint-flop.

The twelfth and final delay D0 is never triggered, but during TCR and RIP the twelfth resistor is gated in to keep the MOD Integrating Single Shot 4303 operative.

TABLE 4-1 MOTION DELAY LOGIC

	Selected by	SMD Generated by	SMD Gated by	Purpose of Delay	Time Constant (msec) for Tape Transport Type				
					DEC 50	DEC 570	IBM 729V	IBM 729VI	IBM 7330
D1	UPS ∧ FRD ∧ LPH(0)	STM	UPS	FRD acceleration (∼ LP)	1.8	5.0	3.0	2.0	9.0
D2	UPS ∧ FRD ∧ LPH(1)	STM	UPS	FRD acceleration (LP)	24.0	33.0	24.0	16.0	50.0
D11	UPS ∧ RSB ∧ LPH(0)	STM	UPS	RSB acceleration (∼ LP)	3.0	5.0	3.0	2.0	9.0
D6	UPS ∧ WRT ∧ LPH(0)	STM	UPS	WRT acceleration (∼ LP)	6.0	5.5	7.5	5.0	7.0
D4	UPS ∧ WRT ∧ LPH(1)	STM	UPS	WRT acceleration (LP)	78.0	66.0	48.0	32.0	140.0
D5	UPS ∧ WEF	STM	UPS	WEF acceleration	48.0	26.7	40.0	26.7	83.0
D10	ERM ∧ FRD	EOR 1	ERM	FRD tape run-out	2.0	1.3	1.0	0.67	9.0
D3	ERM ∧ RSB	EOR 1	ERM	RSB tape run-out	5.5	1.5	4.0	2.7	11.8
D7	ERM ∧ WWE	EOR 1	ERM	WWE tape run-out	2.5	1.8	1.5	1.3	11.0
D8	MOS ∧ REW	MDF	MOS	REW command level duration	20.0	11.0	20.0	11.0	30.0
D9	MOS ∧ NOP	MDF	MOS	MOS deceleration	12.0	11.0	17.0	11.0	17.0
D0	TCR ∨ RIP	NO	NO	Keeps MOD 4303 circuit operative	5.0	5.0	5.0	5.0	5.0

CHAPTER 5

WRITE LOGIC

FUNCTION OF WRITE LOGIC

The write logic (shown in drawing BS-D-516-0-WT) controls the execution of three write commands: WRT, WRF, and WBT (write, write file, and write blank tape). A crystal clock within the write logic governs the rate at which WP pulses are applied to the transport interface and determines the rate at which characters are written on tape. The same internal clock also controls the rate at which the tape control sends TK/GV LT data request command pulses to the Data Control 136.

During the WRT command (see Write later in this chapter), the write logic causes the tape control to request a 6-bit output data character from the Data Control 136, then transfers the character to the transport interface to be written on tape; requests a second output character, writes it on tape, etc. This process continues as long as the Data Control 136 still has data to send (i.e., until the data control reaches the end of the current output record). When the end of the record is reached, the write logic times out an EOR gap of slightly over four character spaces and causes the transport to write an EOR mark.

During the WRF command (see Write File later in this chapter), the write logic causes an end-of-file record to be written. The EOF record consists of a file mark character (17 octal) followed by a 4-character-space EOR gap, followed in turn by an EOR mark. During the WBT command (see Write Blank Tape later in this chapter), the write logic produces a WBT level, but no WP pulses. This causes the tape to advance and erase approximately 3 inches.

CLOCK CIRCUITRY

The write logic contains two 4407 Crystal Clock circuits which run continuously as long as the tape control is turned on. Clock 1 at location 3C14 (–WT:B4) is adjusted to produce CK1 output pulses at twice the character rate of the tape transport operated at a density of 800 characters per inch. Clock 2 at location 3C15 (–WT:B6) produces CK2 output pulses at twice

the character rate of the transport operated at 556 density. Both clock frequencies are directly proportional to both the density and tape speed of the transport to be used with the tape control.

The principal output of the clock network is the CKP (clock pulse) output from pulse amplifier 3C16. The CKP provides the frequency standard for the character frequency flip-flop CF (–WT:A2). During the period when the EWC flip-flop contains 1, each CKP complements the CF (to see Start Sequence below). For each of the three available character densities (refer to Table 3-1), the CKP pulses are generated at twice the character rate of the tape transport. If D800 is asserted, indicating that the current write command calls for a character density of 800 characters per inch, a CKP pulse is produced at each CK1 pulse (–WT:B6). If D556 is asserted, CKP coincides with CK2. For D200 write commands, CKP is generated at every fourth CK1 (–WT:A6). The 2-stage counter composed of the C/2 and C/4 flip-flops (–WT:A5) provides the required frequency division. For each of the three available densities, the CKP pulses are always produced at exactly twice the frequency at which output characters are currently to be written on the tape.

Besides being used to generate CKP during D800 and D200 write commands, the CK1 pulse is also used directly for several other functions within the tape control. The CK1 pulse is used during noncontinuous commands to trigger JNU, JNC, and CTT. The CK1 is also used to advance the command sync quad-flop from XNC state to CHF whenever a new command is loaded into CHB.

WRT COMMAND CHARACTER TIMING

The transitions of the character frequency flip-flop CF govern the character rate during write commands. Every 0 to 1 set transition of CF causes the generation of a WCP write clock pulse (–WT:A4). Every 1 to 0 reset transition of CF causes a TK/GV LT (–CS:B4) data request command pulse to be sent to the Data Control 136.

WP Write Pulse

The WCP is sent to the transport interface as WP (–WT:A4) (at present it is not used within the tape control). At each WP, those flip-flops of the write buffer register which are receiving 1 data levels are complemented, causing the selected tape transport to write a 1 in the corresponding tape channel.

TK/GV LT Data Request Command Pulse

At the Data Control 136, the TK/GV LT pulse shifts the data accumulator left by six bit-positions, and increments the character counter. During both WRT and RDC commands, the data control applies the contents of its six high-order accumulator DA0-5 to the tape control via the write data output levels WD0(1)-WD5(1) (–WT:C2-4)(see Write-Data Outputs below). The 6-bit left shift caused by each TK/GV LT data request command pulse advances the next output character into position to be sampled by the tape control. As long as output data remains to be written, the sequence continues: a TK/GV LT data request command pulse is sent to the data control at each 1 to 0 transition of CF, and the ensuing output data is written on tape at the 0 to 1 transition of CF.

Synchronizing Circuit

The CKP pulses provide the complementing signals that cause the transitions of the character frequency flip-flop CF. However, although CKP pulses are continuously generated whenever any one of the three density levels is asserted, they are allowed to complement CF only while the EWC flip-flop contains 1 (–WT:B2). The EWC and WES flip-flops make up a 2-stage synchronizing network; during WRT commands, this network ensures that the write clock pulses sent to the transport interface are complete and are in phase with the CKP.

Start Sequence

The WRT command can start to write a record only after the tape status quint-flop has reached the RIP state. The MDF pulse, marking the UPS to RIP transition of the quint-flop, produces a BRP pulse (–CS:B6). During the WRT command, the BRP pulse sets the WES flip-flop to the 1 state (–WT:B3). The first CKP that arrives in time sets EWC to 1. (The transition of EWC is not necessarily in exact synchronization with the leading edge of the CKP.) The next CKP, and every succeeding CKP that occurs while EWC(1) is still asserted, complements CF. The transitions of CF are all synchronized with the leading edges of the CKP pulses.

Stop Sequence

After the last character of a record has been written, the CF flip-flop is in the 1 state. It is reset to the 0 state once more, generating a final TK/GV LT character request command pulse.

The EWD delay (–WT:C7) is set 2.4 μsec after this final TK/GV LT (see End of Record Write Delay below). The set of the EWD delay causes both the EWC and the WES flip-flops to be reset to 0. Because the EWC(1) level is no longer asserted, the following CKP is not permitted to complement the CF, so CF remains reset. Although the final TK/GV LT pulse does cause the data control to shift its accumulator contents left six bit-positions, this final shift does not bring any new data into position to be sampled by the tape control. The final TK/GV LT pulse shifts the last data character out the left end of the data control data accumulator, resets the data control character counter, and sets the DA RQ status bit to 1.

The first output character is always sampled by the tape control and written on tape at the WP that precedes the first TK/GV LT shift. Similarly, the last character of the output record is sampled and written on tape by the WP that precedes the final TK/GV LT shift. The first CF transition is always a set transition which writes the first character of the record. (This character need not be shifted; it is initially in the correct position to be sampled and written out.) The last CF transition is always a reset transition. This reset applies the final TK/GV LT to the data control after the last character has been written.

WRITE-DATA OUTPUTS

During the WRT and WRF commands, the WD output levels WD0(1)-WD5(1) (–WT:C2-4) transfer write-data to the write buffer register in the transport interface. Ground levels represent 1s in the corresponding bit-positions of the output data character. For the RDC command, the WD output levels apply the same write-data to the LPA; refer to Chapter 6, Read Compare Command for a description of the RDC command.

WRT and RDC

During the write and read compare commands, the WD outputs are the direct complements of the DA levels from the Data Control 136 for all output characters except 0 BCD. A single-pole, double-throw switch is mounted in front of logic panels 3D and 3E. The switch is necessary if there is a need to write (N) characters per record BCD so that the total number of characters written is not a factor of 6. This is accomplished by filling in with blank characters, left, (BCD 00) in the first word of a record written.

In BCD (even) parity mode, 00 octal is converted to the 0 character Hollerith code (12 octal) before being written or read-compared, provided the front panel switch is set to 12. If the front panel switch is set to 12, the unmodified 0 character is prohibited in BCD mode because with even parity the parity bit would be 0 also, leaving an entire tape frame with no 1 bits whatever. Such a frame could not be distinguished from blank tape or the total failure to write a character.

To write less than six characters in the first word of a record, the front panel switch must be set to the 00 or up position. This inhibits the conversion of BCD 00 to BCD 12. Caution must be exercised when the switch is set to this position. Conversion of BCD 00 to 12 within a record must be done by the program prior to the execution of a BLKO or data instruction to the 136 Data Control.

When the record is read, the blank character will be ignored. Thus, each word read will be shifted to the left n characters as determined by the number of blank characters written.

For all characters except the BCD 0, the negative output of inverter 3C18 (–WT:C1) causes the DA levels to be inverted and gated through to the corresponding WD outputs. Although the DA levels are negatively asserted, the corresponding WD outputs are asserted at ground. When the 00 octal character occurs in BCD parity (CB21 = 0) and the switch is set to 12, the level DA = 0 appears at terminal J of diode 3C21 (–WT:D3). This level causes two inverters at location 3C5 to ground the WD2(1) and WD4(1) outputs. The remaining four WD outputs are held negative because the corresponding DA levels represent 0 bits. In this way, the six WD outputs form the 0 character Hollerith code 001 010 (12 octal).

WRF

During the write file command, the WD outputs are not affected by the DA levels from the Data Control 136. The WRF level from negative diode 3C23 (–WT:B4) is applied to terminal P of inverter 3C18. The WRF level causes four inverters at location 3C18 to ground the four low-order WD outputs. The remaining two WD outputs are held negative by diode 3D22. The WRF level causes the WD outputs to assume the configuration of the file mark character 001 111 (17 octal).

END OF RECORD WRITE DELAY

The 4303 Integrating Single Shot EWD (–WT:C7) controls the interval between the final write pulse of each WRT or WRF record and the EOR 2 pulse which produces the EOR mark (the longitudinal parity check character). The EWD delay is adjusted to write the EOR mark four character spaces beyond the final data character of the record. The delay varies inversely with the density of characters per inch of tape. Any one of three motion delay resistors can be gated into the single shot circuit; each of them produces a delay period appropriate to one of the three available character densities, D200, D556, or D800.

Triggering of EWD

The delay period is started by setting EWD to the 1 state. The EWD(1) level resets EWC and WES, terminating both the write pulses and the TK/GV LT character request command pulses (refer to Stop Sequence described previously). During the WRF command, the WCP which writes the last, and only, data character of the file mark record is triggered by BRP (–WT:A4). The same BRP pulse starts the EWD delay. The EWD expires four character spaces later, producing the correct 4-character EOR gap between the file mark character and the EOR mark at the end of the file mark record. During the WRT command, the final TK/GV LT pulse of the record sets the EWD delay.

The DA RQ(1) signal (–WT:D6) from the Data Control 136 distinguishes the final TK/GV LT pulse from the preceding TK/GV LT pulses. The Data Control 136 sets its DA RQ status bit to 1 slightly more than 1 μsec after it receives the final TK/GV LT command pulse of each 6-character data word. Provided that additional data words follow, the data control then resets DA RQ to 0 approximately 1 μsec after setting it to 1. If DA RQ is still in the 1 state 2.4 μsec after a given TK/GV LT pulse, this indicates that no further data words are available to be written out; i.e., that the given TK/GV LT command pulse is not only the final pulse of a data word, but further that it is the final pulse of the current record.

All of the TK/GV LT pulses applied to input K of pulse amplifier 3C24 are delayed 2.4 μsec by the chain of three pulse amplifiers 3C24 and 3C25. This allows sufficient time to determine whether or not the data control will keep DA RQ set to 1. The write pulse that writes the last character of the record precedes the final TK/GV LT pulse of the record by one-half a

character interval (refer to Stop Sequence). Therefore, the total record gap delay is one-half a character interval plus 2.4 μsec greater during the WRT command than during the WRF command.

Output of EWD

At the expiration of the EWD delay, the EWD resets itself to the 0 state and causes pulse amplifier 3D19 (–WT:A7) to generate the EOR 2 pulse. The EOR 2 pulse causes the writing of the EOR mark (the longitudinal parity check character). The EOR 2 is sent to the transport interface as START 1; at present EOR 2 is not used within the tape control. At each START 1, all flip-flops of the write buffer are cleared. The clear complements just those flip-flops that are in the 1 state after the final character of the data record; i.e., just those flip-flops that have already been complemented an odd number of times. Because each time a write buffer flip-flop is complemented the transition causes a 1 to be written on tape, the START 1 clear writes a longitudinal parity check character which yields an even parity for each channel of the record (including the parity channel). Pulse amplifier 3D19 also produces an EOR 2 at MCL as an extra safeguard to ensure that each record begins with every write buffer flip-flop reset to 0.

WRITE COMMANDS

There are three write commands, WBT, WRF, and WRT. The write blank tape command advances and erases about 3 inches of tape. The write file command causes the writing of a record containing only the single file mark character 17 octal followed by an EOR gap and an EOR mark. The write command writes whatever data record is sent to the tape control from the Data Control 136 and generates an EOR gap followed by an EOR mark. All three write commands cause the transport to read the same data that it is writing. This concurrent read operation terminates the write and write file commands by generating an EOR 1 pulse when the EOR gap is sensed. The write blank tape command is also terminated by the EOR 1 pulse, but for the WBT command EOR 1 is produced by the BRP that marks the end of the UPS acceleration state rather than by sensing the EOR gap. The generation of the EOR 1 pulse is described in the read logic chapter (Chapter 6, End of Record).

Write Blank Tape (Figure 5-1)

The WEF level is present during each WBT command and causes the application of the WRITE level to the transport interface. Since neither NOP, REW, nor RSB is asserted, a FOW level is also applied to the transport interface for WBT commands (–CS:A1-2). Although the WRITE signal is applied to the transport interface during the WBT command, no write pulses are sent to the interface and no data is written. The write amplifiers are turned on, but the write-head polarity is not switched. The write heads produce the same sense of magnetization throughout the entire length of the tape.

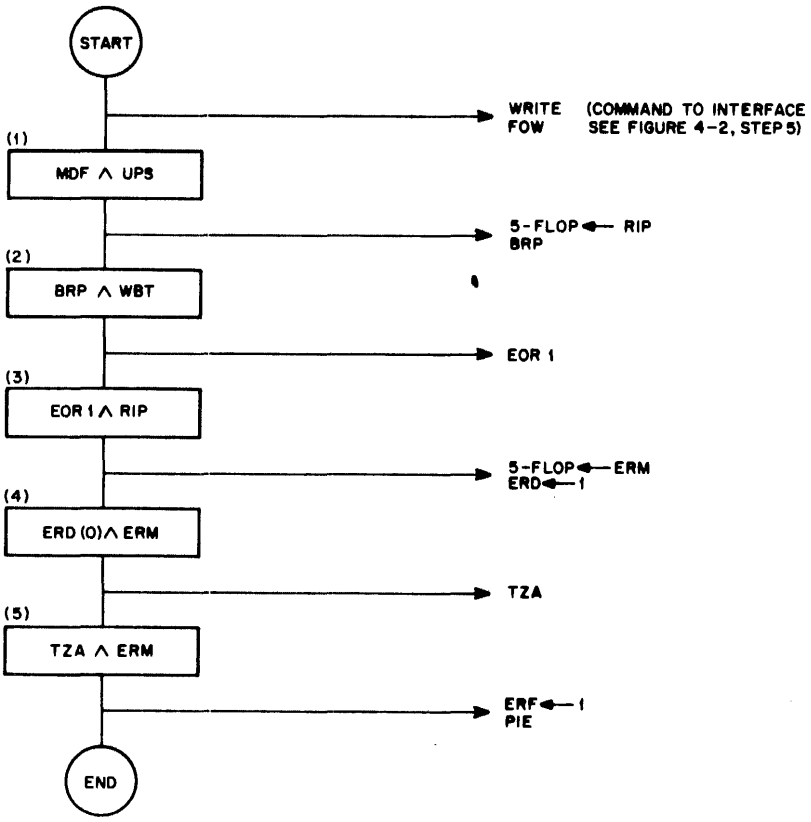


Figure 5-1 Write Blank Tape - WBT

During UPS state of the tape status quint-flop, the transport accelerates the tape. At the end of UPS state, the quint-flop is advanced to RIP, and a BRP pulse is generated (Figure 5-1, step 1). During the write blank tape command, the BRP directly generates the EOR 1 pulse (Figure 5-1, step 2). This EOR 1 causes the tape status quint-flop to advance from RIP to ERM state (Figure 5-1, step 3). The ERF flip-flop will be set to a 1, 3.5 character spaces

after EOR 1, asserting PIE (Figure 5-1, steps 4 and 5). Because the WRT level is not asserted during WBT, the BRP has no effect on the WES flip-flop, and no write clock pulses can be generated (–WT:B3).

The remainder of the command cycle is shown in Figure 3-2, step 13, etc.

Write File (Figure 5-2)

The WEF level causes the application of the WRITE level to the transport interface. A FOW level is also applied to the transport interface for WRF commands (–CS:A2). During UPS state of the tape status quint-flop, the transport accelerates the tape. At the end of UPS state, the quint-flop is advanced to RIP, and a BRP pulse is generated; Figure 5-2, step 1. The WRF level enables the BRP to produce a single WCP (–WT:A4) which is applied to the transport interface as a WP write pulse (step 2). This WP pulse causes the WD outputs to write the file mark character, 17 octal, on tape; refer to WRF, discussed previously in this chapter.

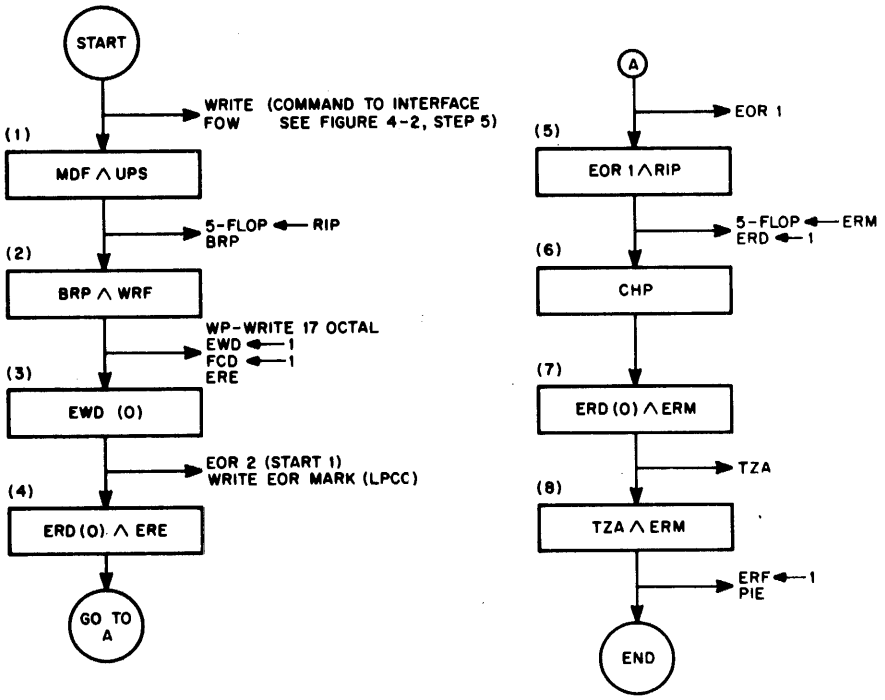


Figure 5-2 Write File - WRF

The BRP also sets EWD to 1 and begins the EOR gap delay. After the write heads write the file mark character, the read heads read it. The FCD flip-flop is set to 1, and the ERE level is asserted. The ERE level permits the generation of an EOR 1 pulse when the EOR gap is sensed. When the EWD times out, an EOR 2 pulse is generated and sent to the transport interface as START 1. The START 1 writes the EOR mark (longitudinal parity check character) on the tape (step 3).

The WRF command is terminated by the generation of an EOR 1 pulse. The read logic ERD is set each time a character is read and is reset only if the tape displays an empty interval of at least 3.5 character spaces with no characters to be read. Provided that the ERE level is present, the reset of ERD generates an EOR 1 pulse. During the WRF command, the ERD times out and is reset to 0 when the read heads sense the EOR gap (which is four character spaces long). The ERD times out before the EOR mark is reached, producing the EOR 1 pulse (step 4). The EOR 1 pulse advances the tape status quint-flop to ERM state (step 5). The EOR 2 pulse that is generated prior to when the EOR mark is reached sets the ERD delay to 1 (step 6). One μsec after the ERD times out and is reset, pulse amplifiers 3D4 (–RD:B2) produce a TZA pulse (step 7). Because the quint-flop is in ERM state, that TZA pulse sets the end-of-record flag ERF (–MD:B6). Provided that the EFE flip-flop contains 1, the ERF set produces a priority interrupt enable signal PIE (Figure 5-2, step 8). The remainder of the command cycle is shown in Figure 3-2, step 13, etc.

Write (Figure 5-3)

The WRT and FOW levels are applied to the transport interface. During the UPS state of the tape status quint-flop, the transport accelerates the tape. At the end of the UPS state, the quint-flop is advanced to RIP, and a BRP pulse is generated (Figure 5-3, step 1).

The WRT level enables the BRP pulse to set the WES flip-flop (step 2). The first CKP that arrives in time sets the EWC flip-flop to 1 (step 3). The next CKP, and every succeeding CKP that occurs while EWC(1) is still asserted, complements CF. These CKP pulses alternately write the WD output data on tape (step 4) and apply TK/GV LT data request command pulses to the Data Control 136 (step 5).

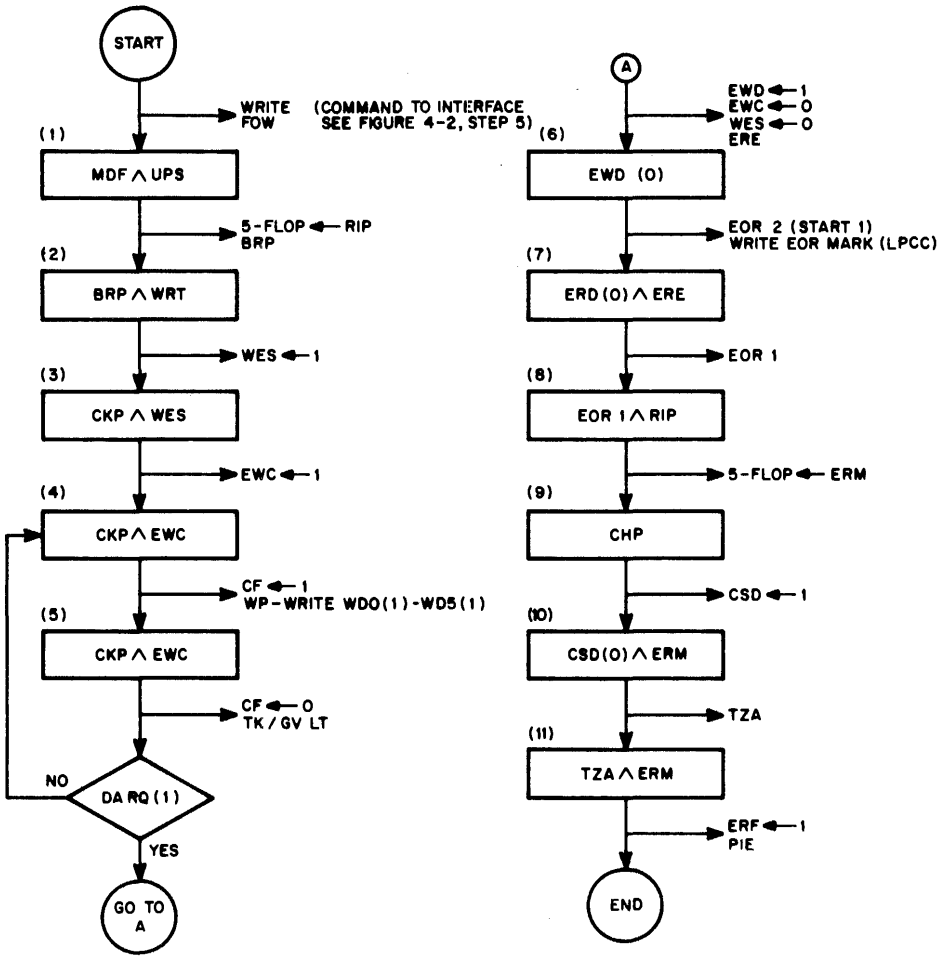


Figure 5-3 Write - WRT

This process continues as long as there is data to be written. When the DA RQ status bit in the Data Control 136 is finally set to 1 (refer to Triggering of EWD, described previously), the end of the current record has been reached and there is no more data to write. The WRT command escapes from the loop represented by steps 4 and 5 of Figure 5-3 and proceeds to execute the remaining operations shown below the DA RQ(1) decision box.

The final TK/GV LT pulse of the record causes the EWD delay to be set to 1. The set of this delay, in turn, resets the EWC and WES flip-flops, signalling that the last data character of the record has been written. The remaining steps of the WRT command (step 6 through 11) are identical to the final steps of the WRF command described under Write File.

CHAPTER 6

READ LOGIC

FUNCTION OF READ LOGIC

The read logic is shown in drawing –RD. It controls the execution of the five read commands: RED, SPF, BER, SPB, and RDC shown on –CM (read forward, space forward, backward read, space backward, and read compare).

The read and space forward commands are almost identical. In both cases, the tape transport reads forward one record while applying the resulting input characters to the tape control. The only difference between the two commands is that TK/GV LT command pulses are applied to the Data Control 136 only for the read command, but not for the space forward. As a result, during the space forward command, the data is never read into the data control accumulator.

The backward read and space backward commands are also almost identical. In both cases, the tape transport reads backward one record while applying the resulting input characters to the tape control. The only difference between the two commands is that TK/GV RT command pulses are applied to the Data Control 136 only for the backward read command, but not for the space backward. As a result, during the space backward command, the data is never read into the data control accumulator.

During the read compare instruction, the tape control compares an output record from the Data Control 136 with an input record read from the tape. The comparison is executed on a character-by-character basis, and at the first discrepancy the RCE flip-flop is set to 1. Whether or not RCE is set in the course of the record, the comparison proceeds until reaching the end of the input record being read from the tape.

INPUT LEVELS

The pulse 0⟶RB (–RD:C1) clears the interface read buffer at the beginning and end of each command, preparing it for the 1s transfer of the following character to be read. After a

character has been read from the tape and stored in the read buffer, the read buffer contents are applied to the tape control.

The ground-assertion 1 outputs of the six read buffer data bits RB0-5 are applied to output buffer inverters within the transport interface, and the inverter outputs are, in turn, applied to the tape control as RBB0(1)-RBB5(1) (–RD:C2-3). Negative levels represent 1s in the corresponding read buffer bit positions. The RBB inputs are applied directly to the complement inputs of the LPA to accumulate the longitudinal parity of each of the six data channels (–RD:B2-5).

The RBB levels also provide the input data to the Data Control. For all characters except one, the RBB levels are applied to the data control just as they arrive from the interface reader buffer. The single exception is the Hollerith code 0-character (12 octal). If that character arrives from the transport interface during a BCD parity mode read command, it is always reconverted to 00 octal before being read into the Data Control 136. For further information indicating why this reconversion is required, refer to Chapter 5, WRT and RDC.

LATERAL PARITY

The tape read parity error level TRPE indicates that a lateral parity error has been detected for the character that has just been read into the read buffer. The TRPE level is sent from the transport interface to the tape control (–RD:B3). Provided that MDC(1) is asserted at RCP time, the tape control sets the parity error flip-flop PER in the transport interface (–RD:B3).

A TRPE level will be produced by an EOR mark with incorrect lateral parity, but the PER flip-flop is only set when the character displaying the lateral parity error is a data character. Consequently, when PER is set to 1, it indicates that a real lateral parity error has been detected. The MCD(1) condition prevents PER from being spuriously set by an EOR mark (which is not required to have correct lateral parity). The MCD delay is described under Miss Character Detection later in this chapter.

If the file mark is read in binary parity, through a programming error, the PER flip-flop is set to 1. The EFF(1) level causes the PER flip-flop to be reset to 0 at EOR 1, removing this unwanted parity error indication. The EFF flip-flop is located in the transport interface; its function is to sense the end-of-file character; refer to EFF Set and Reset later in this chapter.

CHARACTER SENSING

The purpose of the character sensing logic is to generate a read character present pulse RCP (–RD:B8) whenever a data character has been completely read into the read buffer in the transport interface. This is done in the following manner. The RBCP input signal from the transport interface (–RD:D4) indicates that there is a 1 present in at least one of the five data bits RB0-5 of the interface read buffer. The RBCP level is ORed with the parity bit RB6(1) to produce the character present level CHP. The CHP level is first asserted when the first 1 bit of the current character is read; it terminates when the read buffer is reset.

The commencement of CHP sets the character skew delay CSD (–RD:B7), a 4303 Integrating Single Shot. The CSD provides a sufficient delay to ensure that even a character having the worst permissible degree of skew will have been completely read by the time the delay times out. Any one of three delay resistors can be gated into the CSD circuit; each produces a delay period appropriate for one of the three character densities: D200, D556, or D800.

At the termination of the CSD delay, the single shot output is reset to 0. This reset produces the 0 $\longrightarrow$ RB pulse which clears the interface read buffer (–RD:C1). Furthermore, if the tape status quint-flop is in RIP state (indicating that the current character is a data character and not an EOR mark), the reset of CSD also causes the generation of an RCP pulse. The RCP pulse indicates that a data character has been completely read into the interface buffer.

During the read command, the RCP pulse triggers the TK/GV LT command pulse that causes the Data Control 136 to sample the data character in the read buffer (–CS:A6). During the backward read command, the RCP pulse triggers the TK/GV RT pulse (–CS:B4). For the read compare command, the RCP pulse applies the RBB input levels to the LPA and also initiates the triggering of the TZA pulse (–RD:B1).

MISS CHARACTER DETECTION

The tape control sets the miss character flip-flop MIS (–RD:A6) whenever either one or two data characters have been missed while reading a record or during the read operation that occurs when writing. If three or more characters in succession are missed, the tape control will sense the gap as an end of record. (There is no way of detecting a character or characters that are

missing from the very end of the data record, but this condition is statistically unlikely and not worth the additional circuitry which would be required to detect it.)

The miss character condition is detected by the miss character delay MCD, a 4303 Integrating Single Shot (–RD:C7). While EMC is asserted, the MCD is set at each CHP pulse. Because of the FER(0) condition required to produce EMC, the MCD is set only by data characters and never by the EOR mark (refer to Second Character Detection later in this chapter for further description of the FER flip-flop). The MCD delay is set to time out at an interval of 1.5 character spaces. If no new CHP occurs within the 1.5-character space interval, it is assumed that a character has been missed.

The MCD then resets itself to 0 and sets the MIS flip-flop (–RD:A6). If the MDC reset was a spurious miss character indication caused by the EOR gap, the MIS flip-flop is reset by the EOR 1 pulse. If this is not the case, the next RCP causes the MCS flip-flop (–RD:A7) to be set, indicating that a character (or two characters) have really been missed. The set of MCS confirms the miss character indicating by locking the MIS flip-flop in the 1 state so that it can no longer be reset by the EOR 1 that occurs at the end of the current record.

READ COMPARISON

The tape control sets the read compare error flip-flop RCE (–RD:B5) to 1 if there is any discrepancy between the output record from the Data Control 136 and the input record from the tape. The actual comparison takes place in the longitudinal parity accumulator LPA (–RD:A2-5) on a character-by-character basis.

The ERC level enables the CSD set (–RD:B2) to generate the read compare strobe pulse RCS. During RDC commands, ERC is asserted throughout the RIP state of the tape status quint-flop (–RD:C5). The ERC level is removed when the quint-flop advances to ERM state to prevent the CSD set corresponding to the EOR mark from producing an RCS. The RCS pulse strobes the write data outputs WD0(1)-WD5(1) into the corresponding LPA flip-flops LPA0-5. The RCS pulse also produces a TK/GV LT pulse which causes the Data Control 136 to advance the next output character into position for read comparison.

When CSD times out and is reset to 0, the resulting RCP applies the RBB0(1)-RBB5(1) input data levels to the complement inputs of the LPA and 1 μsec later generates the TZA pulse (–RD:B8 and B2). The LPA is completely cleared if the WD output character from the data control is identical to the RBB input character from tape. However, if the two characters differ in any bit position, then some bits of LPA are left in the 1 state, and the LPA = 0 level (–RD:D3) is not asserted.

The negation of LPA = 0 enables the TZA pulse to set the RCE flip-flop (–RD:C5). This indicates that the read comparison has failed. After detecting a read compare error, the RCE flip-flop remains in the 1 state until it is reset by the MCL at the beginning of the next command. The LPA is cleared by TZA after each character comparison and is also cleared at the beginning of each command by MCL.

END OF RECORD

The tape control sets the ERF flip-flop to 1 when the current command has reached the end of the record (–MD:B6). During all commands except RWD, ERF is set by the TZA pulse occurring in the ERM state. When the EFE flip-flop contains 1, ERF(1) is a sufficient condition for generating the priority interrupt enable level PIE (–MD:C7). The MCL pulse resets ERF at the beginning of each command.

EOR 1 Pulse

The EOR 1 pulse must always be generated before ERF can be set. During forward commands, EOR 1 steps the tape status quint-flop from RIP to ERM, and the ERM level enables TAZ to set ERF.

Pulse amplifier 3D8 generates EOR 1 under three sets of conditions (–RD:C8). For the write blank tape command, an EOR 1 is initiated by BRP at the completion of UPS acceleration state. Should the tape reach load point during an RSB command, an EOR 1 is generated and ERF is set. The third, and most frequent condition for generating the EOR 1 pulse is the reset of the ERD delay in conjunction with the ERE level.

End of Record Delay and ERE Level

The end of record delay ERD, a 4303 Integrating Single Shot (–RD:C7) is set at each CHP while RIP is asserted. The delay is set to time out an interval equal to 3.5 character spaces at the current character density. If no new CHP occurs within the delay interval, the ERD resets itself to 0, producing an EOR 1 pulse if ERE is asserted. During forward commands, ERD is not set by the EOR mark because RIP is not present at the time the EOR mark is read. During RSB commands, the EOR mark does set ERD, but because ERE is not asserted at the subsequent ERD reset, no EOR 1 pulse is produced.

During RSB commands, the ERE level is not present when the EOR mark is read because the first character detect flip-flop FCD has not yet been set (–RD:A4). There are two other input conditions required for ERE. The RIP condition prevents the generation of an EOR 1 while the ERD delay is timing out to sample LPA parity accumulation. The CHB(0) condition prevents the generation of EOR 1 when a CONO instruction has set the disable-EOR 1 maintenance bit CHB32 to 1. (The tape control then treats the entire tape as essentially a single record.)

LONGITUDINAL PARITY

In addition to the lateral parity test described previously under Lateral Parity, the tape control also tests for correct longitudinal parity. For all commands except RDC and REW, the longitudinal parity is checked at the end of each record by testing the contents of LPA. Errors are indicated by setting the longitudinal parity error flip-flop LPE to 1 (–RD:B4).

The RBB0(1)-RBB5(1) data input levels are strobed into the complement inputs of the LPA at each RCP pulse and also at the ERD reset corresponding to the EOR mark (–RD:B1). This accumulates the longitudinal parity of each of the six data channels. During either backward or forward commands, the correct longitudinal parity demands an even number of 1 bits in each of the six data channels. If this requirement is met, all six bits of LPA are left cleared at the completion of the record.

The LPE flip-flop is reset by MCL at the beginning of each command. If any bit of LPA contains 1 at the end of the record, a longitudinal parity error is indicated and the LPE flip-flop is set to 1. When the error occurs during an RSB command, LPE is set by diode 3D13; during

forward commands, diode 3D9 sets LPE. For RSB commands, the state of LPA is tested by the EOR 1 pulse that follows the first data character of the record (i.e., the last character to be read). For all forward commands except RDC, the LPA is tested by the TZA pulse that occurs during the ERM state of the tape status quint-flop (the TZA initiated by the EOR 1 pulse and ERD reset).

END OF FILE

The tape control sets the end-of-file flag EFF in the transport interface whenever a file mark is sensed. Termination of the EOF(0) level ends the SEF level (–CM:D1) and permits the tape control to escape from the space to end-of-file loop; see Chapter 4, Space to End-of-File Command - CON(1) ∧ SEF. The EFF(1) level resets the PER flip-flop at EOR 1, removing the false lateral parity error indication that is generated if the file mark is read in binary parity.

Second Character Detection

A file mark is always the first and only data character of a file mark record. If a second data character is sensed, the record cannot be a file mark record. The FCD (first character detect) flip-flop (–RD:A7) is set to 1 by the first data character of each record. The SCD (second character detect) flip-flop (–RD:A8) is set by the second data character of each record. This occurs whether the current command is a forward command or a reverse command. However, for RSB commands, the FER flip-flop (–RD:D3) (Figure 3-6:D3) prevents the EOR mark from setting the FCD flip-flop. The FER flip-flop is also used to prevent the assertion of EMC during RSB commands (see previously described Miss Character Detection) to prevent the EOR gap from setting MCD.

EFF Set and Reset

The interface sends an RBEF level to the tape control whenever a 17 octal character is read. If RB6(0) is also asserted, indicating BCD (even) parity and if the character is the first data character of the current record (SCD = 0), the character is treated as a file mark and an EOF level is generated (–RD:D5). The EOF enables RCP to set EFF (–RD:B2). If additional data characters are in the record, SCD(1) is asserted, and the EFF is reset by the EOR 1 at the end of the record. The EFF is always reset by MCL (STATUS CLEAR) at the beginning of each command.

READ COMMAND AND SPACE FORWARD COMMAND

The RED and SPF commands (Figure 6-1) are identical except that no TK/GV LT pulses are sent to the DATA Control 136 during the SPF. A FOW level is applied to the transport interface for both commands.

During UPS state, the transport accelerates the tape. At the end of UPS, the tape status quint-flop is advanced to RIP state; see Figure 6-1, step 1. As each data character is sensed, CHP sets the three read logic delays CSD, MCD, and ERD (step 2). The expiration of CSD produces an RCP after each character has been read completely (step 3). During the RED command, but not the SPF, each RCP produces a TK/GV LT pulse. Each of these TK/GV LT pulses causes the Data Control 136 to strobe the character currently being read into the six low-order bits of its data accumulator (at the same time, the accumulator is shifted left six bit-positions). The CSD reset also produces the 0 $\longrightarrow$ RB pulse to clear the read buffer in the transport interface.

The RCP that is produced by the first data character of the record sets the first character detect flip-flop FCD (step 4). The next RCP sets the second character detect flip-flop SCD (step 5). The tape control then continues to cycle through the loop containing steps 2 and 3 until reaching the EOR gap at the end of the record. At the EOR gap, the MCD delay times out and then the ERD delay times out. The MCD reset causes the MIS flip-flop to be set to 1. The ERD reset produces an EOR 1 pulse. This EOR 1 pulse resets MIS, and steps the tape-status quint-flop from RIP to ERM state (step 6). EOR 1 resets the ERD delay that remains set while the EOR mark is read.

The CHP produced by the EOR mark does not set CSD to 1 (step 7). Instead, the ERD acts as a skew delay while the EOR mark is accumulated. One μsec after the ERD times out and is reset, a TZA pulse is generated (step 8). If a longitudinal parity error has occurred during the command, this TZA sets the LPE flip-flop to 1. Because the quint-flop is in ERM state, the same TZA pulse sets the end-of-record flag ERF (–MD:B6). Provided that the EFE flip-flop contains 1, the ERF set produces a priority interrupt enable signal PIE (Figure 6-1, step 9). The remainder of the command cycle is shown in Figure 3-2, steps 13, etc.

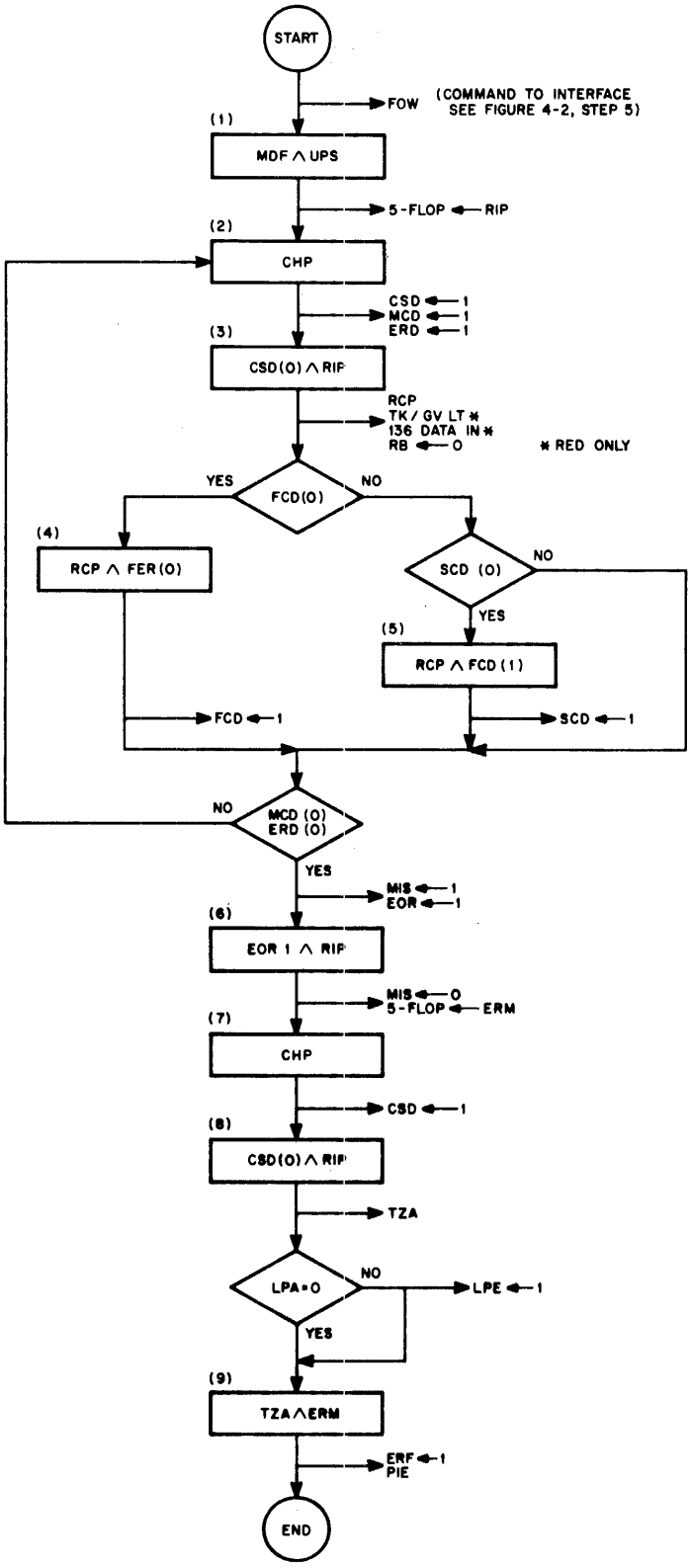


Figure 6-1 Read (RED) and Space Forward (SPF)

BACKWARD READ COMMAND AND SPACE BACKWARD COMMAND

The BER and SPB commands (Figure 6-2) are identical except that no 0 ⟶ CCT pulse and no TK/GV RT pulses are sent to the Data Control 136 during the SPB. A BAC level is applied to the transport interface for both commands.

During UPS state, the transport accelerates the tape. At the end of UPS, the tape status quint-flop is advanced to RIP state, and a BRP pulse is generated, setting the false end of record flip-flop FER; see Figure 6-2, step 1. The first character that is sensed during the BER and SPB commands is not a data character, but the EOR mark. The CHP generated by the EOR mark sets the CSD and ERD delays, but not the MCD delay (step 2).

The expiration of CSD produces an RCP (step 3). The RCP resets the FER flip-flop and causes EMC to be asserted. Assertion of the EMC level permits the MCD delay to be set by all subsequent data characters of the record. During the BER command, each RCP, including the EOR mark RCP, produces a TK/GV RT pulse. The CSD reset also produces the 0 ⟶ RB pulse to clear the interface read buffer.

After reading the EOR mark, the transport read heads traverse the EOR gap. The ERD delay times out; but because the FCD flip-flop contains 0, ERE is not asserted, and no EOR 1 pulse is produced. The reset of ERD does produce a 0 ⟶ CCT pulse (–CS:B5) during the BER command (Figure 6-2, step 4). This pulse prevents the Data Control 136 from treating the EOR mark as data.

The first data character to be read is the last character of the record. The CHP produced by this character sets all three of the read logic delays CSD, MCD, and ERD (step 5). The expiration of the CSD produces an RCP after each character has been completely read (step 6). During the BER command, but not the SPB, each RCP produces a TK/GV RT pulse that causes the Data Control 136 to strobe the character currently being read into the six high-order bits of its data accumulator (at the same time the accumulator is shifted right six bit-positions). The CSD reset also produces the 0 ⟶ RB pulse to clear the interface read buffer.

The RCP that is produced by the last data character of the record (the first character to be read during a BER or SPB command) sets the FCD (step 7). The next RCP sets the SCD (step 8).

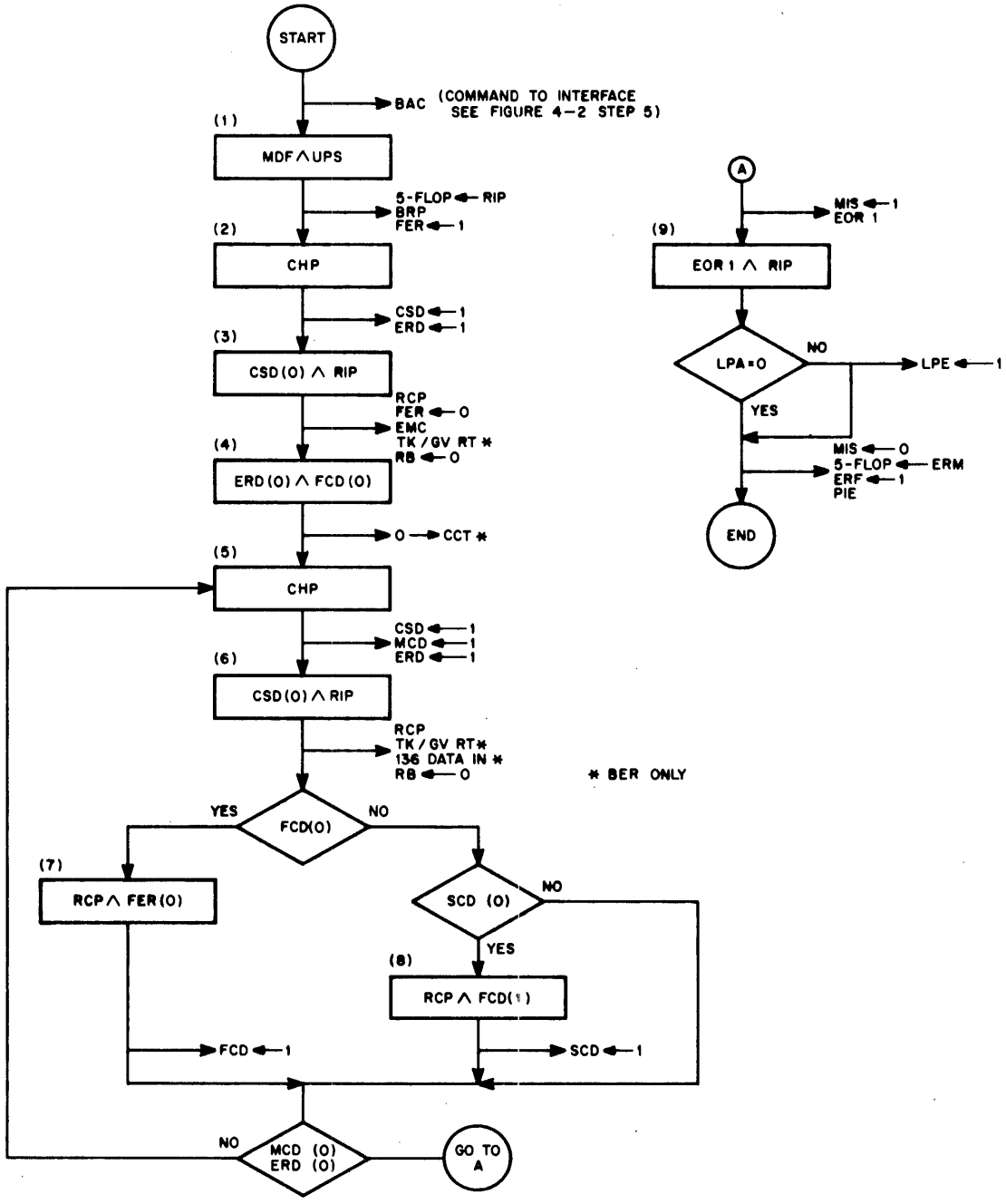


Figure 6-2 Backward Read (BER) and Space Forward (SPF)

The tape control then continues to cycle through the loop containing steps 5 and 6 until reaching the beginning of the record. After the first character of the record is read, the tape continues backward into the record gap (not the EOR gap; that is traversed at the beginning of the command).

At the record gap, the MCD delay times out and the ERD delay times out. The MCD reset causes the MIS flip-flop to be set to 1. The ERD reset produces an EOR 1 pulse. This EOR 1 pulse resets MIS, steps the tape status quint-flop from RIP to ERM state, and resets ERD. Thus, the ERF flip-flop is set seven character times after reading the last character during RSB or 3.5 character times after EOR 1. Provided that the EFE flip-flop contains 1, the ERF set produces a priority interrupt enable signal PIE (Figure 6-2, step 11). Likewise, if a longitudinal parity error has occurred during the command, the LPE flip-flop will be set when ERF is set. The remainder of the command cycle is shown in Figure 3-2, steps 13, etc.

READ COMPARE COMMAND

A FOW level is applied to the transport interface for the RDC command (Figure 6-3). During UPS state, the transport accelerates the tape. At the end of UPS, the tape status quint-flop is advanced to RIP state and ERC is asserted; see Figure 6-3, step 1. As each data character is read, CHP sets the three read logic delays CSD, MCD, and ERD (step 2). The set of CSD produces an RCS pulse that transfers the WD outputs into LPA0-5 and generates a TK/GV LT pulse (step 3). By shifting the data control accumulator left six bit-positions, the TK/GV LT brings the next output character into position to be sampled by the tape control.

When the CSD delay times out, an RCP pulse is generated, applying the RBB input levels to the complement inputs of the LPA (step 4). The resulting data in LPA is the exclusive OR of the WD outputs and the corresponding RBB inputs; i.e., each bit of LPA contains the WD output or its complement, depending upon whether the RBB input level represents a 0 bit or a 1. One μsec after the RCP, a TZA pulse is generated, testing the state of LPA and simultaneously clearing LPA. If the LPA contains any 1 bits, the read compare error flip-flop RCE is set to 1.

The tape control continues to cycle through the loop containing steps 2 through 5 until reaching the EOR gap at the end of the record. At the EOR gap, the MCD delay times out and then

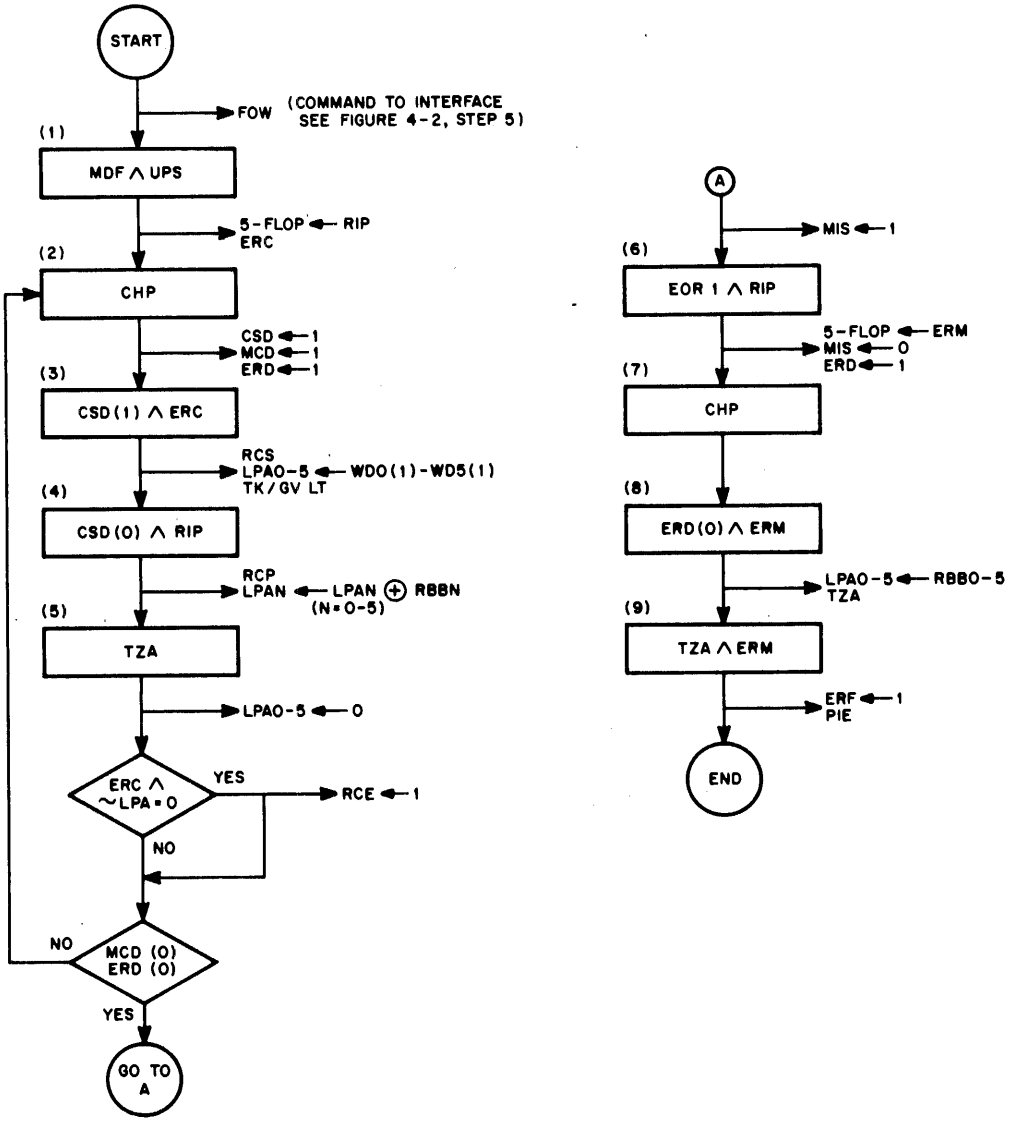


Figure 6-3 Read Compare (RDC)

the ERD delay times out. The MCD reset causes the MIS flip-flop to be set to 1. The ERD reset produces an EOR 1 pulse. This EOR 1 pulse resets MIS and steps the tape status quint-flop from RIP to ERM state (step 6). EOR 1 resets the ERD delay that remains set while the EOR mark is read.

The CHP produced by the EOR mark does not set the CSD to 1 (step 7). The subsequent reset of ERD (step 8) causes the EOR mark to be applied to the complement inputs of LPA0-5, and 1 μsec later produces a TZA pulse. However, because ERC is no longer asserted after the

quint-flop enters ERM state, this TZA pulse has no effect on the state of the RCE flip-flop. The transfer of the EOR mark into the LPA is, therefore, harmless for the RDC command. (Moreover, LPA is cleared by MCL at the beginning of the subsequent command.) The TZA pulse produced by the reset of ERD sets the end of record flag ERF (–MD:B6). Provided that the EFE flip-flop contains 1, the ERF set produces a priority interrupt enable signal PIE (Figure 6-3, step 9). The remainder of the RDC command cycle is shown in Figure 3-2, steps 13, etc.

CHAPTER 7

TAPE CONTROL INTERFACE 520

The 520 Interface permits the Magnetic Tape Control 516 to control up to eight DEC Tape Transports Type 50. The interface contains write and read buffers, write and read parity networks, and associated logic. The end-of-file flag and lateral-parity-error flag are also located in the interface, as are the drivers that amplify command and selection levels from the tape control and apply these levels to the tape transports.

SELECTION AND COMMAND INPUTS

Eight unit selection levels UNIT 0-UNIT 7 are also applied to the interface from the tape control (BS-D-520...:D1). The selection level corresponding to the selected tape transport is asserted at ground; the other seven levels remain negative. All eight levels are amplified by 1684 Bus Drivers before being applied to the transports at T UN 0-7. The tape control always selects a transport before applying any command levels to the interface. One of the eight unit selection levels is grounded at the tape control JNU pulse; the command levels are not applied to the interface until the tape control generates a JNC pulse.

Four input command levels are applied to the interface from the tape control: WRITE, FOW, BAC, and REW (-520...:C1). The WRITE level is asserted at ground; the remaining three levels are negatively asserted. The WRITE level is amplified by a 1684 Bus Driver and applied to the tape transport as T WRITE. This signal sets the transport to write status by enabling both the write logic and the read logic. (The capacitor from T WRITE to ground is used to prevent switching noise from reaching the transport.) The WRITE level is also inverted and amplified by a 1690 Inverting Bus Driver, and the resulting signal is applied to the tape transport as T READ. Unless REW is also asserted, the T READ signal sets the transport into the read status, disabling the write logic and enabling the read logic.

The FOW, BAC, and REW levels are inverted and amplified by 1690 Inverting Bus Drivers, and the resulting ground assertion signals are applied to the tape transport as T FOW, T REV, and T REW. The T FOW signal causes the transport to begin forward tape motion; T REV starts

reverse tape motion at normal reading speed; the T REW signal causes the transport to reverse at rewind speed (approximately twice the normal reading speed). The read amplifiers are not enabled during rewind.

WRITE LOGIC

The write logic accepts the write data outputs from the tape control, generates a lateral parity bit, and complements those bits of the write buffer that correspond to 1 bits of the character being written. Both outputs of each WB flip-flop are applied to the transport write logic.

Write Data Outputs

During WRT and WRF commands, the six write data outputs WD0-WD5 apply output data characters from the tape control to the interface (-520...:B1-3). Ground levels correspond to 1 bits. The WD levels determine which of the data flip-flops in the write buffer are to be complemented at the write pulse. Both polarities of the six WD outputs are applied to the write parity network. The parity network then generates a seventh level which determines whether or not the write buffer parity flip-flop is to be complemented at the write pulse.

Write Parity Net

As each 6-bit output character is transferred out to the interface by the WD outputs, the write parity net (-520:B and C4-7) generates an accompanying parity bit. If CB21 (in the tape control command buffer) contains 0, the writing is to be done in BCD (even) parity. The 0 state of CB21 causes a negative PAR 0 input to be applied to the write parity net (-520...:C7). This input causes the net to generate a negative WPAR 1 output if an odd number of the six WD outputs are 1s. The transport then writes a 1 bit in the parity channel for the current character, yielding a character with an even number of 1 bits. If CB21 contains a 1, the writing is to be done in binary (odd) parity. A ground PAR 0 input is then applied to the parity net, and the WPAR 1 output is generated only if an even number of the six WD outputs are 1s.

The write parity net is composed of a 3-stage, 7-input exclusive OR circuit. All of the WD inputs are negative assertion levels. The WPAR 1 output is asserted only if an odd number of the seven inputs are 1 (counting the six WD bits and the contents of CB21). The assertion levels

shown in the two upper stages of the net assume further that the PAR 0 input is asserted negative (i.e. that CB21 is 0, indicating BCD parity). In inverter 3B13, three external 3.3K resistors have been substituted for the normal 1.5K load resistors. This reduces the current through the lower legs of the parity net and avoids excessive loading.

The WEF command level is inverted in the tape control (-CM:C4) and applied to the write parity net as $\overline{\text{WRITE EOF}}$; (-520...:B5). When this level is negative (as it is for write file commands), diode 3B8 holds the WPAR output at ground, inhibiting the generation of a 1 parity bit. The end-of-file character 001 111 (17 octal) should always be written in BCD parity (with a 0 parity bit). The $\overline{\text{WRITE EOF}}$ input ensures that the write parity net produces the correct 0 parity bit for the EOF character even if the programmer should inadvertently give a WRF command with binary parity.

Write Buffer

The write buffer (-520...:A1-4) is composed of seven 500-kc flip-flops. Both outputs of each flip-flop are amplified by 1684 Bus Drivers and applied to the transport write amplifiers. Depending on the state of the associated WB flip-flop, each write amplifier drives one or the other of two oppositely-wound write-head coils. Each transition of a WB flip-flop terminates the current through one coil, and starts it in the other, changing the sense of tape magnetization, and writing a 1 in the corresponding tape channel. The T-3 driver-power control level (-520:A1) permits switching in resistors to −15 vdc to augment the bus driver power. At present, this feature is not used.

WB Complement

The data bits WB0-5 are complemented whenever the associated WD outputs are in the 1 state at the write pulse WP. The parity bit WB6 is complemented if the write parity level WPAR 1 is asserted at WP. When a given WD output is in the 1 state, a ground level is applied to the associated input terminal, and inverted (-520...:B1-3).

For WD0-4, the resulting negative level enables Capacitor-Diode 4127 to produce a positive output pulse at the arrival of the WP. This pulse complements the corresponding WB flip-flop

WB0-4. The complement inputs applied to WB5 and WB6 use 4113 Diodes instead of capacitor-diodes, but the result is logically identical. Data bit WB5 is complemented only when WD5 is a 1, and the parity bit WB6 is complemented only when the WPAR 1 level represents a 1 parity bit.

WB Clear

The START 1 input to the interface (-520...:A1) originates in the tape control as the EOR 2 pulse. This pulse clears all seven WB flip-flops each time it is applied to the interface. A START 1 is produced at the beginning of every command to ensure that every record begins with WB0-6 cleared.

Another START 1 is applied to the interface four to five character spaces after the last data character of each record is written on tape (4 spaces during WRF records, and 4.5 spaces + 2.4 μsec during WRT records). The resulting WB clear produces the EOR mark at the end of the record. The clear complements just those WB flip-flops that are left in the 1 state after the final data character of the record; i.e., those flip-flops that have already been complemented an odd number of times in the course of the record. This yields an even longitudinal parity for each of the seven channels in the record, and thus permits the EOR mark to be used as a longitudinal parity check character. (Only the six data channels are actually checked for longitudinal parity.)

READ LOGIC

The read logic accepts seven transport signals corresponding to each input data character, assembles the input character in the read buffer, tests it for the correct lateral parity, and applies it to the data control (via the tape control).

Read Buffer

The read buffer (-520:A4-7) is composed of seven 500-kc flip-flops; six data flip-flops RB0-5 and a lateral parity flip-flop RB6. Each of the RB flip-flops is set to 1 at the peak of every transport input signal on the corresponding read channel. The ground assertion 1 outputs of the six RB data flip-flops are applied to output buffer inverters within the interface, and the inverter outputs are, in turn, applied to the tape control as RBB0(1)-RBB5(1). Negative levels

represent 1s in the corresponding read buffer bit positions. The negative assertion 1 output of the RB parity flip-flop RB6(1) is applied to the tape control directly without inversion.

RB Clear

The read buffer is cleared by the 0 ⟶ RB pulse from the tape control. This clear occurs at the beginning of each command to ensure that the buffer starts reading in the correct initial condition. It also occurs after each character is read to prepare the read buffer for the 1s transfer of the next character. The latter clear occurs at CSD(0), the end of the character skew delay. At that time, the current character has been fully entered into the read buffer. The contents of the buffer are sampled at the same time that the buffer is cleared.

RB Set

At successive changes of tape magnetization, the differential amplifier at each of the transport read heads produces output signals of alternating polarity (the polarity of each signal depending on the sign of the change). Before being applied to the interface, the bipolar output signal for each of the seven tape channels is passed through a rectifying slicer in the transport. The slicer rectifies the output of the differential amplifier and produces a positive output pulse at each input of either polarity. However, no slicer output is generated unless the input exceeds a preset level. (This is to prevent low-level noise inputs from producing spurious output pulses.)

The negative output pulses from each of the seven transport channels are applied to a 1539 Peak Detector and Slicer circuit in the interface (-50:B4-7). When pulsed by an output from the rectifying slicer, each 1539 Peak Detector-Slicer generates two outputs, a negative level, and a negative pulse. The level and pulse are both applied to a 4127 Negative Capacitor-Diode Gate. The capacitor-diode then applies a positive output pulse to the complement input of the corresponding RB flip-flop (setting it to the 1 state, because it was cleared at the preceding 0 ⟶ RB pulse). The negative enable level is produced by the 1539 Circuit as long as the driving pulse from the transport is above a minimal amplitude. The pulse output of the 1539 coincides with the peak of the input signal.

Therefore, as each character is read by the transport, each of the seven RB flip-flops RB0-6 is set to 1 if there is a change in the magnetization of the corresponding tape channel. Since neither the write heads nor the read heads are ever aligned exactly, there is a finite time dispersion between the peaks of the 1 signals from different channels. The tape control character skew delay CSD is calculated to provide enough delay so that each character is fully entered into the read buffer before the buffer is sampled and cleared.

RBCP and RBEF

The RBCP level (read buffer character present) is asserted whenever any of the six read buffer data bits RB0-5 contain 1 (-520:C3). This level is sent to the tape control where it is ORed with RB6(1) to produce the CHP (character present) signal.

The RBEF read buffer end-of-file level (-520:C3) is asserted and applied to the tape control when the six data bits of the read buffer RB contain the file mark character (17 octal).

End-of-File Flag

The EFF flip-flop (-520:A7) is set to 1 to indicate that a file mark has been sensed. Although the EFF flip-flop is physically located in the interface, the signals controlling its set and reset originate in the tape control. The interface sends an RBEF level to the tape control whenever a file-mark character (17 octal) is read. If BR6(0) is also asserted, indicating BCD parity, and the character is the first data character of the current record, the EOF level is asserted, and the character is treated as a file mark.

At the RCP pulse (which coincides with the end of the CSD delay), the tape control applies a 1 $\longrightarrow$ EFF ground pulse to interface input terminal 3B81M, setting EFF. If additional data characters are in the current record, the file mark indication is assumed to be the result of a programming error, and EFF is reset at the EOR 1 pulse that ends the record. The tape control accomplishes this reset by applying a 0 $\longrightarrow$ EFF ground pulse to interface input terminal 3B88K. The EFF is also reset at the beginning of each command by STATUS CLEAR (designated MCL within the tape control).

Read Parity Net

As each input character is read into the read buffer RB0-6, the read parity net (-520:C and D4-8), checks the character for correct lateral parity. If the parity is not correct, a negative TRPE (tape read parity error) level is sent to the tape control.

The read parity circuit is virtually identical to the write parity net described previously under Write Parity Net, except that it contains one extra leg. Both the PAR 0 and PAR 1 outputs of the CB21 flip-flop are used as inputs to the read parity net. When binary (odd) parity is specified, the PAR 1 level is negatively asserted; and when BCD parity is specified, PAR 0 is negatively asserted. The RB inputs are also negative assertion levels. A parity error TRPE is asserted when PAR 0 is asserted and an odd number of the seven RB bits RB0-6 contain 1s; TRPE is also asserted when PAR 1 is asserted and an even number of the RB bits contain 1s.

Parity Error Flip-Flop

The parity error flip-flop PER (-520:A8) is set to 1 to indicate that a lateral parity error has occurred. The PER flip-flop is physically located in the interface, but the signals controlling its set and reset originate in the tape control. The interface sends a TRPE level to the tape control whenever the character currently being read displays an apparent parity error. If the character is a data character rather than an EOR mark, the tape control sets the parity error flip-flop by applying a ground pulse to interface input terminal 3B81N. However, if the character displaying the apparent parity error is an EOR mark, MCD(1) is not asserted at RCP time, and the parity error flip-flop is not set. (The EOR mark serves as the longitudinal parity check character and may or may not have correct lateral parity. For this reason, it is necessary to prevent any TRPE signals that are produced by EOR marks from setting the PER flip-flop.)

If the EFF is set to 1, it causes the PER flip-flop to be reset to 0 at EOR 1. This removes the false parity error that would appear if the file mark were read in binary parity through a programming error. The PER is also reset at the beginning of each command by STATUS CLEAR (designated MCL within the tape control).

CHAPTER 8

TAPE CONTROL INTERFACE 521

The 521 Interface permits the Magnetic Tape Control 516 to control up to eight DEC Tape Transports, Type 570. The interface contains a read buffer, write and read parity networks, and associated logic. The end-of-file flag and lateral-parity-error flag are also located in the interface, as are various control circuits that process command and selection levels from the tape control and apply these levels to the tape transports.

SELECTION AND COMMAND INPUTS

The selection and command inputs select a specific tape transport for the current command and apply appropriate command levels and pulses to that transport.

Selection Inputs

Eight unit selection levels UNIT 0-7 are applied to the interface from the tape control (-521:D1). The selection level corresponding to the selected tape transport is negatively asserted; the other seven levels remain at ground. All eight levels are sent through buffer inverters before being applied to the transports as T UNIT 0-7.

The tape control always selects a transport before applying any command levels to the interface. At the tape control JNU pulse, one of the eight unit selection levels becomes negative. The JNU pulse is applied to the interface as AC1 $\longrightarrow$ UN (-521:A1). That pulse is delayed 1 μsec by pulse amplifier 3B25 and amplified by pulse amplifier 3B24 and applied to the transports as UNIT STROBE. The UNIT STROBE pulse selects the transport specified by the asserted unit selection level. The command levels are not applied to the interface until the tape control JNC pulse.

Command Inputs

Three input command levels are applied to the interface from the tape control: REWIND, WRITE, and FOW (-521:C1). These inputs, together with the CON 1 level (from CB24 in the

tape control command buffer) determine the type of operation the transport is to commence when it receives a start pulse from the tape control. The WRITE level is asserted at ground; the other two command levels and the CON 1 level are negatively asserted.

The WRITE level is inverted and amplified by a 1690 Inverting Bus Driver, and the resulting signal is applied to the tape transport as T WRITE/READ. A negative T WRITE/READ sets the transport to write status by enabling both the write logic and the read logic. A ground T WRITE/READ prevents the write logic from being enabled. (The capacitors from T WRITE/READ and from T REW to ground are used to prevent switching noise from reaching the transport.)

The FOW level is also inverted and amplified by a 1690 Inverting Bus Driver, and the resulting signal is applied to the tape transport as T FOW/REV. At ground, that signal causes the transport to begin forward tape motion when the start pulse arrives; when negative, the signal calls for reverse tape motion at normal reading speed.

The REW level is amplified by a 1684 Bus Driver and applied to the transport as T REW. That signal causes the transport to reverse at rewind speed (about twice normal reading speed) when the start signal is given. If CB24 (in the tape control command buffer) contains 0, the transport rewinds only as far as the tape load point and stops. However, if CB24 contains 1, a rewind unload command is specified, and the entire tape is rewound. The REWIND COMMAND level and CON 1 (the negative assertion 1 output of CB24) are ANDed and amplified by diode 3B13 and inverter bus driver 3B21 and applied to the transport as T REW/UNLOAD (-521:D3 and C2).

OTHER CONTROL FUNCTIONS

The present paragraph describes the start sequence, the process of reserving a transport or releasing it to pool (when operating in multicontrol mode), and the use of the T A/B signal.

The tape control CTT pulse is sent to the transport interface as IOT 7104, and returns from the interface to the tape control designated START 2 (-521:C5). If the tape control command buffer bits CB25-27 are all 0, specifying no operation (NOP), the IOT 7104 pulse has no effect on the interface or transports. However, if CB25-27 are nonzero, indicating that the current command calls for tape motion, the $\overline{\text{NOP}}$ level is asserted negative, and the IOT 7104 pulse sets the GO flip-flop.

The ground assertion 1 output of GO is inverted and amplified by a 1690 Inverting Bus Driver and applied to the transports as T GO. It causes the selected transport to begin the mode of tape motion specified by the command levels (FOW, REV, or REW). The GO flip-flop remains set until the tape control command buffer is cleared by the CCB pulse. The CCB is sent to the interface as 0⟶CM. It clears GO and terminates T GO. The ground assertion 1 output of the GO flip-flop is also used to enable the peak detectors in the read buffer set logic.

Pool

The 521 Interface and the associated 570 Transports can be operated in multicontrol mode. This means that the same transports can be used by two tape control units (each tape control having its own associated interface). There is a 3-state device in each transport which reflects transport status (A, B, or POOL). These transport-status conditions mean that the transport is currently reserved by the A tape control, that it is reserved by the B tape control, or that it is in the POOL condition (free to be selected).

If CB28 (in the tape control command buffer) contains 1, the transport is to be kept reserved after the completion of the current command. However, if CB28 contains 0, POOL 0 is asserted (-521:B1), and the transport is returned to pool at the conclusion of the current command. The reset of the GO flip-flop causes pulse amplifier 3B24 to generate a RETURN TO POOL pulse at output terminal P. The POWER CLEAR pulse (designated IOB RESET in the 516 Tape Control) also generates a RETURN TO POOL pulse. POWER CLEAR is applied directly to input terminal M of pulse amplifier 3B24; input terminal N is unused.

T A/B Select

If the current command selects a transport that is already reserved by the other control (the A or B control as the case may be), the reserved transport returns a T A/B signal to the interface (-521:D3). From the interface, the T A/B signal is routed to two places in the tape control. Designated BCT, it is applied via interface output terminal 3B95C to the tape control illegal command logic (-CS:A2). With the designation A/B, it is also sent through terminal 3B88R to the tape control status logic (-ST:B4).

WRITE LOGIC

The write logic amplifies the write data outputs and write pulses from the tape control and applies them to the write buffer in the transport. The write logic also generates the lateral parity bit and the WRITE RESET level.

Write Data Outputs

During WRT and WRF commands, the six write data outputs WD0-WD5 apply output data characters from the tape control to the interface (-521:B1-4). Ground levels correspond to 1 bits. The WD levels determine which of the data flip-flops in the transport write buffer are to be complemented at the write pulse. Both polarities of the six WD outputs are applied to the write parity network. The parity network then generates a seventh level which determines whether or not the write buffer parity flip-flop is to be complemented at the write pulse.

Write Buffer

The write buffer is a 7-bit flip-flop register in the transport logic. Both outputs of each flip-flop are applied to the transport write amplifiers. Depending on the state of the associated write buffer flip-flop, each write amplifier drives one of two oppositely wound write head coils. Each transition of a write buffer flip-flop terminates the current through one coil and starts it in the other, changing the state of tape magnetization and writing a 1 in the corresponding tape channel.

Write Parity Net

As each 6-bit output character is transferred out to the interface by the WD outputs, the write parity net (521-:B and C1-5) generates an accompanying parity bit. If CB21 (in the tape control command buffer) contains 0, the writing is to be done in BCD (even) parity. The 0 state of CB21 causes a negative PAR 0 input to be applied to the write parity net (-521:C5). This input causes the parity net to generate a 1 parity output (a negative level) if an odd number of the six WD outputs are 1s. The negative parity 1 level is amplified by a 1684 Bus Driver and applied to the transport write buffer via interface output terminal P4-13. The transport then writes a 1 bit in the parity channel for the current character, yielding a character with an even

number of 1 bits. If CB21 contains a 1, the writing is to be done in binary (odd) parity. A ground PAR 0 input is then applied to the parity net, and the 1 parity output is generated only if an even number of the six WD outputs are 1s.

The write parity net is composed of a 3-stage, 7-input exclusive OR circuit. All of the WD inputs are negative assertion levels. The 1 parity output level is asserted negative only if an odd number of the seven inputs are 1 (counting the six WD bits and the contents of CB21). The assertion levels shown in the two upper stages of the net assume that each of the three pairs of WD bits includes a 1 bit and a 0 bit and assumes that the PAR 0 input is negative (i.e. that CB21 is 0, indicating BCD parity). In inverter 3B17, three external 3.3K resistors have been substituted for the normal 1.5K load resistors. This reduces the current through the lower legs of the parity net and avoids excessive loading.

The WEF command level is inverted in the tape control (Figure 3-1C4) and applied to the write parity net as $\overline{\text{WRITE EOF}}$ (-521:B2). When this level is negative (as it is for write file commands), diode 3B12 holds the parity net output at ground, inhibiting the generation of a 1 parity bit. The end-of-file character 001 111 (17 octal) should always be written in BCD parity (with a 0 parity bit). The $\overline{\text{WRITE EOF}}$ input ensures that the write parity net produces the correct 0 parity bit for the EOF character even if the programmer should inadvertently give a WRF command with binary parity.

Write Reset Flip-Flop

The write reset flip-flop (-521:C3) governs the state of the WRITE RESET output to the transports. Until the write reset flip-flop is set to 1 (grounding WRITE RESET), the transport write buffer flip-flops cannot be complemented. When the write reset flip-flop is reset to 0, the resulting drop in the WRITE RESET level clears the write buffer.

The write reset flip-flop is set to 1 at the first WP write pulse of the record, causing the WRITE RESET level to rise to ground and permitting the write buffer flip-flops to be complemented. All WP pulses are delayed 1 μsec at pulse amplifier 3B25 (-521:C2) before being applied to the transport as WP1 pulses. This gives the WRITE RESET level time to rise to ground before the first WP pulse is applied to the transport write buffer.

The write reset flip-flop is reset at each START 1 pulse. The START 1 pulse originates in the 516 Tape Control as EOR 2. A START 1 is produced at the beginning of each command to ensure that every record begins with the write reset flip-flop and the write buffer cleared. Another START 1 is applied to the interface four to five character spaces after the last WP pulse of each record (4 spaces during WRF records, and 4.5 spaces + 2.4 μsec during WRT records). The resulting clear of the write reset flip-flop causes the WRITE RESET output to drop to a negative level and clears the transport write buffer. The clear complements the write buffer flip-flops that are left in the 1 state after the final data character of the record; i.e. those flip-flops that have already been complemented an odd number of times in the course of the record. This yields an even longitudinal parity for each of the seven channels of the record and permits the EOR mark to be used as a longitudinal parity check character. (Only the six data channels are actually checked for longitudinal parity.)

READ LOGIC

The read logic accepts seven transport signals corresponding to each input data character, assembles the input character in the read buffer, tests it for the correct lateral parity, and applies it to the data control (via the tape control).

Read Buffer

The read buffer (-521:A5-8) is composed of seven 500-kc flip-flops; six data flip-flops RB0-5 and a lateral parity flip-flop RB6. Each of the RB flip-flops is set to 1 at the peak of every transport input signal on the corresponding read channel. The ground assertion 1 outputs of the six RB data flip-flops are applied to output buffer inverters within the interface, and the inverter outputs are, in turn, applied to the tape control as RBB0(1)-RBB5(1). Negative levels represent 1s in the corresponding bit positions. The negative assertion 1 output of the RB parity flip-flop RB6(1) is applied to the tape control directly without inversion.

RB Clear

The read buffer is cleared by the 0⟶RB pulse from the tape control. This clear occurs at the beginning of each command to ensure that the buffer starts reading in the correct initial condition. It also occurs after each character is read to prepare the read buffer for the 1s

transfer of the next character. The latter clear occurs at CSD(0), the end of the character skew delay. At that time, the current character has been fully entered into the read buffer. The contents of the buffer are sampled at the same time that the buffer is cleared.

RB Set

At successive changes of tape magnetization, the differential amplifier at each of the transport read heads produces output signals of alternating polarity (the polarity of each signal depending on the sign of the change). The bipolar output signal from each of the seven tape channels is applied to one channel of the 1534 Variable Slicing Rectifier circuits in the interface (-521:B4-8). The 1534 channel rectifies the output of the differential amplifier and produces a positive output pulse at each input of either polarity. However, no output is generated unless the input exceeds a preset slice level. (This is to prevent noise inputs from producing spurious output pulses.)

Slice Control - Three separate slice levels are used:

1. The slice level is set at 25 percent of the signal amplitude for the read after write (when the tape is most strongly magnetized).

2. A 15 percent slice level is used for normal read operations.

3. A 5 percent slice level is used for old tapes with weak magnetization.

The slice level is controlled by the X and Y inputs to the 1534 circuits. When T WRITE/READ is negative (indicating a write, write file, or write blank tape command), both the X and the Y inputs to the 1534 circuits are grounded, producing the 25 percent slice level. When T WRITE/READ is ground (indicating that one of the read commands is being used), the slice level is determined by the state of CB20 (in the tape control command buffer). If CB20 contains 0, the SLICE 1 input from the tape control is ground. This condition causes a ground level to be applied to the X inputs of the 1534 circuits, and a negative level to be applied to the Y inputs. As a result, the 15 percent normal read slice level is used. However, if CB20 contains 1, the SLICE 1 input is negative, and the X and Y inputs to the 1534 circuits are reversed; X is then negative, and Y is ground. This causes the 1534 circuits to use the 5 percent low read slice level which is appropriate for weakly magnetized tapes.

Read Enable - The outputs from the seven 1534 Slicing Rectifier Channels are applied to the seven channels of the 1535 Peak Detector Circuits shown in (-521:B5-8). These circuits are enabled only when both the GO flip-flop and the RDE flip-flop (-521:B4) are in the 1 state. The GO flip-flop is set at the IOT 7104 start pulse; the RDE flip-flop is set at the first subsequent MP pulse. This MP pulse originates in the tape control as the MDF pulse marking the end of the UPS acceleration state. Both the GO flip-flop and the RDE flip-flop are cleared at the 0⟶CM pulse. This read enable gating ensures that the peak detectors are enabled only when the tape is moving at full speed.

Peak Detector Outputs - When enabled and pulsed by an output from the corresponding slicing rectifier channel, each peak detector generates a negative output pulse. The pulse is inverted, and the resulting positive pulse is applied to the complement input of the corresponding RB flip-flop (setting it to the 1 state because it was cleared at the preceding 0⟶RB pulse).

RBCP and RBEF

The RBCP level (read buffer character present) is asserted whenever any of the six read buffer data bits RB0-5 contain 1 (-521:C7). This level is sent to the tape control where it is ORed with RB6(1) to produce the CHP (character present) signal.

The RBEF read buffer end-of-file level (-521:C6) is asserted and applied to the tape control when the six data bits of the read buffer RB contain the file mark character (17 octal).

End-of-File Flag

The EFF flip-flop (-521:C8) is set to 1 to indicate that a file mark has been sensed. Although the EFF flip-flop is physically located in the interface, the signals controlling its set and reset originate in the tape control. The interface sends an RBEF level to the tape control whenever a file mark character (17 octal) is read. If BR6(0) is also asserted, indicating BCD parity, and if the character is the first data character of the current record, the EOF level is asserted, and the character is treated as a file mark.

At the RCP pulse (which coincides with the end of the CSD delay), the tape control applies a 1⟶EFF ground pulse to interface input terminal 3B81M, setting EFF (-521:C8). If additional

data characters are in the current record, the file mark indication is assumed to be the result of a programming error, and EFF is reset at the EOR 1 pulse that ends the record. The tape control accomplishes this by applying a 0⟶EFF ground pulse to interface input terminal 3B88K. The EFF is also reset at the beginning of each command by STATUS CLEAR (designated MCL within the 516 Tape Control).

Read Parity Net

As each input character is read into the read buffer RB0-6, the read parity net (-521:C and D4-8) checks the character for correct lateral parity. If the parity is not correct, a negative TRPE (tape read parity error) level is sent to the tape control.

The read parity circuit is virtually identical to the write parity net described previously under Write Parity Net, except that it contains one extra leg. Both the PAR 0 and PAR 1 outputs of the CB21 flip-flop are used as inputs to the read parity net. When binary (odd) parity is specified, the PAR 1 level is negatively asserted; and when BCD parity is specified, PAR 0 is negatively asserted. The RB inputs are also negative assertion levels. A parity error TRPE is asserted when PAR 0 is asserted and an odd number of the seven RB bits RB0-6 contain 1s; TRPE is also asserted when PAR 1 is asserted and an even number of the RB bits contain 1s.

Parity Error Flip-Flop

The parity error flip-flop PER (-521:C8) is set to 1 to indicate that a lateral parity error has occurred. The PER flip-flop is physically located in the interface, but the signals controlling its set and reset originate in the tape control. The interface sends a TRPE level to the tape control whenever the character currently being read displays an apparent parity error. If the character is a data character rather than an EOR mark, the tape control sets the parity error flip-flop by applying a ground pulse to interface input terminal 3B81N. However, if the character displaying the apparent parity error is an EOR mark, MCD(1) is not asserted at RCP time, and the parity error flip-flop is not set. (The EOR mark serves as the longitudinal parity check character and may or may not have correct lateral parity. For this reason, it is necessary to prevent any TRPE signals that are produced by EOR marks from setting the PER flip-flop.)

If the EFF is set to 1, it causes the PER flip-flop to be reset to 0 at EOR 1. This removes the false parity error that would appear if the file mark were read in binary parity through a programming error. The PER is also reset at the beginning of each command by STATUS CLEAR, (designated MCL within the 516 Tape Control).

CHAPTER 9

TAPE CONTROL INTERFACE 522A

The 522A Interface permits the Magnetic Tape Control 516 or the Magnetic Tape Control 75A to control up to eight Type 729 or Type 7330 Tape Transports. The interface contains a read buffer, write and read parity networks, and associated logic. The end-of-file flag and lateral-error-parity flag are also located in the interface, as are various control circuits that process command and selection levels from the tape control and apply appropriate control signals to the tape transports.

The system logic is shown in four engineering logic drawings, BS-D-522A-0-3(1), (2), (3), and (4). A flow diagram of the interface start sequence and turn around sequence is presented in Figure 9-1.

COMMAND AND SELECTION INPUTS

The command and selection inputs select a specific tape transport for the current command and cause the interface logic to apply appropriate control signals to that transport.

Selection Inputs

Eight unit selection levels UNIT 0-7 are applied to the interface from the tape control (-522A(1):B-D1). The selection level corresponding to the selected tape transport is negatively asserted; the other seven levels remain at ground. All eight levels are sent through 4659 DEC-to-IBM line drivers before being applied to the transports as T UNIT 0-7. The tape control always selects the transport to be used in the current command (by asserting the associated unit selection level) before applying command levels to the interface.

Command Inputs

Four command levels are applied to the interface from the tape control: FOW, BAC, REWIND, and WRITE (-522A(2):B1). These inputs, together with the CON(1) level, (from CB24 in the tape control command buffer (-522A(1):B6), determine the type of operation the transport is to

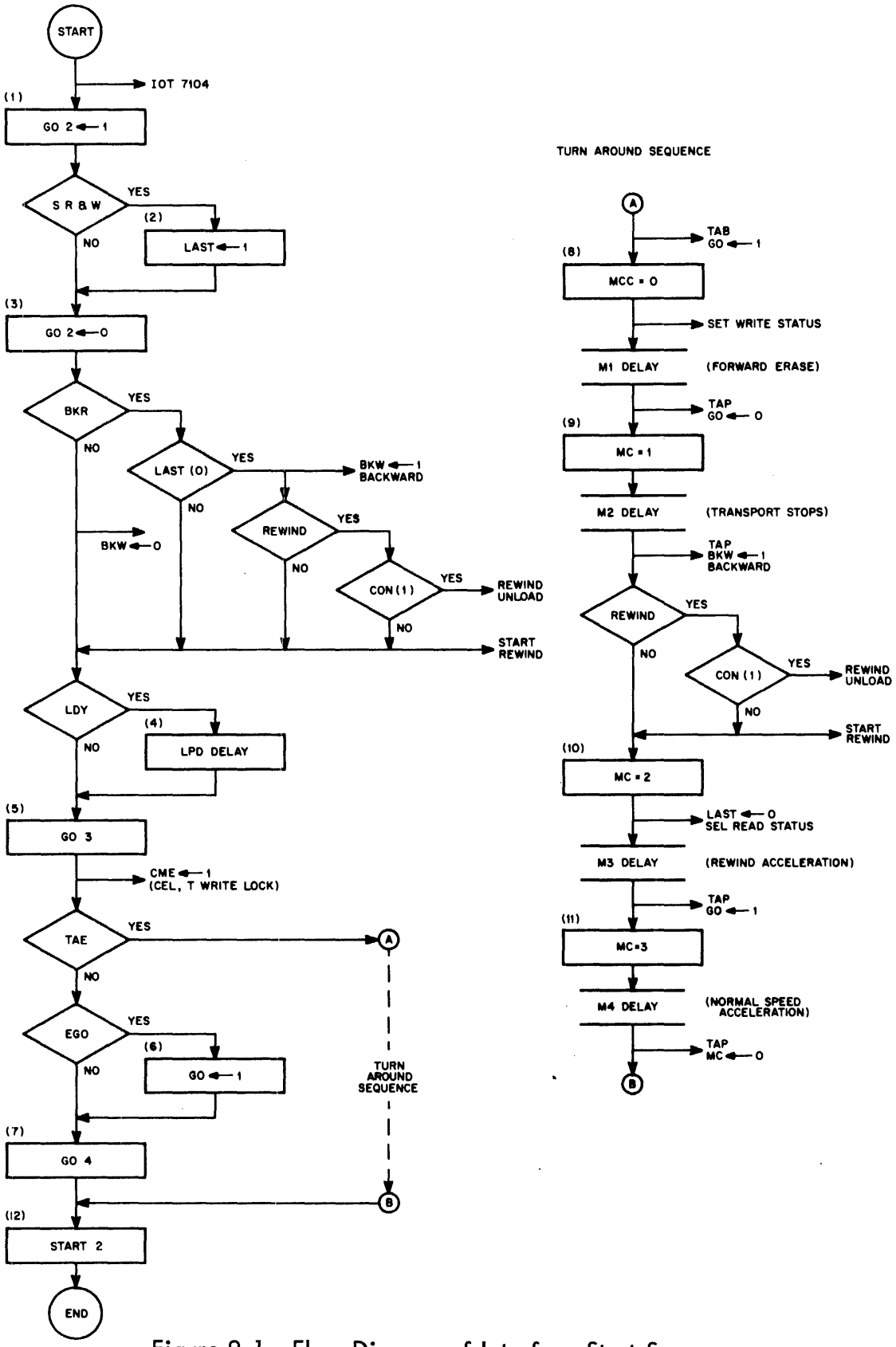


Figure 9-1 Flow Diagram of Interface Start Sequence
(Including Turn Around Sequence When Used)

commence after receiving a start pulse from the tape control and executing the necessary start sequence. The WRITE level is asserted at ground; the other three command levels and the CON(1) level are negatively asserted.

The WRITE level is used to generate T WRITE. The T WRITE, in turn, causes the SET WRITE STATUS signal to be applied to the transports, provides a necessary input condition for generating T WRITE LOCK, and is used to control the read logic slice level. When WRITE is asserted at ground, SET WRITE STATUS sets the selected transport to write status by enabling both the write logic and the read logic. A negative WRITE input does not enable the transport write logic, but rather allows the selected transport to be held in read status, or (through a proper turn around sequence; see Turn Around Sequence below) to be switched from write status to read status.

The FOW command input indicates that the current command calls for forward tape motion and the BAC input calls for backward tape motion (at normal reading speed). The REWIND command input causes the transport to reverse at rewind speed (about twice normal reading speed). If CB24 (in the tape control command buffer) contains 0, the transport rewinds only as far as the tape load point and then stops. However, if CB24 contains 1, CON(1) is asserted, and a rewind unload command is specified; this command rewinds the entire tape.

SIGNALS BETWEEN INTERFACE AND TRANSPORTS

Various control signals are sent from the interface to the selected transport, and a number of status signals are returned from the transport to the interface. The present paragraph groups these signals by function and summarizes their principal uses. More detailed descriptions of the control and status signals are included in the signal glossaries in Chapter 12.

Converters

The Tape Transports Type 729 and Type 7330 use and furnish IBM 7090 N and P signals. Consequently, the N and P input signals from the transports are converted to DEC logic levels before being used in the interface logic, and the DEC logic levels from the interface are converted to N and P signals before being applied to the transports. The 7090-to-DEC conversion is

accomplished by DEC 4505 and 4506 modules; for DEC-to-7090 conversion DEC 4659 and 4660 modules are utilized. Table 9-1 summarizes the inputs and outputs of these four converter modules. (For more detailed specifications see DEC System Module Catalog, C-100).

TABLE 9-1 CONVERTER MODULE INPUTS AND OUTPUTS

Module Type	Input Signal	Resulting Output
	7090 to DEC	
4505	+ P	0 vdc
	- P	- 3 vdc
4506	- N	0 vdc
	+ N	- 3 vdc
	DEC to 7090	
4659	0 vdc	- N
	- 3 vdc	+ N
4660	0 vdc	- P
	- 3 vdc	+ P

Motion Control Signals

The interface controls tape motion by applying four motion control signals to the selected transport: T GO, BACKWARD, START REWIND, and REWIND UNLOAD.

For all nonrewind commands, the T GO signal (-522A(2):C8) actually starts tape motion at the selected transport. If the BACKWARD signal (-522A(1):A8) is not present, T GO starts forward tape motion. If BACKWARD is asserted, T GO starts backward tape motion. In either case, when T GO ends, the transport stop sequence begins.

Neither of the two rewind signals is ever asserted unless the BACKWARD signal is also asserted. The BACKWARD and START REWIND signals (-522A(1):A8) cause the selected transport to begin reverse tape motion at rewind speed. The BACKWARD and REWIND UNLOAD signals

(-522A(1):B8) cause the transport to continue the rewind command beyond load point to rewind the entire tape. The BACKWARD signal with either the START REWIND or the REWIND UNLOAD is sufficient to start tape motion; the T GO signal is not required during rewind commands.

The same motion control signals are employed during turn around sequences. To produce the forward erase phase of the turn around, T GO is asserted and ends, and forward tape motion stops. If the turn around sequence calls for a rewind, reverse tape motion begins as soon as BACKWARD and either START REWIND or REWIND UNLOAD is asserted. If the turn around calls for a normal speed backward command, BACKWARD is asserted, and T GO is again asserted, starting backward tape motion.

Other Transport Control Signals

Besides the motion control signals described above, other control signals are applied to the transports; these signals are described briefly below.

SET WRITE STATUS (-522A(2):B8)

The SET WRITE STATUS signal is asserted during the forward erase phase of the turn around sequence, and during all write, write file, and write blank tape commands. It enables the write circuits of the selected tape transport, and sets the transport to write status.

SEL READ STATUS (-522A(1):B8)

If the selected transport is in write status (for the previous command), and if it is necessary to execute a turn around sequence for the current command, the SEL READ STATUS signal switches the transport from write to read status at the proper stage of the turn around sequence. If the transport was initially in read status and is executing another command not requiring write status, the SEL READ STATUS signal merely holds the transport in read status.

SET HIGH (LOW) DENSITY (-522A(1):B8)

The SET HIGH (LOW) DENSITY signals light the high-(low) density indicator lamps at the 729 Transports. High density is shown when D200 is not asserted; low density is shown when D200 is asserted.

TURN OFF TAPE INDICATE (-552A(1):AB)

This signal turns off the TAPE INDICATE light at the transport and ends the TAPE INDICATE signal returned from the transport to the interface. The TAPE INDICATE is turned on only if a write command passes tape end point. A backward or rewind command following such a write command produces TURN OFF TAPE INDICATE.

WRITE CHARACTER (-522A(2):B8)

Until the write reset flip-flop is set, the WRITE CHARACTER signal prevents any complementing of the transport write buffer. At the end of the record, the write reset flip-flop is reset, and the resulting change in the WRITE CHARACTER signal clears the transport write buffer and writes the EOR mark on tape.

Status Signals from Transports

The selected tape transport sends the interface status signals to indicate its current condition. These signals are described briefly below.

SELECT & READY (-522A(1):C, D8)

The SELECT & READY signal indicates that the selected transport is not rewinding and that it is ready to accept the next command. SELECT & READY is called T READY within the interface and RDY in the 516 Control.

SELECT READY & WRITE (-522A(2):C8)

The SELECT READY & WRITE signal indicates that the transport is set to write status. This means that it has received a write command, and, having met all necessary conditions for writing, has set the read-write status indicator to write status.

T WRITE LOCK (-522A(1):D6)

The T WRITE LOCK signal indicates that although a write command has been given, the selected transport has remained in read status. (The most common cause of this occurrence is failure to remove the tape write enable lockout ring from the tape reel.) In the 516 Control, T WRITE LOCK is designated TWL.

TAPE INDICATE (-522A(1):C8)

This signal indicates that the tape has reached or passed the end point while writing. TAPE INDICATE is called T END POINT within the interface, and TEP in the 516 Control.

REWIND STATUS (-522A(1):C8)

The REWIND STATUS signal indicates that the selected transport is rewinding. This signal is asserted until the transport reaches end point. In the interface, it is called T REWIND; in the 516 Control, it is designated RWD.

START SEQUENCE

The interface start sequence incorporates a sequential chain of programmed delays designed to provide correct timing for the control signals used with each set of command conditions. A flow diagram of the interface start sequence is presented in Figure 9-1.

Set LAST

In all cases, the IOT 7104 interface start pulse begins the start sequence by setting the GO 2 delay (-522A(1):A2). The SELECT READY & WRITE signal (-522A(2):C8) is sampled during the time that the GO 2 delay remains set. (When the tape is at load point, the sample period is extended for the duration of the T LOAD POINT signal (-522A(2):D7). If SELECT READY & WRITE is asserted during the sample period, LAST(1) sets the LAST flip-flop; see Figure 9-1, steps 1 and 2. The STATUS LAST OPERATION WRITE level (-522A(2):A1) sends the state of LAST to the tape control status register.

The set of the LAST flip-flop indicates that the selected transport is in write status. If this is the case, certain constraints are imposed on the start sequence. Unless the next command is a write, write file, or write blank tape command, a turn around sequence (see Turn Around Sequence p. 9-10) must be executed to free the transport from write status. The turn around sequence requires either a backward command or a rewind command to switch the transport from write status to read status.

For example, a write command could not be followed immediately by a forward read command. (If this were attempted, the transport would not start.) It is, however, permissible to follow the write command with a backward read, and then to program a forward read. The backward read produces a turn around sequence that switches the transport to read status. When the transport enters read status, the LAST flip-flop is reset to 0, and the transport is free to execute a forward read as its next command.

Set or Reset BKW

At the expiration of the GO 2 delay, the BKW flip-flop may be either set or reset (Figure 9-1, step 3). If the current command is neither a backward command nor a rewind command, the BKR flip-flop is reset (-522A(1):A2). However, if the current command is a backward or rewind command, BKR is asserted.

The BKR level may be asserted with either of two conditions: LAST (1), or LAST (0). If LAST (1) is asserted, the BKR level calls for a turn around sequence, and the BKW flip-flop is left for the time being in the 0 state. (Assertion of LAST (1) indicates that the preceding command must have been a write, write file, or write blank tape. Because these are all forward commands, BKW must have been in the 0 state at the beginning of the current command.) However, if LAST (0) is asserted, no turn around sequence is required, and at the termination of the GO 2 delay, the BKW flip-flop is immediately set to 1. The BACKWARD signal is applied to the selected transport as long as the BKW flip-flop remains set.

The effect of setting BKW to 1 depends upon two more decision points (designated REWIND and CON (1) in Figure 9-1, step 3). If the REWIND command level is not present, a normal speed backward command is called for, so tape motion does not start when BKW is set and BACKWARD is applied to the transport. (For all normal speed commands, the GO flip-flop must be set and T GO must be applied to the transport to start tape motion. The GO flip-flop is not set until the GO 3 pulse occurs later (step 5) in the start sequence.) However, if the REWIND command level is present, tape motion begins as soon as BKW is set. If CON (1) is asserted, the BACKWARD and REWIND UNLOAD signals are applied to the transport when BKW is set. If CON (0) is asserted, BACKWARD and START REWIND are applied to the transport when BKW is set. In either case, tape motion begins when BKW is set at the expiration of the GO 2 delay.

Load Point Delay

Depending upon whether or not the LDY level is asserted, the LPD delay (-522A(1):C3) may or may not be included between the termination of the GO 2 delay and the generation of the GO 3 pulse (Figure 9-1, step 4). If LDY is not asserted, the LPD is not set, and the GO 3 pulse is produced as soon as the GO 2 delay times out. However, if LDY is asserted, the GO 2 reset sets LPD, and the GO 3 pulse is not generated until LPD times out.

The LPD provides the delay needed for the tape to move off the load point and for the transport to enter the ready state. For the Type 7339 Transport, the LPD delay is also used for another function. The Type 7330 requires an appreciable time to change direction; relays must switch, the motor fields must change polarity, and the capstan must reverse direction. The LPD provides this change of direction delay for systems using the 7330 Transport.

Command Enable Last Level

The GO 3 pulse (Figure 9-1, step 5) sets the CME flip-flop (-522A(1):B4). If LAST (0) is asserted, indicating that the selected transport is in read status, the CME set causes the command enable last level CEL to be asserted. The CEL level is a sufficient condition for applying the SEL READ STATUS signal (-522A(1):B8) to the transports. This signal holds the selected transport in read status.

If the current command is a write, write file, or write blank tape command, T WRITE is asserted, and the CEL level causes the T WRITE LOCK signal (-522A(1):D6) to be applied to the tape control. The T WRITE LOCK signal indicates that although a write command has been given, the selected transport has remained in read status. The most common cause of this occurrence is failure to remove the tape write enable lockout ring from the tape reel. In the 516 Control, the T WRITE LOCK signal is designated TWL.

Normal Start

If no turn around sequence is required, TAE is not asserted, and the interface executes the normal start operations shown in steps 6 and 7 of Figure 9-1. (When a turn around sequence is indicated, TAE is asserted, and the interface executes steps 8 through 11 instead of steps 6 and 7.)

Provided that EGO is asserted, the GO flip-flop is set to 1 at GO 3 (see step 6). The set of GO causes the T GO signal to be applied to the transport, beginning normal speed tape motion. If the BACKWARD signal is asserted, the tape starts to move backward when T GO appears; if BACKWARD is not asserted, the tape starts to move forward at T GO; refer to Set or Reset BKW above.

There are conditions for which EGO is not asserted, and the GO flip-flop is not set. The GO flip-flop is not set for a NOP command. It is not set for a write command given beyond end point or for a write command during which the transport has remained in read status. The GO flip-flop is not set when a write command is followed by a forward read or space command (refer to Set LAST above). Finally, the GO flip-flop is not set for a rewind command given with the transport in read status (no turn around required); for such a command, the REWIND UNLOAD or START REWIND signal described in Set or Reset BKW starts tape motion, and there is no need to set GO.

One μsec after the beginning of GO 3, the trailing edge of that pulse initiates the GO 4 pulse (Figure 9-1, step 7). Provided that no turn around sequence is required, TAE is negated (-522A (1):C5) and the GO 4 pulse triggers the tape control start pulse START 2 (Figure 9-1, step 12). In the 516 Control, START 2 is designated STM.

Turn Around Sequence

To switch the selected transport from write status to read status, a turn around sequence is required. The turn around first causes the transport to forward erase approximately 3/4 inch of tape. The transport then stops forward tape motion, switches from write status to read status, and begins reverse tape motion at rewind or normal speed (the choice of speed depending on the nature of the turn around command).

A turn around sequence is initiated by giving a rewind or backward command when the selected transport is in write status. A rewind or backward command given while LAST(1) is present causes the assertion of the turn around enable level TAE (-522A(1):B3) and initiates the turn around sequence at GO 3. Instead of following the normal start sequence described under Normal Start, the interface follows the turn around sequence, jumping directly under step 5 of Figure 9-1 to step 8, and proceeding through steps 9-12.

When TAE is asserted, the GO 3 pulse triggers the turn around begin pulse TAB (-522A(2):C5) and starts M1, the first delay of the turn around sequence. Because the EGO level is asserted during turn around, the GO 3 pulse also sets the GO flip-flop (-522A(2):A6), starting forward tape motion for the forward erase phase of the turn around sequence. The first TAD set begins the MCC = 0 level (Figure 9-1, step 8); that level, in turn, applies the SET WRITE STATUS signal (-522A(2):B8) to the selected transport, enabling the transport write circuits during the forward erase phase of the turn around. The forward erase continues until M1, the first TAD delay, times out. The resulting TAP pulse advances the MC counter to MC = 1 (Figure 9-1, step 9) and resets the GO flip-flop (-522A(2):A6). The reset of GO begins to stop the forward motion of the tape. The timeout of the first TAD delay starts the second TAD delay, M2.

During the M2 delay, the tape's forward motion comes to a full stop. The TAP pulse that terminates the M2 delay sets the BKW flip-flop (-522A(1):A3) and applies the BACKWARD signal to the transport. If the current command is a normal speed backward command, no tape motion results. However, if the current command is a rewind unload or rewind, a REWIND UNLOAD or START REWIND signal is applied to the selected transport with the BACKWARD signal, and tape motion begins immediately. The same TAP pulse that sets BKW also advances the MC counter to MC = 2 (Figure 9-1, step 10), resetting the LAST flip-flop (-522A(2):A7) and applying the SEL READ STATUS signal to the selected transport (-522A(1):B8). The SEL READ STATUS signal switches the transport from write status to read status. The timeout of the second TAD delay starts the third TAD delay, M3.

If the current command is a rewind unload or a rewind command, the tape accelerates during the M3 delay. If the current command is a normal speed backward command, acceleration does not begin until the end of the M3 delay. The TAP pulse that terminates the M3 delay and steps the MC counter to MC = 3 (Figure 9-1, step 11), also sets the GO flip-flop to the 1 state. The set of GO applies the T GO signal to the selected transport and starts tape motion (if the current command is a normal speed backward command rather than a rewind command). The timeout of the third TAD delay starts the fourth and final TAD delay, M4.

During the M4 delay, if the current command is a normal speed backward command, the tape reverses across the 3/4 inch section of tape that was forward erased during M1. The TAP pulse

that terminates the M4 delay resets the MC counter to MC = 0. This same TAP (-522A(1):B4) terminates the turn around sequence by generating the tape control start pulse START 2 (Figure 9-1, step 12). In the 516 Control, START 2 is designated STM.

WRITE LOGIC

The write logic receives the write data outputs and write pulses from the tape control, converts them to -N signals, and applies them to the write buffer in the transport. The write logic also generates the lateral parity bit, the WRITE CHARACTER signal, and the WRT NOK alarm condition.

Write Data Outputs

During WRT and WRF commands, the six write data outputs WD0-WD5 apply output data characters from the tape control to the interface (-522A(1):B2-5). Ground levels correspond to 1 bits. The WD levels determine which of the data flip-flops in the transport write buffer are to be complemented at the write pulse. The WD outputs are inverted, and both the original WDn(1) levels and their BWDn(0) complements are applied to the write parity network. (At the parity net inputs (-522A(1)C3-6), these signals are shown as WDn(0) and BWDn(1); i.e. at negative assertion rather than ground assertion.) The parity network then generates a seventh output which determines whether or not the write buffer parity flip-flop is to be complemented at the write pulse.

The six write data outputs WD(1)-WD5(1) and the output of the write parity network are applied to the transport write buffer through seven 4659 Line Drivers. At the outputs of the line drivers, these seven signals are redesignated WRITE CH B, A, 8, 4, 2, 1, and C, respectively. The WD (or parity) outputs corresponding to 1 bits produce WRITE CH signals at -N; the 0 bits produce +N signals. The seven WRITE CH signals control the response of the transport write buffer to the write pulses. Those write buffer flip-flops which correspond to 1 outputs (and hence -N signals) are complemented at the write pulse; the remaining write buffer flip-flops are not complemented.

Write Buffer

The write buffer is a seven-bit flip-flop register in the transport logic. Both outputs of each flip-flop are applied to the transport write amplifiers. Depending on the state of the associated write buffer flip-flop, each write amplifier drives one or the other of two oppositely wound write head coils. Each transition of a write buffer flip-flop terminates the current through one coil and starts it in the other, changing the state of tape magnetization and writing a 1 in the corresponding tape channel.

Write Parity Net

As each 6-bit output character is transferred out to the interface by the WD outputs, the write parity net (-522A(1):B, C3-7) generates an accompanying parity bit.

If CB21 (in the tape control command buffer) contains 0, the writing is to be done in BCD (even) parity. The 0 state of CB21 causes a negative PAR 0 input to be applied to the write parity net (-522A(1):C7). This input causes the parity net to generate a 1 parity output (a ground level) if an odd number of the six WD outputs are 1s.

The ground parity 1 level is applied to line driver 3A19, and the WRITE CH C output signal from that circuit is, in turn, applied (at - N) to the transport write buffer parity flip-flop. The parity flip-flop is complemented by the write pulse, and the transport writes a 1 bit in the parity channel for the current character. This yields a character with an even number of 1 bits.

If CB21 contains a 1, the writing is to be done in binary (odd) parity. A ground PAR 0 input is applied to the parity net, and the 1 parity output is generated only if an even number of the six WD outputs are 1s.

The write parity net is composed of a 3-stage, 7-input exclusive OR circuit. All of the WD and BWD inputs are negative assertion levels (-522A(1):C3-6). The 1 parity output level is asserted at ground (-(1):A6) only if an odd number of the seven inputs are 1 (counting the six WD bits and the contents of CB21). The assertion levels shown in the two upper stages of the net assume that each of the three pairs of WD bits includes a 1 bit and a 0 bit and that the PAR 0 input is negative (i.e. that CB21 is 0, indicating BCD parity). In inverter 3B17,

three external 3.3K resistors have been substituted for the normal 1.5K load resistors. This reduces the current through the lower legs of the parity net and avoids excessive loading.

The WEF command level is inverted in the tape control (-CM:C4) and applied to the write parity net as $\overline{\text{WRITE EOF}}$; refer to -522A(1):B5. When this level is negative (as it is for write file commands), inverter 3B22 holds the parity net output negative, inhibiting the generation of a 1 parity bit. The end-of-file character 001 111 (17 octal) should always be written in BCD parity (with a 0 parity bit). The WRITE EOF input ensures that the write parity net produces the correct 0 parity bit for the EOF character even if the programmer should inadvertently give a WRF command with binary parity.

Write Reset Flip-Flop

The state of the write reset flip-flop (-(2):A3) controls the WRITE CHARACTER signal applied to the transports (-(2):B8). Although the write circuits of the selected transport are enabled when SET WRITE STATUS is asserted, the transport write buffer flip-flops cannot be complemented until the write reset flip-flop is set to 1.

The write reset flip-flop is set to 1 at the first WP pulse of the record. (The WP pulses are delayed 1 μsec before being applied to the transports; this gives the WRITE CHARACTER signal time to change state so that even the first WP of the record is enabled to complement the write buffer bits of the selected transport.)

The write reset flip-flop is reset at each START 1 pulse. The START 1 pulse originates in the 516 Tape Control as EOR 2. A START 1 is produced at the beginning of each command to ensure that every record begins with the write reset flip-flop and the write buffer cleared. Another START 1 is applied to the interface four to five character spaces after the last WP pulse of each record (4 spaces during WRF records, and 4.5 spaces +2.5 μsec during WRT records). The resulting clear of the write reset flip-flop causes the WRITE CHARACTER signal to change state and clears the transport write buffer.

The START 1 clear complements the write buffer flip-flops that are left in the 1 state after the final data character of the record; i.e. those flip-flops that have already been complemented an odd number of times in the course of the record. This yields an even longitudinal parity for

each of the seven channels of the record and permits the EOR mark to be used as a longitudinal parity check character. (Only the six data channels are actually checked for longitudinal parity.)

The Type 729 and Type 7330 Transports require WRITE CHARACTER signals of opposite polarity; consequently, different outputs of the write reset flip-flop are jumpered to the input of line driver 3B21 (-522A(2):B7), the choice depending on the type of transport used. However, for either transport type, the write reset flip-flop is set to 1 so that the write buffer can be complemented, and the write reset flip-flop is reset to 0 to produce the EOR mark.

Write Pulse Generation

The WP write pulses originate in the tape control. They are delayed 1 μsec by pulse amplifier 3A22 (-522A(2):A6), are stretched to the correct pulse width by 4301 Delay 3A3, and are applied to the transports via line driver 3A5 (as - N pulses) with the designation WRITE PULSE. The width of the WRITE PULSE may vary from 2 to 12 μsec depending on the type of transport used.

Write Echo

The write echo flip-flop WOK and the write not OK flip-flop WRT NOK make up a 2-stage alarm network (-(2):A4). During write and write file commands, this network ensures that the programmer has flag notice of any total failure to write a character. The WOK flip-flop is set at each WP pulse. Provided that a 1 is written in (and read back from) at least one of the seven tape channels, a WRITE ECHO signal (-2):B8) is returned from the selected tape transport, resetting the WOK flip-flop.

Normally, each WP pulse writes a character that produces a WRITE ECHO signal return, and the WOK flip-flop is reset by the resulting WOK(0) ground level (-(2):B7); in this case, the WOK flip-flop is reset prior to the arrival of the next WP pulse from the tape control. However, if any WP pulse should fail to produce a WRITE ECHO return, it is likely that a write failure has occurred, and WOK is not reset before the next WP pulse arrives from the tape control. Because the WOK flip-flop remains in the 1 state, the next WP pulse strobes WOK(1) into the WRT NOK flag.

The WRT NOK flag is set whenever there is a total failure to write any character. It remains set throughout the command during which the failure occurs; it is not reset until the STATUS CLEAR at the beginning of the following command. Whenever WRT NOK contains 1, the negative assertion level TO STATUS REGISTER WRITE NOK (-(2):C1) is applied to the tape control status register. (In the 516 Control, this level is designated TFR - AB - WRK NOK.)

READ LOGIC

The read logic accepts seven transport signals corresponding to each input character, assembles the input character in the read buffer, tests it for the correct lateral parity, and applies it to the data control (via the tape control).

Read Buffer

The read buffer (-(3):A4-7) is composed of seven 500-kc flip-flops; six data flip-flops RB0-5 and a lateral parity flip-flop RB6. Each of the RB flip-flops is set to 1 at the peak of every transport input signal on the corresponding read channel. The ground assertion 1 outputs of the six RB data flip-flops are applied to output buffer inverters within the interface, and the inverter outputs are, in turn, applied to the tape control as RBB0(1)-RBB5(1). Negative levels represent 1s in the corresponding bit positions. The negative assertion 1 output of the RB parity flip-flop RB6(1) is applied to the tape control directly without inversion.

RB Clear

The read buffer is cleared by the 0 $\longrightarrow$ RB pulse from the tape control. This clear occurs at the beginning of each command to ensure that the buffer starts reading in the correct initial condition. It also occurs after each character is read to prepare the read buffer for the 1s transfer of the next character. The latter clear occurs at CSD(0), the end of the character skew delay. At that time, the current character has been fully entered into the read buffer. The contents of the buffer are sampled at the same time that the buffer is cleared.

RB Set

At successive changes of tape magnetization, the differential amplifier at each of the transport read heads produces output signals of alternating polarity (the polarity of each signal depending

on the sign of the change). The bipolar output signal from each of the seven tape channels is applied to one channel of the 1534 Variable Slicing Rectifier circuits in the interface (-(3):A and C2). The 1534 channel rectifies the output of the differential amplifier and produces a positive output pulse at each input of either polarity. However, no output is generated unless the input exceeds a preset slice level. (This is to prevent noise inputs from producing spurious output pulses.)

Slice Control - Three separate slice levels are used:

1. The slice level is set at 25 percent of the signal amplitude for the read after write (when the tape is most strongly magnetized).

2. A 15 percent slice level is used for normal read operations.

3. A 5 percent slice level is used for old tapes with weak magnetization.

The slice level is controlled by the X and Y inputs to the 1534 circuits. When T WRITE (-(3):C3) is asserted negatively (indicating a write, write file, or write blank tape command), both the X and the Y inputs to the 1534 circuits are grounded, producing the 25 percent slice level. When T WRITE is ground (indicating that one of the read commands is being used), the slice level is determined by the state of CB20 (in the tape control command buffer). If CB20 contains 0, the SLICE 1 input from the tape control is ground. This causes a ground level to be applied to the X inputs of the 1534 circuits, and a negative level to be applied to the Y inputs. As a result, the 15 percent normal read slice level is used. However, if CB20 contains 1, the SLICE 1 input is negative, and the X and Y inputs to the 1534 circuits are reversed; X is then negative, and Y is ground. This causes the 1534 circuits to use the 5 percent low read slice level which is appropriate for weakly magnetized tapes.

Read Enable - The outputs from the seven 1534 Slicing Rectifier circuits are applied to the seven channels of the 1535 Peak Detector circuits shown in -(3):A-C2. These circuits are enabled only when both the GO flip-flop (-(2):A6) and the RDE flip-flop (-(1):C8) are in the 1 state. This read enable gating ensures that the peak detectors are enabled only when the tape is moving at full speed. The RDE flip-flop is not set until the first MP pulse following IOT

7104. This MP pulse originates in the tape control as the MDF pulse marking the end of the UPS acceleration state (for normal speed commands). Both the GO flip-flop and the RDE flip-flop are cleared at the 0 ⟶ CM pulse. For rewind commands not involving a turn around sequence, GO is not set, and the peak detectors are not enabled. For rewind commands that do involve a turn around sequence, the peak detectors are enabled, but because the transport read signals are disabled, no reading can occur.

Peak Detector Outputs - When enabled and pulsed by an output from the corresponding slicing, rectifier channel, each peak detector generates a negative output pulse 1 ⟶ RBn (-(3):A-C2). The negative output pulse from each peak detector is inverted, and the resulting positive pulse is applied to the complement input of the corresponding RB flip-flop (setting it to the 1 state because it was cleared at the preceding 0 ⟶ RB pulse).

RBCP and RBEF

The RBCP level (read buffer character present) is asserted whenever any of the six read buffer data bits RB0-5 contain 1; see -(3):C3. This level is sent to the tape control, where it is ORed with RB6(1) to produce the CHP (character present) signal.

The RBEF, read buffer end-of-file level, (-(3):C2) is asserted and applied to the tape control when the six data bits of the read buffer RB contain the file mark character (17 octal).

End-of-File Flag

The EFF flip-flop (-(1):B7) is set to 1 to indicate that a file mark has been sensed. Although the EFF flip-flop is physically located in the interface, the signals controlling its set and reset originate in the tape control. The interface sends an RBEF level to the tape control whenever a file mark character (17 octal) is read. If RB6(0) is also asserted, indicating BCD parity, and the character is the first data character of the current record, the EOF level is asserted, and the character is treated as a file mark.

At the RCP pulse (which coincides with the end of the CSD delay) the tape control applies a 1 ⟶ EFF ground pulse to interface input terminal 3B81M, setting EFF (-(1):B7). If additional data characters are in the current record, the file mark indication is assumed to be the result

of a programming error, and EFF is reset at the EOR 1 pulse that ends the record. The tape control accomplishes this by applying a 0 ⟶ EFF ground pulse to interface input terminal 3B88K. The EFF is also reset at the beginning of each command by STATUS CLEAR (designated MCL within the 516 Tape Control).

Read Parity Net

As each input character is read into the read buffer RB0-6, the read parity net (-(3):C and D4-8) checks the character for correct lateral parity. If the parity is not correct, a negative TRPE (tape read parity error) level is sent to the tape control.

The read parity circuit is vitually identical to the write parity net described previously under Write Parity Net, except that it contains one extra leg. Both the PAR 0 and PAR 1 outputs of the CB21 flip-flop are used as inputs to the read parity net. When binary (odd) parity is specified, the PAR 1 level is negatively asserted; and when BCD parity is specified, PAR 0 is negatively asserted. The RB inputs are also negative assertion levels. A parity error TRPE is asserted when PAR 0 is asserted, and an odd number of the seven RB bits RB0-6 contain 1s; TRPE is also asserted when PAR 1 is asserted and an even number of the RB bits contain 1s.

Parity Error Flip-Flop

The parity error flip-flop PER (-(1):B7) is set to 1 to indicate that a lateral parity error has occurred. The PER flip-flop is physically located in the interface, but the signals controlling its set and reset originate in the tape control. The interface sends a TRPE level to the tape control whenever the character currently being read displays an apparent parity error. If the character is a data character rather than an EOR mark, the tape control sets the parity error flip-flop by applying a ground pulse to interface input terminal 3B81N. However, if the character displaying the apparent parity error is an EOR mark, MCD(1) is not asserted at RCP time, and the parity error flip-flop is not set. (The EOR mark serves as the longitudinal parity check character and may or may not have correct lateral parity. For this reason, it is necessary to prevent any TRPE signals that are produced by EOR marks from setting the PER flip-flop).

If the EFF is set to 1, it causes the PER flip-flop to be reset to 0 at EOR 1. This removes the false parity error that would appear if the file mark were read in binary parity through a programming error. The PER is also reset at the beginning of each command by STATUS CLEAR (designated MCL within the 516 Tape Control).

CHAPTER 10

MAINTENANCE

GENERAL

The Magnetic Tape Control 516 and the Tape Control Interface 520, 521, or 522A are ordinarily used as peripheral systems of the PDP-6 installation. The maintenance chapters of the PDP-6 Circuits manual and the PDP-6 Processor and Memory manuals include maintenance procedures and suggestions of general relevance not repeated in this manual. Familiarity with this material is essential for the efficient maintenance of the tape control, the interface units, and all other DEC systems used with PDP-6. The topics covered include: Tools and Test Equipment, Removal and Replacement of Modules, Module Troubleshooting, Use of Marginal Check, System Troubleshooting, and Maintenance Logs.

CAUTION

> The procedures described in the maintenance chapters of the PDP-6 Circuits manual and the PDP-6 Processor and Memory manuals should be thoroughly understood before undertaking troubleshooting and repair of the Magnetic Tape Control 516 and the Tape Control Interface 520, 521, or 522A.

USE OF DRAWINGS

The complete system logic of the Magnetic Tape Control 516 and the Tape Control Interfaces 520, 521, and 522A is shown in the engineering logic drawings of Chapter 13, Vol. II. Because these engineering logic drawings are the most frequently used source of troubleshooting information, it is important to be familiar with the conventions and symbols which they employ.

Figure 10-1 shows the standard cabling and module location numbering used within the DEC module racks.

EQUIPMENT LAYOUT

The tape control logic is contained in four DEC logic racks designated from top to bottom 3C-3F (see Figure 10-1). The logic for Interface Units 520 and 521 is contained in logic rack 3B. The 522A Interface requires two racks, 3A and 3B.

The module location and use are shown in the five module location diagrams in Chapter 13, UML-516 (2 drawings), UML-520, UML-521, and UML-522A. Circuit schematics of all modules are presented in order of module type number in Chapter 13.

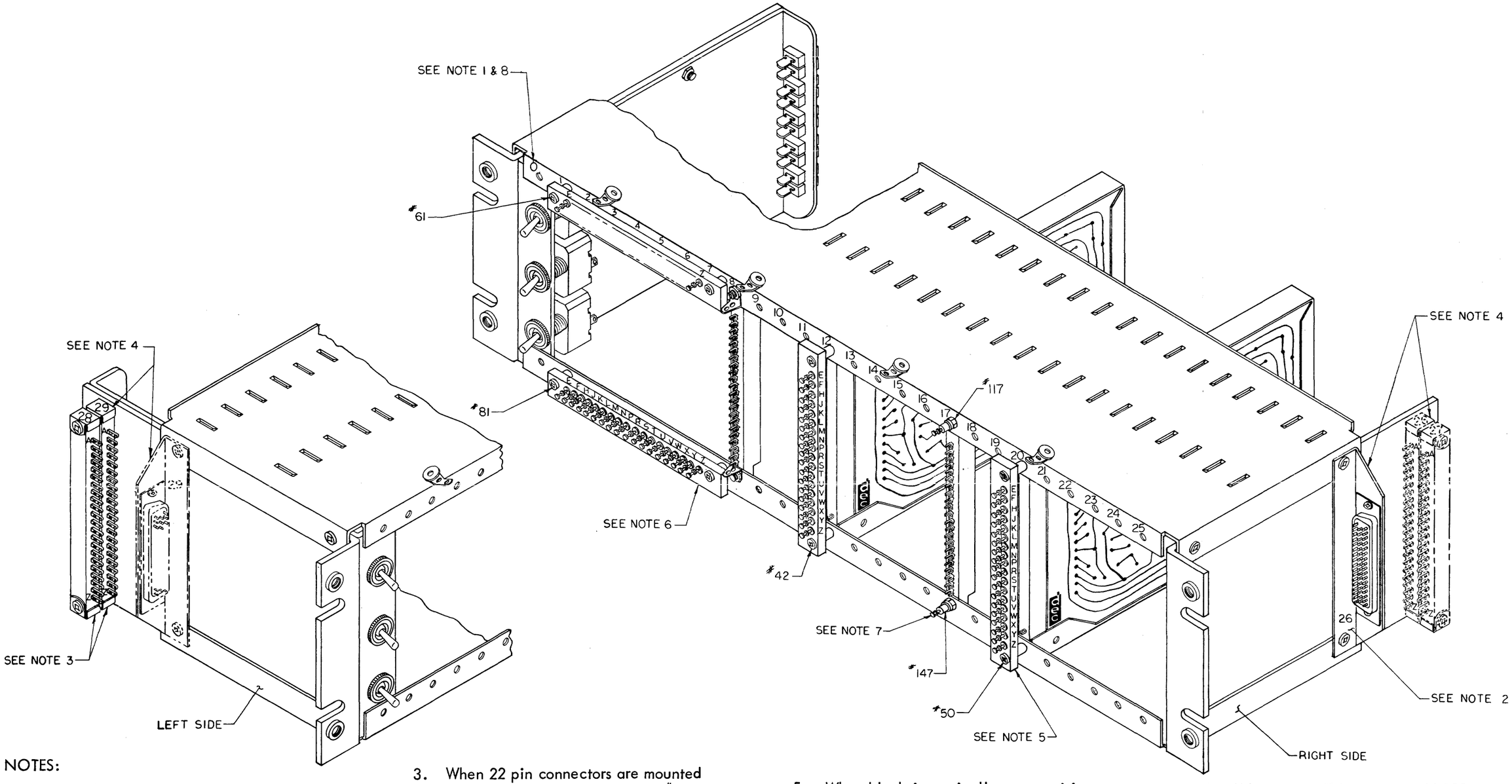
PREVENTIVE MAINTENANCE

The cooling fan in the bottom of the bay should be checked daily for proper operation and free flow of air. Under normal operating conditions, the air filter at the bottom of the bay should be changed and cleaned monthly. The correct procedure for changing and cleaning filters is described in the PDP-6 processor manual maintenance chapter.

Preventive maintenance procedures for the system logic should also be scheduled on a regular basis. The same considerations that call for preventive maintenance of the PDP-6 processor apply equally to the peripheral systems. Appropriate maintenance procedures will detect and eliminate most potential malfunctions before they can cause operating errors. Refer to the PDP-6 processor manual for suggested procedures and schedules. The same basic techniques that are used in maintaining the processor (trouble isolation methods, maintenance programs, marginal check, etc.) are equally useful for maintaining the tape control and interface units.

ADJUSTMENT AND CALIBRATION

There are five 4303 Integrating Single-Shot modules in the Magnetic Tape Control 516 and three such modules in the Tape Control Interface 522A. Each of these modules is used to produce a number of programmed delays. Normally, no adjustment or calibration is required; the delays are set to the proper time constants before the system leaves the factory and there is no appreciable drift. Similar considerations apply to the two 4407 Crystal Clocks in the tape control. Nevertheless, clock frequencies and delay time constants are given below with



NOTES:

1. Numbers 0 thru 25 are called basic numbers.
2. When 50 pin connector is mounted on right side of panel, it is #26; when mounted on left side of panel, it becomes #29.
3. When 22 pin connectors are mounted on left side of panel, they are #28 and #29; when mounted on right side of panel, they become #26 and #27.
4. At no time will a 50 pin connector and 22 pin connector mount on the same side.
5. When block is vertically over position, add 30 to basic number.
6. When block is horizontal on top of panel, add 60 to basic number. When block is horizontal on bottom of panel, add 80 to basic number.
7. When standoff is on top, add 100 to basic number. When standoff is on the bottom, add 130 to basic number.
8. Numbers will always read from left to right, when viewed from the front, or wired side, of the rack.

Figure 10-1 Digital Cable and Component Numbering

procedures for adjustment and calibration. Note, however, that these procedures should not be made a part of routine maintenance schedules. Adjustment and calibration of clocks and delays should be undertaken only if it is necessary.

Crystal Clocks

The two 4407 Crystal Clocks CK1 and CK2 (shown in —WT:B5 and B6) are adjusted to produce output pulses at twice the character rate of the tape transport operated at 800 density and 556 density, respectively. Clock frequency is proportional to tape speed and depends upon the type of transport used. In Table 10-1, the correct frequency and period are listed for both clocks used with each of the four available tape transports. The adjustment can be made in a single step by mounting the clock on a plug-in extender and monitoring its output with an oscilloscope while varying the setting of the 20K internal potentiometer.

TABLE 10-1 CRYSTAL CLOCK FREQUENCY AND PERIOD

Clock	Frequency pps	Period μsec
	Tape Transport Type 50	
CK1 (D800)	120 kc	8.33
CK2 (D556)	83.3 kc	12.0
	Tape Transport Type 570	
CK1 (D800)	180 kc @ 112.5 ips	5.55
	120 kc @ 75 ips	8.33
CK2 (D556)	125 kc @ 112.5 ips	8.0
	83.3 kc @ 75 ips	12.0
	Tape Transport Type 729 Mod 5	
CK1 (D800)	120 kc	8.33
CK2 (D556)	83.3 kc	12.0

TABLE 10-1 CRYSTAL CLOCK FREQUENCY AND PERIOD (continued)

Clock	Frequency pps	Period μsec
	Tape Transport Type 729 Mod 6	
CK1 (D800)	180 kc	5.55
CK2 (D556)	125 kc	8.0
	Tape Transport Type 7330	
CK1 (D800)	56.6 kc	17.3
CK2 (D556)	40 kc	20.0

MOD Delays

The motion delay logic (–MD) contains a 4303 Integrating Single Shot module MOD and a set of twelve logically gated motion delay potentiometers. The correct time constants for each of the resulting delays are listed in Table 4-1. The delays vary depending upon the type of transport used.

Six of the twelve delays are UPS acceleration delays. These delays are calibrated by monitoring the UPS output of the tape status quint-flop (–CS:C2) with an oscilloscope while giving an appropriate series of commands. For the two UPS load point delays (D2 and D4), a series of write file commands alternating with rewind commands is given. The other four UPS delays are monitored while giving a series of write file commands in noncontinuous mode.

The three ERM tape run-out delays are calibrated in much the same way, except that the ERM output of the tape status quint-flop (–CS:C4) is the terminal monitored. Similarly, the two MOS deceleration delays are monitored at the MOS output of the quint-flop (–CS:C5). The twelfth and final delay D0 need not be calibrated. The function of this delay is to keep the MOD 4303 circuit operative during TCR and RIP; as long as the coarse-delay capacitor connection is not tampered with, the setting of the fine-delay potentiometer does not matter.

Other Tape Control Delays

Besides the MOD circuit described above, there are four other 4303 Integrating Single Shot modules in the tape control. These four modules are:

1. EWD - End of Record Write Delay (–WT:C7)
2. CSD - Character Skew Delay (–RD:B7)
3. MCD - Miss Character Delay (–RD:B7)
4. ERD - End of Record Delay (–RD:B7)

Associated with each of these four delay modules are three logically gated delay potentiometers, one for each of the three tape densities, D200, D556, and D800. The delays also vary depending upon the type of transport used. The correct time constants are listed in Table 10-2.

The delays listed in Table 10-2 are checked directly at the delay outputs while giving a series of write file commands. (The reason for checking the MOD delays at the quint-flop rather than at the delay outputs is the difficulty of differentiating between successive delays from the same MOD module.)

Delays in Interface 522A

The 522A Interface contains three 4303 Integrating Single Shot modules, and one 4301 Delay module. These four modules are:

1. GO 2 - Start Delay (UML-516(1):A2)
2. LPD - Load Point Delay (UML-516(1):C3)
3. TAD - Turn Around Delay (UML-516(2):B5)
4. Write Pulse Duration Delay (UML-516(2):A7)

The TAD delay has four logically gated delay potentiometers M1 through M4, each of which produces a different delay period. The other three delays each produce only a single delay period. The delay intervals vary depending upon the type of transport used. The correct time constants are listed in Table 10-3.

TABLE 10-2 TAPE CONTROL DELAY TIME CONSTANTS

Delay Density	Time Constant μsec D200	D556	D800
	Tape Transport Type 50		
EWD	270	97.5	69.5
CSD	35	12	9
MCD	100	36	25
ERD	230	86	58
	Tape Transport Type 570		
EWD	180	65	46
CSD	23.5	8.0	5.0
MCD	67	24	17
ERD	156	56	38
	Tape Transport Type 729 Mod 5		
EWD	270	97.5	69.5
CSD	35	12	9
MCD	100	36	25
ERD	230	86	58
	Tape Transport Type 729 Mod 6		
EWD	180	65	46
CSD	23.5	8.0	5.0
MCD	67	24	17
ERD	156	56	38
	Tape Transport Type 7330		
EWD	556	200	not used
CSD	50	25	not used
MCD	200	75	not used
ERD	470	160	not used

TABLE 10-3 INTERFACE 522A DELAY TIME CONSTANTS

Delay Tape Transport Type	Time Constant 729 Mod 5	729 Mod 6	7330
GO 2	30 μsec	20 μsec	30 μsec
LPD	16.5 msec	11.0 msec	250 msec
TAD-M1	12.0 msec	8.0 msec	25 msec
TAD-M2	5.0 msec	5.0 msec	10 msec
TAD-M3	12.0 msec	6.0 msec	250 msec
TAD-M4	5.0 msec	4.0 msec	10 msec
Write Pulse	4.0 μsec	3.0 μsec	12 μsec

PROGRAMMING SUGGESTIONS

A few programming features of the Tape Control 516 and the Interface Units 520, 521, and 522A, which may be of special interest to maintenance personnel are summarized below. Refer to the PDP-6 programming manual for more programming information.

Successive CONO 220 Commands

It is not necessary to wait until the completion of a given CONO 220 command to program a second one. The command hold buffer is free, and another command can be given whenever the command sync quad-flop is reset to XNC state.

Writing Past Tape End Point

There is no logical restriction to prevent writing beyond TEP. Consequently, the programmer must use care to avoid writing too far and winding the tape off the supply reel.

Leaving Rewinding Transports Unselected

While a given transport is rewinding, the programmer may wish to use the tape control to give a new command to another transport. Do not give the nonselected rewinding transport any further commands until its deceleration is complete. (For an explanation of why such premature

commands would be undesirable, refer to Chapter 5, Deceleration Interval.) By keeping the rewinding transport selected, the LIF flag can be used to give a load point interrupt as soon as the tape has come to a full stop. This is the best procedure if the programmer wishes to use the rewinding transport as soon as possible.

Programmed JNU Pulse

The programmed JNU pulse (-CS:A, B6) provides a means of monitoring the status of any tape transport without applying any command to the transport being monitored. This option is particularly useful when operating with 521 Interface Units in multicontrol mode; refer to Chapter 8. The programmed JNU permits checking the status of a transport that is currently selected to the other tape control.

The programmed JNU is used in the following manner. First, a CONO 220 NOP instruction is given with bits 29-31 containing the device selection code of the transport that is to be monitored. This NOP instruction stores the device selection code in CHB29-31. Next, a CONO 224 instruction is given with a 1 in bit 34. The IOB CONO SET of this CONO 224 instruction produces a programmed JNU pulse which advances the device selection code from CHB29-31 to CB29-31.

The purpose of the two CONO instructions described above is to transfer the device selection code of the transport that is to be monitored into CB29-31. Once the device selection code is stored in CB29-31, the status of the selected transport can then be monitored at will by using a CONI 224 to sample the tape control transport-status register.

There is, however, one programming refinement which should be kept in mind when using the programmed JNU. Every CONO 224 instruction sets the four priority interrupt enable switches EFE, LIE, ICE, and XNE to the states determined by bits 33, 20, 35 and 21 of the CONO 224 instruction. Unless these four flip-flop switches are to be cleared, it is important to code all CONO 224 instructions with 1s in the correct bit positions to maintain the current switch settings.

Programmed CHP Pulse

The programmed CHP pulse is used as a maintenance aid. It permits the tape control to be operated without the use of an interface or transports. Each CONO 230 instruction produces a single CHP pulse at the IOB CONO SET (–RD:D8), simulating the reading of a single character from tape. The CONO 230 instructions can be programmed at any desired frequency in order to match the transport character rate.

Simulation of the interface and transport is completed by jumpering the CTT pulse (–CS:B8) back to the START 2 input (–CS:B3) to produce a simulated STM pulse, and by grounding T READY (–ST:B7).

MAINTENANCE PROGRAMS

At present, there are three MAINDEC programs available for the Magnetic Tape Control 516. The first program facilitates oscilloscope testing; the second program yields hard copy that shows system timing relationships without the use of an oscilloscope, and the third program provides an overall summary of system performance. Brief descriptions of these three programs are presented below.

Cursory Operation Test for Type 516 Tape Control

This simple program permits the repetition of a single command or an ordered sequence of commands throughout an entire tape. The commands making up the sequence are in a fixed order, but by DATA switches at the console any of these commands may be eliminated from the sequence. All command parameters such as slice level, parity mode, density, etc. are under switch control. Three characters (18 data bits) are set up from the console; these three characters are duplicated in the left and right half of each word written on the tape. Record length is controlled by varying a constant in memory.

Dynamic Status Presentation for Type 516 Tape Control

This real time maintenance program checks the relative timing of the various tape control state transitions by printing out the contents of all three 18-bit status registers at fixed-time intervals.

The time standard is based on the 60-cps clock and is sufficiently accurate to calibrate the tape control clocks and delays in terms of the numerical time constants given in Tables 10-1 and 10-2. A second operating mode of this program will give decimal readings of specific delay times.

PDP-6 Magnetic Tape Data Test 516

This program thoroughly tests the data handling accuracy of the Magnetic Tape Control 516. The program is designed to measure the system's current level of performance rather than to locate such malfunctions as may be detected. Once it has been determined that the tape control is not performing properly, the Cursory Operation Test and the Dynamic Status Presentation programs described above are used to locate the malfunction that is producing the errors.

Typical error printouts for the data test are listed below.

Normal Error Printout

OPER:	READ						
SR200:	004022	SR220:	000001	SR224:	240037	SR230:	000030
REC No.	LNGTH	No.ERRORS	PAR.	DENS.	SLEV.	UNIT No.	PAR.ERRORS
185	4096	3	E	L	H	6	Lat. Long
Word No.	WRITTEN	010111	010111	010111	010111	010111	010111
3700	READ	010101 ↑	010110 ↑	010101 ↑	010111	010101 ↑	010111

Summary Printout

RELIABILITY RUN			
UNITS TESTED:	0	1	2
FULL LNGTH RCDS:	13	57	102
RD ERR RCDS:	0	0	0
RD CMP ERR RCDS:	0	0	54
RD BCK ERR RCDS:	0	0	0
TOTAL WORDS:	1664	7296	13056

Summary Printout (continued)

RD ERR WDS:	0	0	0
RD BCK ERR WDS:	0	0	0
WR PAR ERRS:			
RECOVERABLE:	0	0	2
PERMANENT:	0	0	0
RD PAR ERRS:			
RECOVERABLE:	0	0	0
PERMANENT:	0	0	0
RD BCK PAR ERRS:			
RECOVERABLE:	0	0	0
PERMANENT:	0	0	0
RD CMP PAR ERRS:			
RECOVERABLE:	0	0	0
PERMANENT:	0	0	0
ABORTED RECORDS:	0	0	0
SHORT RECORDS:	0	0	0

RECOMMENDED SPARE PARTS

The most economical quantity of spare parts to be maintained depends on the requirements of the individual user, but the following listing is suitable for most installations.

Module Spares

Quantity: One module of each type that is used in the tape control or interface (or both).

Module	Used in			
	Tape Control 516	Interface 520	Interface 521	Interface 522A
1110	X			
1151	X			
1534			X	X
1535			X	X
1539		X		
1684		X		
1685			X	
1690		X	X	
4102	X	X	X	X
4106		X	X	X
4111		X	X	X
4112	X		X	X
4113	X	X	X	X
4114	X			
4115	X			X
4118	X			
4127	X	X		X
4213	X			
4215	X	X	X	X
4217	X			
4218	X			

Module	Used in			
	Tape Control 516	Interface 520	Interface 521	Interface 522A
4301				X
4303	X			X
4304	X			X
4305	X			
4407	X			
4505*				X
4506				X
4604	X		X	X
4606	X			
4657	X			
4659				X
4660*				X
6684	X			

*Required only for systems using tape transports Type 7330.

Component Spares

Type	Quantity			
	Tape Control 516	Interface 520	Interface 521	Interface 522A
D-668	11	--	6	7
DEC 2894-1	21	8	8	11
DEC 2894-3	4	--	--	--
1N276	18	13	11	14
1N645	24	12	13	16
1N748 - 3.9v	4	--	--	--
1N914	7	5	5	7
1N994	15	2	2	5
1N3208	6	--	--	--
1N3605	5	--	4	5
1N3606	28	14	12	15
2N393	--	--	6	--
2N599	5	--	--	--
2N1304	3	4	9	9
2N1305	10	9	11	12
2N1309	6	--	4	8
2N1499	5	--	--	4
2N1499A	12	6	--	4
2N1754	4	--	--	--
2N2451	3	4	4	--
2N2714	6	--	4	4
2N2904	4	--	--	--
2N3009	5	--	--	4
16J1	24	11	13	16

Mechanical Spares

Part Number and Description	Quantity
53E168, Type CFG: Rotron fan with #2R blade	1
X-1431, 10' x 10" x 2" EZ Kleen Filter	1
Type 418 Super Filter Coat, pints	1

CHAPTER 11

INSTALLATION AND PREOPERATIONAL CHECK-OUT

SITE SELECTION

Before installing the tape control and interface, select a suitable location near the PDP-6 arithmetic processor. The tape control and interface should be located within 25 feet of the processor, and the tape transports should be located within 25 feet of the tape control.

The tape control and interface are located in a cabinet 69-1/2 inches high, 22-1/4 inches wide, and 27-1/8 inches deep. A 3-foot clearance should be allowed on all sides of the equipment for access during maintenance. A level floor is required because the equipment frames are mounted on casters. The floor should be capable of supporting 150 psf. The system is designed to operate efficiently from 50° to 100°F. The plug-in modules are cooled by blowing air out the front of the bay. No additional cooling equipment is required.

The tape control and interface run on ordinary 115-volt, 600 cycle current. A 10-ampere line is sufficient. The tape control power cable is equipped with a Hubble Twist-Loc 3-prong, 30-ampere, 250-volt plug. The same plug and cable also serve the interface. Although the tape control interface combination draws only 5 to 6 amperes in normal operation, turn-on surges may reach 9 amperes, and system power capacity should be planned accordingly.

UNPACKING

The tape control and interface cabinet is shipped on a skid and may be crated or not, depending on the mode of transportation. For truck shipment, it may be left uncrated. A crate is furnished for air shipment. The crate is approximately 74 inches high, 27 inches wide, and 32 inches deep. The skid upon which it rests is about 6 inches high and 3 feet square. Interconnecting cables are specially made up for each installation and are ordinarily shipped with the equipment.

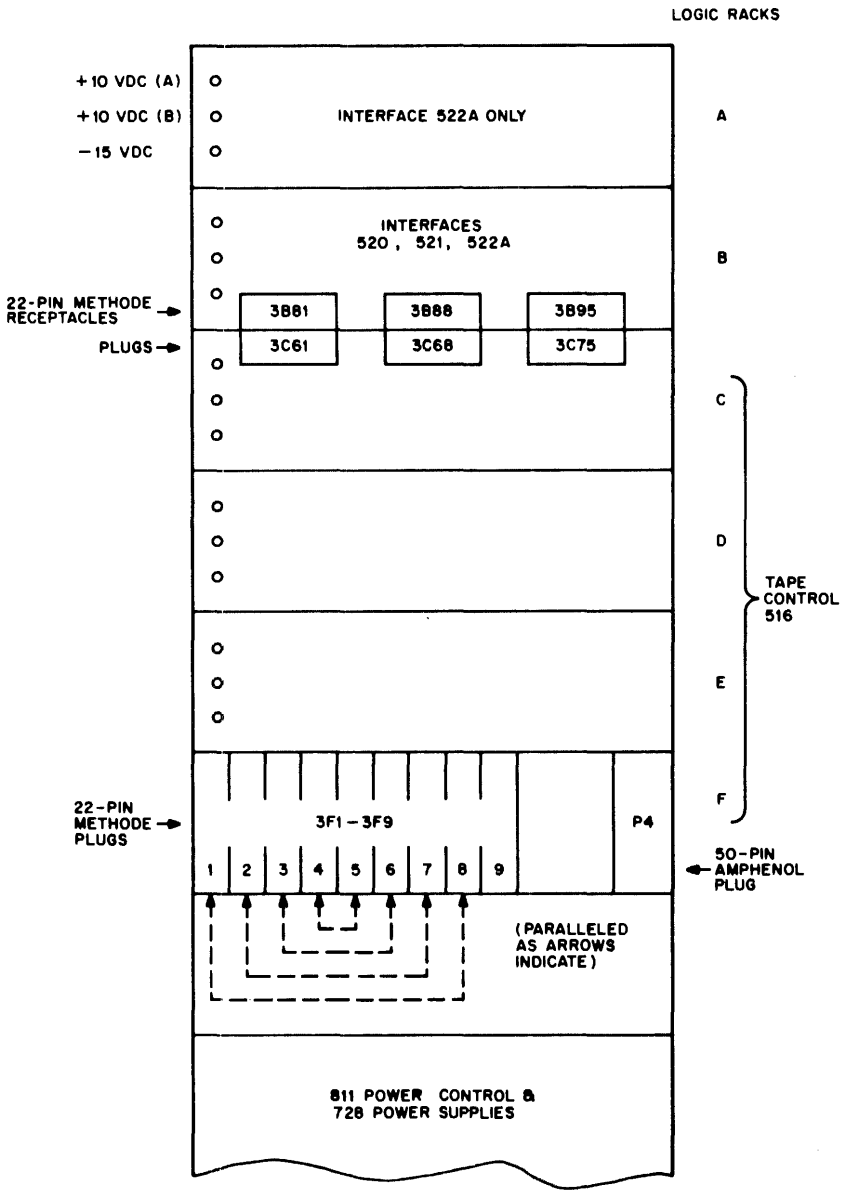


Figure 11-1 Front View Layout, Tape Control 516 and Interface 520, 521, or 522A

1. If the tape control and interface are crated, carefully remove all crating and strapping, and any packing material. If the unit is shipped uncrated, remove the skid and any protective padding.

2. The plenum doors at the rear of the bay have spring catches. To reinforce these doors during shipment, two bolts are used to hold each door shut. Remove these bolts and store them in the plastic loops provided.

3. Remove any packing material, shipping blocks, etc. from the inside of the tape control and interface cabinet.

4. The plug-in modules are taped into the logic panels to prevent damage in shipment. Remove the tape.

> NOTE: If the user plans to reship the equipment (or to move it more than a short distance) in the near future, special containers and packing materials should be saved for reuse. These items are designed especially to accommodate the equipment and are the safest means of packing it for reshipment.

INSPECTION

The tape control and interface are thoroughly tested and checked before they leave the factory. However, both units should be inspected and checked again when installed to make sure that no damage has occurred during shipment. After the equipment is unpacked, check the following:

1. Have all the shipping blocks, packing materials, tape, etc. been removed? If not, remove them.

2. Are all plug-in units inserted firmly in position? Secure any that are loose.

3. Are there any loose nuts or bolts? If so, tighten them.

4. Are there any loose or broken wires? If so, repair them.

CABLE CONNECTIONS

Complete the inspection procedure as given above before connecting the cables.

1. Route the four cables from the PDP-6 processor into the cabinet, and connect the Methode receptacles at the ends of the processor cables to the corresponding Methode plugs at the left (viewed from front) of rack F. The four receptacles can be plugged in either to Methode plugs 3F1-3F4, or 3F5-3F8, depending on the layout of the specific installation. The plugs are paralleled as shown in Figure 11-1 and as listed in Table 11-1.

CAUTION

It is essential that each of the four Methode receptacles be connected to the correct parallel pair of Methode plugs. The receptacles should each be marked with the correct plug position. If they are not so marked, check Table 11-1 to ensure that the correct connections are made.

TABLE 11-1 CABLE CONNECTIONS, METHODE PLUGS 3F1-3F9

Plug Pair	Processor Signals
3F1-3F8	IOB2-17
3F2-3F7	IOB18-35
3F3-3F6	IOB CONO CLR, IOB CONO SET, IOB ← STATUS, PIR0-7
3F4-3F5	IOB RESET, DEVICE SELECT IOS3-9
Plug	**Data Control 136 Signals**
3F9	RBB0(1)-(RBB5(1), DA0(1)-DA5(1), DA RQ B(1)

2. Route the cable from the Data Control 136 into the cabinet, and connect the Methode receptacle at the cable end to Methode plug 3F9.

3. Secure the tie-down brackets over all five Methode receptacles to prevent them from being accidentally loosened or dislodged.

4. Plug in and secure P4, the 50-pin Amphenol receptacle at the end of the cable from the tape transports (the corresponding plug is located at the right of rack F).

5. A coiled ac power cable is taped to the fan at the bottom of the bay. Remove the tape and route the power cable out through the hole in the bottom of the bay, but do not plug it in.

POWER CONTROLS

To perform the preoperational check-out of the tape control and interface, the operator should be familiar with the controls described below.

Power Control Type 834-836

There are two controls on this unit, a circuit breaker and a LOCAL/REMOTE switch. When the switch is in the LOCAL position, the processor has no effect on tape control and interface power turnon. Power can then be turned on or off by means of the circuit breaker. The LOCAL position is used primarily for maintenance purposes.

For normal operation, it is usually more convenient to leave the circuit breaker on and the switch in the REMOTE position. Tape control and interface power is then controlled by the processor. When the processor is turned on, power is also applied to the tape control and interface. If the circuit breaker is turned off, no power is applied to the tape control and interface regardless of the position of the LOCAL/REMOTE switch.

MCV Switches

There are three MCV (Marginal Check Voltage) switches at the left of each logic rack. The top switch in each set governs the +10 vdc (A) power lines, the middle switch governs the +10 vdc (B) power lines, and the bottom switch governs the -15 vdc lines (see Figure 11-1). There are two independent +10 vdc supply lines, A and B; these lines are applied to different

circuits within the same modules. This arrangement is an aid to troubleshooting because it permits selective application of marginal check voltages. In the FIXED position (down), each MCV switch connects the associated logic rack to the fixed supply voltage from the internal 728 Power Supplies. In the MARGINAL position (up), each MCV switch connects the associated logic rack to the variable MCV voltage from the processor. This variable MCV voltage can be adjusted at the MCV controls over the operator's control panel of the PDP-6.

PREOPERATIONAL CHECK-OUT

Before using the tape control and interface, make sure that the system turns on and off properly and that the correct voltages are present at all logic racks. The following check-out procedure should be carried out after completing the cable connections described under Cable Connections. All voltages are measured from chassis ground with a Multimeter (Simpson Model 260A, Triplett Model 630NA, or equivalent).

1. Put the LOCAL/REMOTE switch in LOCAL position.
2. Turn power circuit breaker OFF.
3. Plug in power cable to 110-volt ac outlet.
4. Put all MCV switches in FIXED position.
5. Turn power circuit breaker ON.
6. Check the fixed (internal) supply voltages at terminals A, B, and C of the module occupying the extreme right (viewed from the front) position of each logic rack.

Terminal	Voltage
A	+10 vdc (A)
B	+10 vdc (B)
C	-15 vdc

The +10-vdc fixed supply voltages should be between +9.5 vdc and +11.5 vdc. The −15 vdc should be between −14.5 vdc and −16.5 vdc. If either voltage falls outside the specified range, the 728 Power Supply probably needs maintenance.

7. With the PDP-6 processor turned off, put the LOCAL/REMOTE switch in the REMOTE position. Check each of the three fixed voltages at any of the terminals specified in step 6. No voltage should be present.

8. Turn on the PDP-6 processor. Again check the three fixed voltages. The same voltage noted in step 6 should be present at each of the three terminals checked.

9. Put the LOCAL/REMOTE switch in the LOCAL position.

10. Put the MCV switches that control the +10-vdc (A) lines (upper switch at the left of each logic rack) in the MARGINAL position.

11. Make the following settings at the MCV controls above the PDP-6 operator's control panel:

a. Set polarity switch to +10-volt position.

b. Adjust Variac until marginal check voltage meter indicates +5 vdc.

12. Check the +10-vdc (A) voltage at the terminals specified in step 6. It should coincide with the + 5-vdc voltage shown on the marginal check voltage meter.

13. Return the MCV switches for +10 vdc (A) to the FIXED position and set the MCV switches which control the +10-vdc (B) lines (middle switch at the left of each logic rack) to the MARGINAL position.

14. Check the +10-vdc (B) voltage at the terminals specified in step 6. It should coincide with the + 5-vdc voltage shown on the marginal check voltage meter.

15. Return the MCV switches for +10 vdc to the FIXED position.

16. At the PDP-6 MCV controls, set the MCV polarity switch to the −15 volt position and adjust the Variac until the marginal check voltage meter indicates − 8 vdc.

17. Set the MCV switches which control the −15-vdc lines (bottom switch at the left of each logic rack) to the MARGINAL position.

18. Check the −15-vdc voltage at the terminals specified in step 6. It should coincide with the −8-vdc voltage shown on the marginal check voltage meter.

19. Return the MCV switches for −15 vdc to the FIXED position.

20. Return the LOCAL/REMOTE switch to REMOTE.

21. Return the polarity switch at the PDP-6 to the off position.

22. Turn off the PDP-6.

H-616

JOSS CONSOLE INSTRUCTION MANUAL

PDP-6

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

H-616

PDP-6 JOSS CONSOLE INSTRUCTION MANUAL

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

COPY NO.

CONTENTS

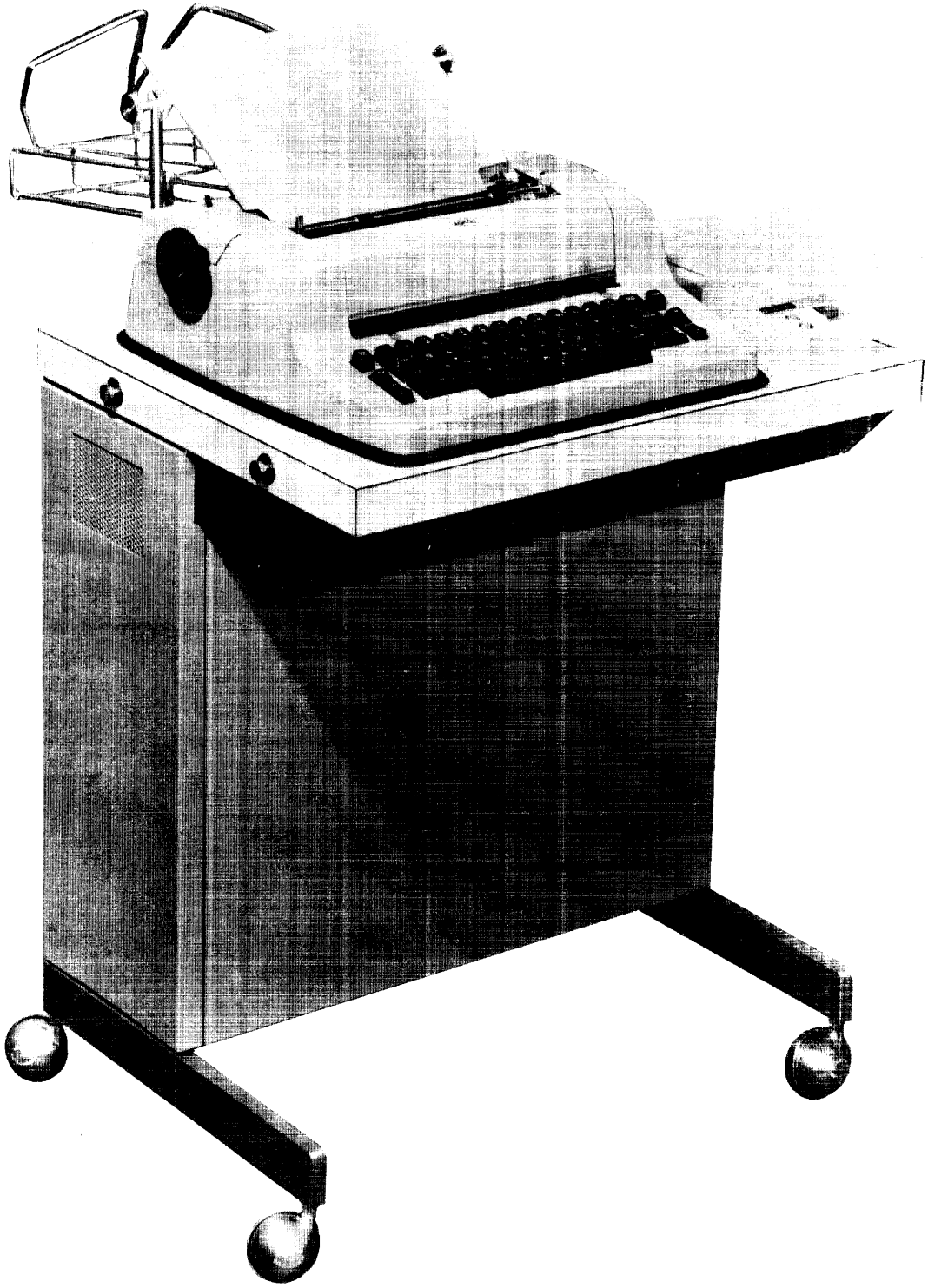
CONTENTS (continued)

ILLUSTRATIONS

TABLES

ENGINEERING DRAWINGS

ENGINEERING DRAWINGS (continued)



JOSS Console

CHAPTER 1
INTRODUCTION

1.1 FUNCTIONAL DESCRIPTION

The JOSS Console is a remote computer input/output communication station. The central element of the JOSS Console is an IBM Model 731 Selectric I/O writer, which can also be used in off-line mode as an office typewriter when desired. The typewriter is modified electrically, and mechanically, for PAGE GO and READY/HOLD signals, to provide a forms-feed mechanism to accommodate 8-1/2 by 11-inch pinfeed, fanfold paper. A control box permits operator control of the JOSS Console and reflects the status of the console and typewriter. The controls are a console POWER ON/OFF switch and an INTERRUPT switch to send a special signal.

When the power is turned on at the control box, an ON signal is sent to notify the computer of the power application. A CONSOLE POWER lamp on the control box illuminates to indicate that power has been successfully applied to the JOSS Console. The computer acknowledges receipt of the power-on signal by sending a system-turn-on signal to the JOSS Console. This signal illuminates the JOSS SYSTEM lamp on the control box and applies power to the typewriter. A short delay permits the typewriter motor to reach operating speed, after which the TYPEWRITER lamp on the control box lights and a READY signal informs the computer of the typewriter ready condition.

The JOSS Console has two control modes: red mode and green mode. Following power application, the JOSS Console is in the red mode. A red or a green lamp on the control box illuminates to indicated the mode of operation. During red mode, the computer controls the typewriter, and the typewriter keyboard is locked to the JOSS Console operator to prevent console-to-computer communication. During green mode, the user controls the typewriter and the system ignores all incoming characters and functions. JOSS Console electronics encode any characters typed on the typewriter or any operational functions executed on the typewriter (e.g. space, shift) and sents them to the computer.

Control of the operating mode is normally proprietary; i.e., the user relinquishes control by giving a CARR RTN or PAGE GO. The computer relinquishes control by sending a SYS RED. (The computer may preempt control by a switch to red, if necessary).

When power is turned off at the control box, a short delay occurs during which a signal (turn-off) informs the computer that power has been removed. After the delay, power is removed (turned off) from the JOSS Console.

An ON/OFF switch on the typewriter permits the operator to use the typewriter in off-line mode only when all console power has been removed. When the typewriter ON switch is depressed with console power removed, the typewriter is turned on. The typewriter then functions as an ordinary typewriter without the PAGE GO facilities.

1.2 PHYSICAL DESCRIPTION

The JOSS Console is a single console which contains the typewriter and the electronics necessary for input and output communications. The console is 29 inches high, 23 inches wide, and 19 inches deep. A removable shelf, on either the left or right side, provides adequate working area for the operator. A normal 120-vac 60-cycle outlet supplies power for the JOSS Console. External power is supplied to a rear panel circuit breaker which supplies power to the system.

1.3 APPLICABLE DOCUMENTS

Digital FLIP CHIP Modules Handbook, C-105
IBM Selectric Input/Output Writer.

CHAPTER 2
THEORY OF OPERATION

2.1 INTRODUCTION

All communication to and from the JOSS Console is via an 8-bit, 11-unit modified Teletype code. Four types of information are conveyed by this code:

a. Characters - Therefore are 44 character keys on the typewriter, each with an upper and lower case interpretation, totalling 88 printing characters.

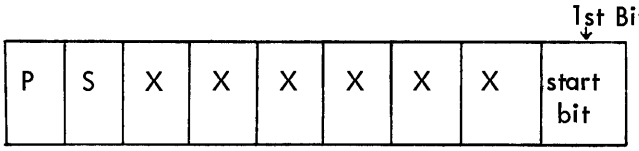
b. Functions - There are seven typewriter functions: shift up, shift down, space, backspace, tabulator, carriage return, and page (forms feed).

c. Signals - Signals to and from the JOSS Console control reflect the changing states of the JOSS Console. These states are described in this section.

d. Status Request/Report - The computer may at any time request a report of the status of the JOSS Console. To satisfy this, the console transmits a special status report code.

When in operation, the status of the JOSS Console is defined by the states of a number of flip-flops (see table 2-1). Lamps on the control panel indicate the power on, system on, ready, red/green, and interrupt-on states.

The 8-bit code format of the word used for data transmission is:



where P is the parity bit (odd) and S is the status request bit. The X bits comprise coded characters, signals, and machine functions as shown in table 2-2.

Power-on and power-off of the JOSS Console consist of a number of steps, performed in sequence. When power is turned on at the control box, the JOSS Console proceeds through a power-on sequence. Similarly, when power is turned off, the JOSS Console proceeds through a similar power-off sequence. These sequences and other operation features from initial application of power to power shut-down are illustrated in the operation flow diagram, figure 2-1.

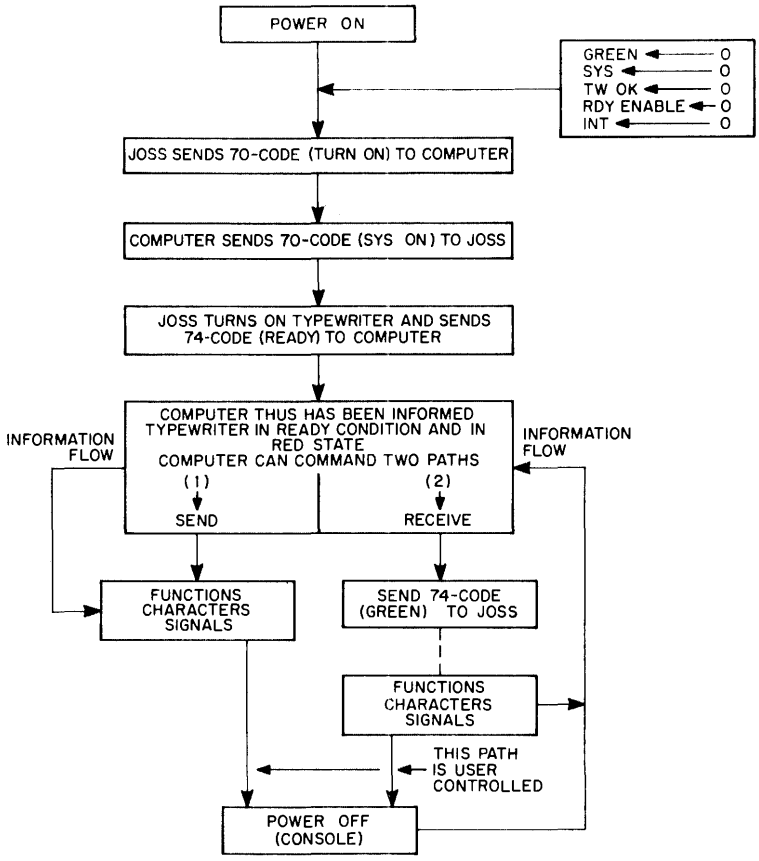


Figure 2-1 Operation Sequence

When power is applied to the JOSS Console, the power sequence applies power to the system, initiates a delay during which control flip-flops are cleared or normalized, and sends a TURN ON signal (70 code) to the computer to indicate that power has been applied to the system. The JOSS Console is not yet fully operational; it must wait for an enabling signal from the computer. The computer does this by sending a SYS ON signal (70 code) to turn on the JOSS Console. System turn-on applies power to the typewriter; however, a typewriter delay permits the typewriter motor to reach operational speed, after which the console sends a READY signal (74 code) to the computer to indicate that the typewriter is ready for operation.

The system is now active. Initial power turn-on placed the JOSS Console in the red state; thus, the computer can communicate with the JOSS Console by sending characters or typewriter machine functions. Each 8-bit word is transferred serially into the JOSS Console. The six bits of a character or

machine function are decoded to perform the specified typewriter operation. During the red state, the typewriter keyboard is locked to prevent the operator from typing which would send meaningless information to the computer.

TABLE 2-1 JOSS CONSOLE STATUS

True State	False State	Comments
Power on	Power off	Determines that electronic power has been applied and the console is ready to use in the on-line mode.
System on	System off	Controlled by the computer. A console is not fully operational until the SYSTEM light is turned on by the computer.
Green	Red	In the green state, the user has control of the typewriter; in the red state, the central processor has control. Transmission of a carriage-return or page-go function automatically sets the red state. It may be set to either state by the computer.
Ready enable	Not ready enable	Reflects the most recent setting of the typewriter ON-OFF switch.
Typewriter OK	Typewriter not OK	Typewriter is not OK if the motor is not up to speed, or if it is in the process of tab, page, or carriage-return functions. Completion of the process sets TYPEWRITER OK.
Ready	Not ready	The console is ready only if TYPEWRITER OK and READY ENABLE are active.
Interrupt on	Interrupt off	Controlled by the centeral processor.
Incoming parity error	No incoming parity error	Set by any incoming parity error; reset only when status report is sent.

TABLE 2-2 JOSS II CONSOLE CODES

	Tens \ Units	0	1	2	3	4	5	6	7
Characters (see note)	0	P p	E e	≠ =	≤ 5	: ;	D d	R r	< 7
	1	Q q	K k	I i	≥ 6	[,	C c	A a	> 8
	2	J j	T t	| •	Z z	G g	X x	M m	' 1
	3	* +	N n	] .	" 2	F f	U u	V v	# 3
	4	Y y	H h	S s	) 0	? /	L l	O o	$ 4
	5	_ -	B b	W w	(9				
Functions	6	SHIFT DOWN	BACK SPACE	TAB	SPACE	CARR. RETN.	PAGE	SHIFT UP	
Signals to Station	7	SYSTEM ON	SYSTEM OFF	INT. ON	INT. OFF	SWITCH to GREEN	SWITCH to RED	SOUND "BEEP"	NO OP
Signals from Station	7	TURN ON	TURN OFF	INT.		READY	HOLD	INT. + READY	INT. + HOLD

NOTE: Both upper case and lower case assignments are shown. Codes 54-57 are not generated by the typewriter; if received, they act as 20-23. The character codes are derived by ordering and interpreting the typewriter input/output lines as follows: $\overline{R1}$, R2, $\overline{R2A}$, R5, T1, T2. Note the negations of R1 and R2A.

During the red state, the operator can place the console in a not-ready condition in order to request the computer to stop sending characters, etc. This is done by depressing the typewriter OFF switch. When this is done, a HOLD signal is sent to the computer so that it is cognizant of the not-ready condition of the console. The computer will not communicate again with JOSS Console until it receives a READY signal. Depressing the typewriter ON switch initiates READY.

The computer can send a signal to the JOSS Console to switch it to the green state which permits JOSS Console-to-computer communications. In the green state, the typewriter keyboard is unlocked and the operator can use the typewriter on-line. For each typewriter character or machine function typed, the console control circuits assemble an 8-bit word and transfer it to the computer. When the operator executes a carriage-return or page-go machine function, the JOSS control automatically switches to the red state.

The computer at any time may request a status report, which the JOSS Console answers after changing state and after a character or signal request. In addition, a status report is initiated if an input parity error occurs, permitting immediate computer recognition of the parity error.

When power is turned off at the JOSS Console, a power-down sequence is performed. During the power-down sequence, a signal is sent to the computer so that it is aware of the power shut-down.

When, and only when the CONSOLE POWER lamp is off at the control panel, can the operator use the typewriter in off-line mode by pressing the typewriter ON switch. The rear-panel circuit breaker must be on and console power removed before off-line power can be applied to the typewriter. The typewriter can then be used as a standard office typewriter without the page-go facilities.

2.1.1 Stare Down

JOSS II was designed to minimize the probability of a "stare down problem"; i.e., a situation in which the console is in one state but the computer thinks it is in another state. For example, if the console is in the red state, but the computer thinks it is in the green state, each is waiting for the other to take some action.

To meet this criterion in the console there are several status toggles available for computer inspection:

a. On state

b. Red/green state

c. Ready/hold state

d. In-request state (This shows the state of a computer generated acknowledgment to an interrupt request by the user.)

The computer can send signals to the console and (optionally) request the state of the status toggles after completion of the action requested. The signals are:

a. NOP: no operation.

b. Set the on state on or off.

c. Set the red/green state to red or green.

d. Set the in-request state on or off.

2.2 DETAILED LOGIC DISCUSSION

The following paragraphs provide a detailed logic discussion of the JOSS Console. Frequent reference is made to the logic drawings in chapter 6. A particular circuit is referenced to a drawing by the last digits of identification followed by a colon and the coordinates of its location on the drawing. For example, the INC LINE signal which appears at coordinates C7 of drawing BS-D-616-0-17 is referenced as 17:C7.

2.2.1 Power-on Sequence

2.2.1.1 JOSS Sends 70 Code to Computer - When power is turned on at the control panel, the POWER CLEAR DLY intergrating one-shot multivibrator (18:B5) reverts to the 1 state. (A later section explains the ac power distribution.) The POWER CLR DLY signal enables the 10-kc clock (18:B5) to generate PWR CLR pulses which normalize control flip-flops in the control circuits. (Refer to timing diagram figure 2-2 and flow diagram figure 2-3.) After 500 msec the PWR CLR DLY one-shot reverts to the 0 state, and the transition sets the TURN ON flip-flop (18:B4). The TURN ON (1) signal enables the OUT SYNC pulse to set the SIGNAL RQST flip-flop (18:A7) and to initiate the 70-code transfer to the computer. The OUT SYNC pulse is the result of ANDing the OUT CLOCK pulse with the OUT DONE signal. The OUT DONE signal is 0v at this time because the initial POWER CLEAR pulses normalized its inputs (10:D1). The OUT CLOCK pulses are derived from a 6-stage counter (CLK0 through CLK5), which counts down the crystal clock frequency of 10.560 kc by a factor of 64. The CLK0 output generates the OUT CLOCK pulse; hence, the OUT CLOCK rate is 165 pulses per sec.

The SIGNAL RQST (1) enables the OSR SIGNALS pulse (18:B2) which is delayed from the OUT CLOCK pulse by 4 μsec. The OSR SIGNALS pulse sets the OUT ENB (output enable) flip-flop (10:A1) and strobes the input gates (10:B4 - 10:A7) to insert a 70 code into the OSR (output shift register).

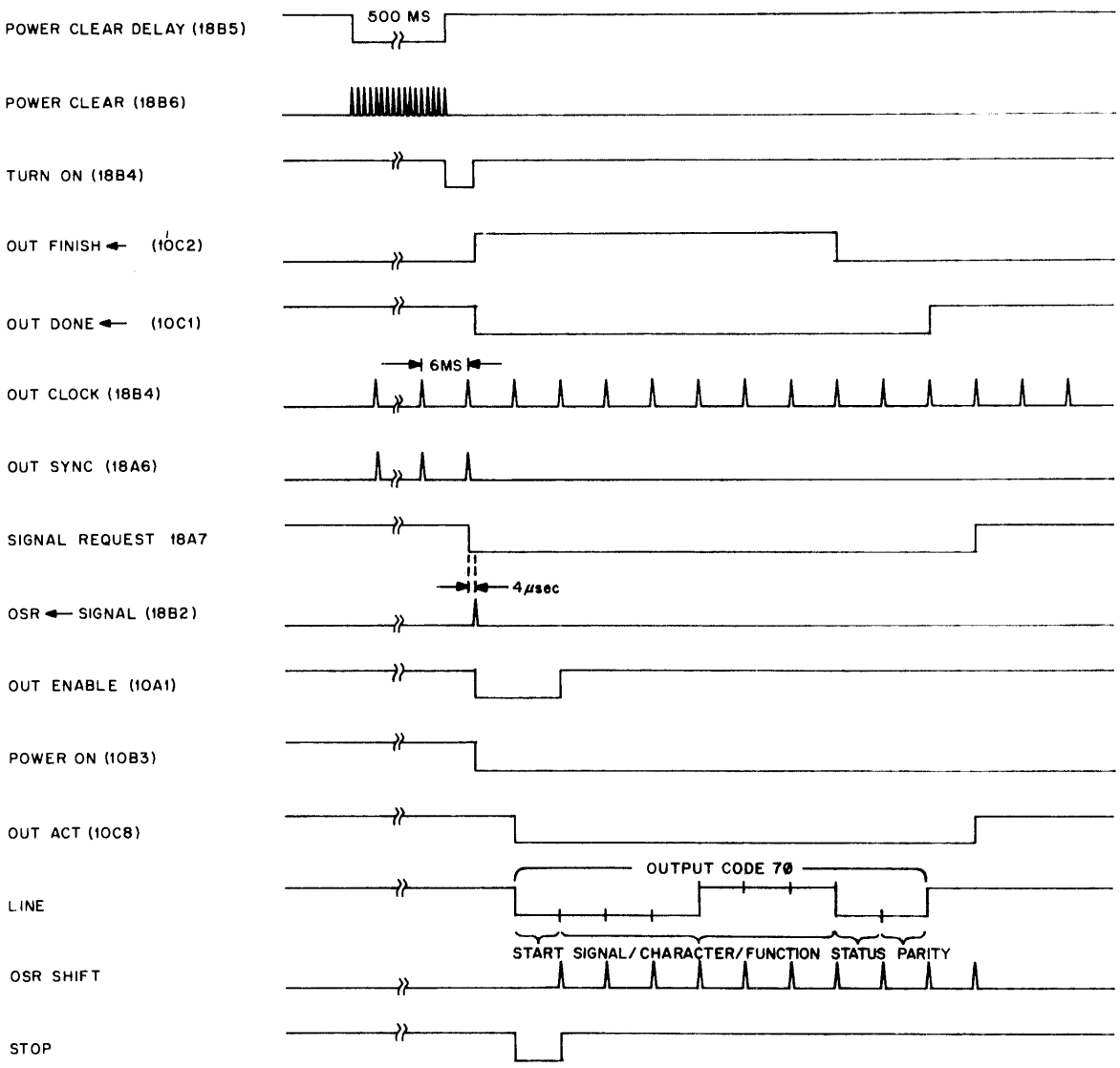


Figure 2-2 Output Timing Diagram

The OSR SIGNALS pulse also resets the TURN ON flip-flop (18:B4). The TURN ON (0) signal then sets the POWER ON flip-flop (18:B, C3) to enable the lamp driver (8:A, B4) which illuminates the POWER lamp. The POWER lamp signifies that power has been applied to the JOSS Console. The POWER ON (1) signal (18:C4) energizes the solenoid driver (TO POWER CONTACTOR) which energizes relay RY1 (refer to drawing PS-C-616-0-29). Relay RY1 holds power on until the power-off sequence is completed after the console ON/OFF switch is turned off.

The OUT ENB (1) signal enables the next OUT CLOCK pulse to set the OUT ACT (output active) flip-flop (10:C8). With the OUT ENB flip-flop set, the OUT FINISH (1) signal (10:C2) goes to 0v and the OUT DONE (0) signal goes to −3v. The OUT ACT (1) signals enables OUT CLOCK pulses to generate OSR SHIFT pulses (10:B1), which shift the contents of the OSR register. The first shift pulse resets the OUT ENB flip-flop and sets the STOP flip-flop (10:A2)l thereafter, the STOP flip-flop shifts 0s into the OSR. The initial 1 in the STOP flip-flop assures that OUT FINISH (1) remains at 0v until the 6-bit character has been shifted through the OSR.

At this point, the 70 code which has been inserted in the OSR is shifted through the LINE flip-flop and transferred to the receiving device. After the seventh shift pulse, OSR0 through OSR7 contain 0s; hence, the OUT FINISH (1) (10:C2) signal goes to −3v. The OUT FINISH (1) signal now enables the OUT PAR flip-flop to shift the parity through the LINE flip-flop (10:A, B8) and to the receiving device. The OUT PAR flip-flop is complemented for each 1 bit shifted through the OSR7 flip-flop to generate odd parity; the bits are odd for the 70 code; therefore, a 0 parity bit is generated.

After two more shift pulses, OSR5 and OSR6 contain 0s and the OUT DONE signal goes to 0v. The OUT DONE signal now enables the next OUT CLOCK pulse to reset OUT ACT (10:B8).

Since OUT ACT is reset, the OSR SHIFT pulses (10:B1) are terminated. The OUT DONE signal enables the OUT CLOCK to generate an OUT SYNC pulse, which resets the SIGNAL RQST flip-flop.

The operation described above sends the 70 code to the computer to signify that initial power has been applied to the JOSS Console. The computer, to turn on the console, now acknowledges this receipt by sending a 70 code to the console.

2.2.1.2 Computer Sends 70 Code to JOSS Console - The first bit (START) of the 70 code on the INC LINE (17:A3) is ANDed with INC CLOCK to set the INC ACT flip-flop. The INC CLOCK (18:B5) is derived from the CLK counter. The INC CLOCK is taken from the CLK3 bit; hence, its output frequency is 10560/8 or 1320 pulses per sec (refer to figures 2-4 and 2-5). The INC ACT signal generates an ISR ←(0) pulse which clears the ISR (input shift register) to prepare for the incoming signal. The INC ACT signal also enables the INC CLOCK to complement the INC CNT2 flip-flop which is the least significant digit of the 3-stage counter consisting of INC CNT2, through INC CNT0. The 3-stage counter produces an INC SHIFT pulse (17:B4) for every eight INC CLOCK pulses starting with the fourth INC CLOCK pulse following the first incoming bit.

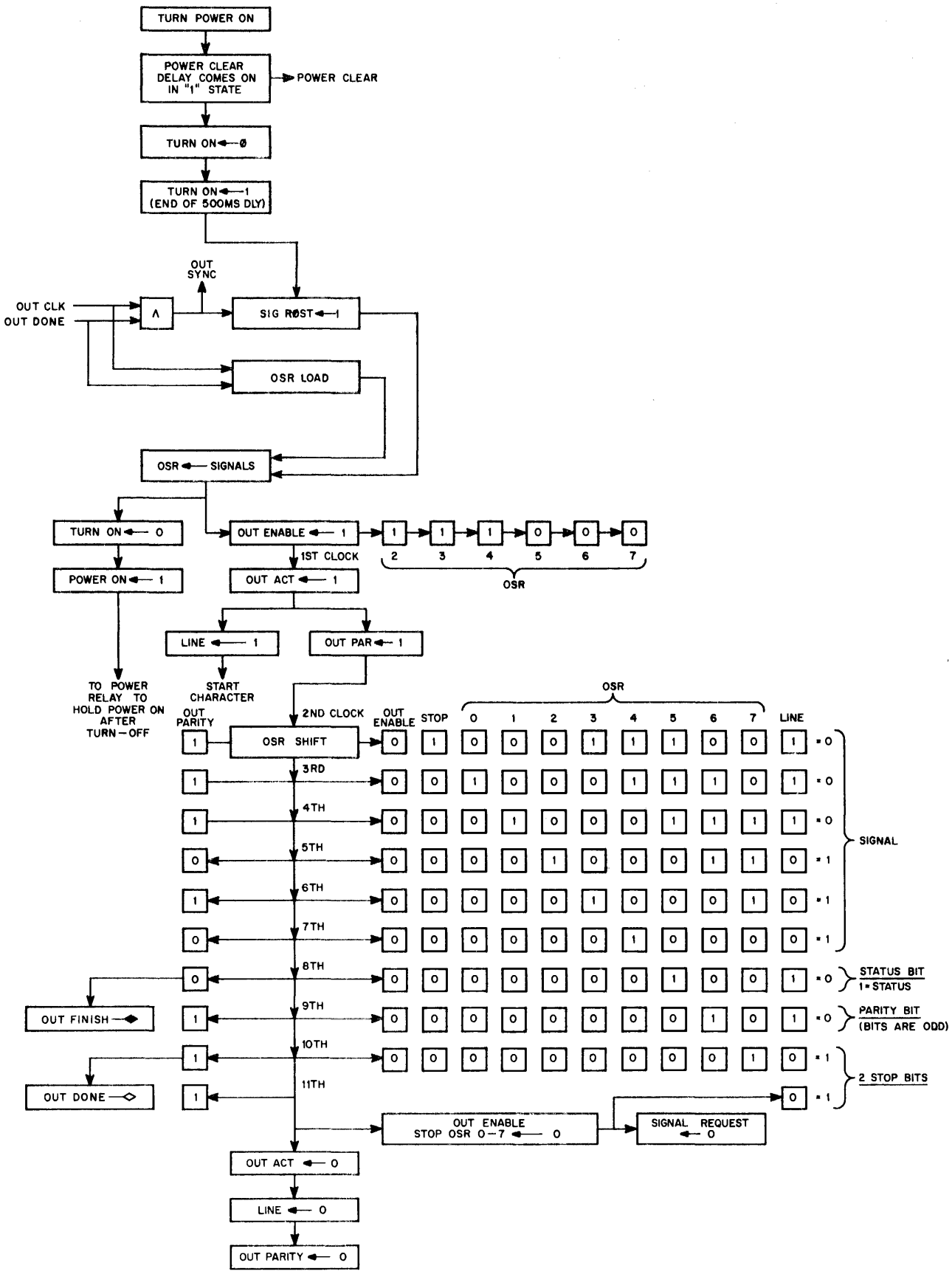


Figure 2-3 Flow Diagram, JOSS 70 Code to PDP-6

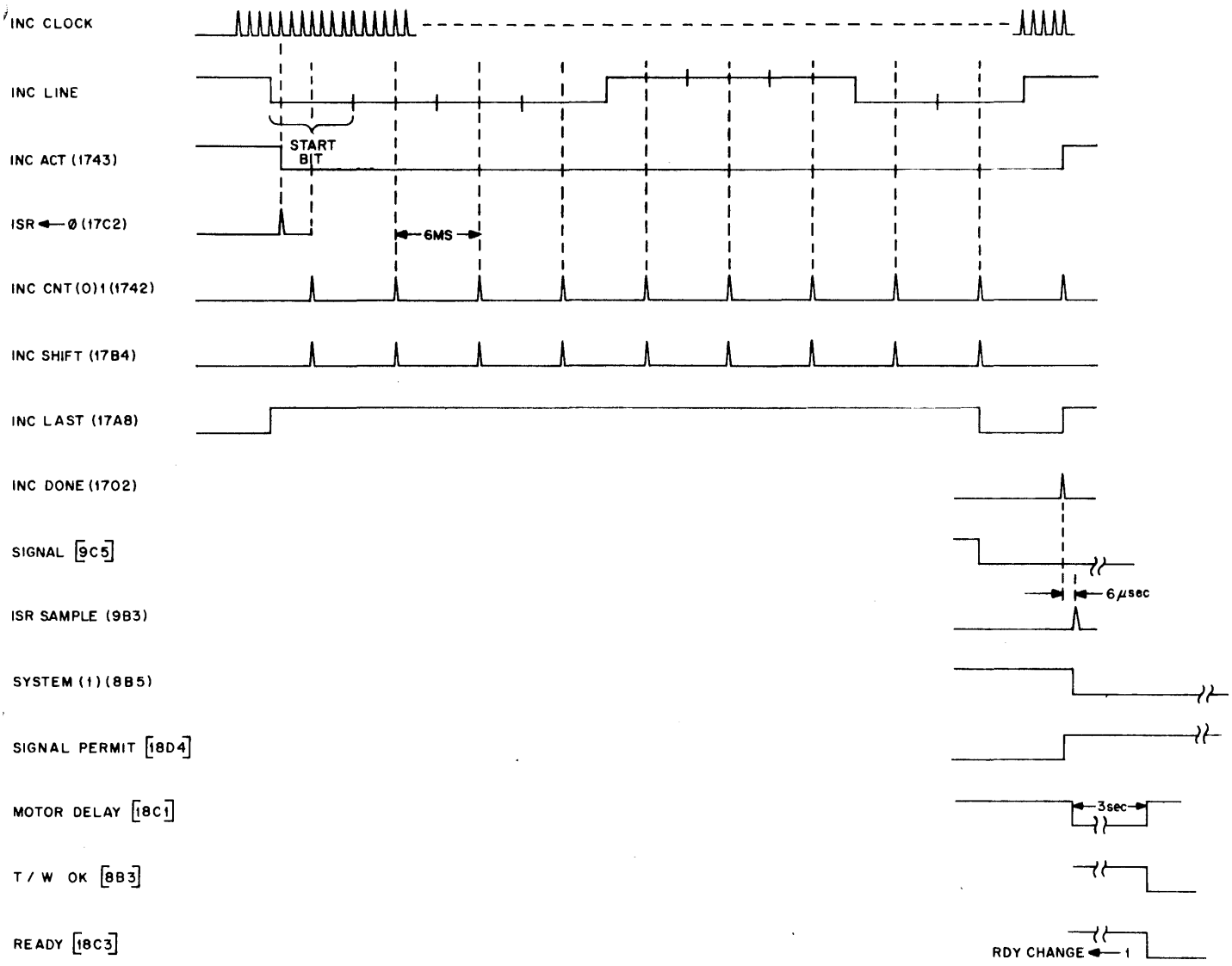


Figure 2-4 Input Timing Diagram

The INC SHIFT pulses start shifting the INC LINE bits into the ISR register. After nine shifts of the ISR register, the original start bit that initiated the incoming operation is in the INC LAST flip-flop (17:A8) and the complete word is assembled in the ISR register. The INC SHIFT pulses (17:B4) are terminated since INC LAST is set. The INC LAST signal enables the INC CNT0 (1) signal to generate an

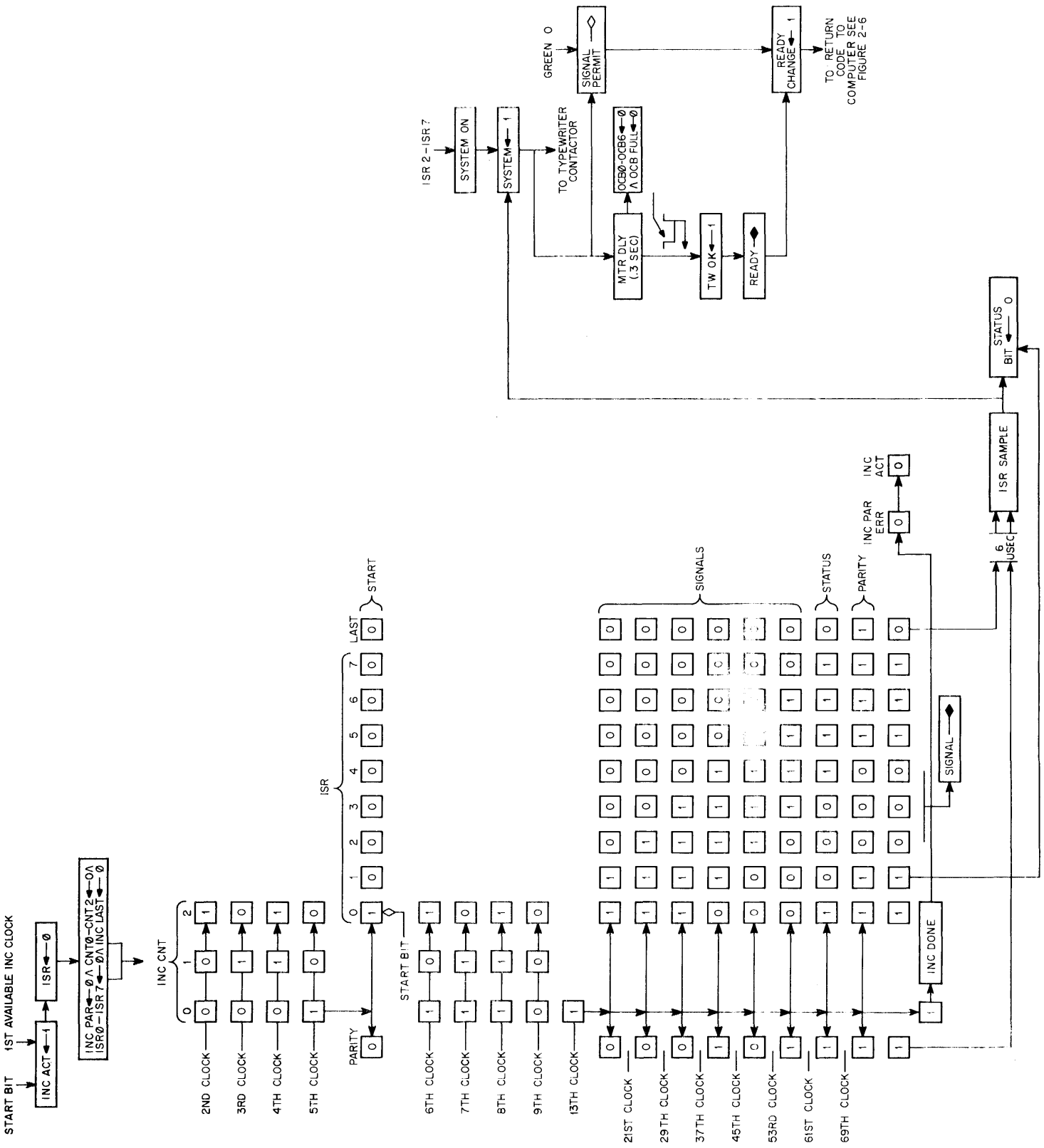


Figure 2-5 Flow, PDP-6 70 Code to JOSS

INC DONE pulse (17:D2), which resets the INC ACT to terminate the input operation by inhibiting the INC CNT counter. The input code is in the ISR register, and the next operation depends upon the ISR content; consequently, the ISR must be decoded.

The INC DONE pulse (9:C1) triggers both the 4-μsec and 6-μsec delays, or just the 6-μsec delay, depending on the contents of the ISR REGISTER. If the ISR contains a character or function (and ~ SIGNAL conditions are present), 4-μsec and 6-μsec delays are selected. At the end of the 4-μsec delay, ICB LOAD is generated and loads the ISR contents into the ICB BUFFER which previously was reset by the ICB ← (0) pulse. At the end of the 6-μsec delay, ISR SAMPLE is generated whose only purpose is to generate the ICB SAMPLE pulse 2-μsec after ICB LOAD (this 2-μsec difference assures that the logic is set up in case a function were present in the ICB). ICB SAMPLE puts the logic in not-ready condition when a long function is decoded. If the ISR contains a signal, the 4-μsec delay is disabled and the 6-μsec delay is selected. Note that ICB ← (0), ICB LOAD, and ICB SAMPLE pulses are also inhibited, preventing the contents of the ICB buffer from being changed while the logic is decoding a signal. This isolates the character and function decoding buffer from signals which enable another signal to be sent directly after a long function command without having the program wait for a READY condition. At the end of the 6-μsec delay, ISR SAMPLE is generated and decodes the signal in the ISR register. Since it is assumed that the input code is a 70, the signal (9:D6) will be a 1, denoting that a signal is being sent to the JOSS Console. The 70 code in the ISR is decoded by an R151 (9:C7) to activate the SYS ON signal. The SYS ON signal enables the ISR SAMPLE pulses to set SYS flip-flop (8:B5). The SYS flip-flop controls illumination of the SYSTEM lamp, which when lit signifies that the computer has acknowledged the initial power-on sequence and is in the process of applying ac power to the typewriter motor.

The SYS (1) signal enables the solenoid driver (8:B1) which applies power to the typewriter. The SD output energizes relay RY2 (drawing PS-C-616-0-29) to apply ac power to the typewriter motor. In addition, the SYS (1) signal triggers a 0.3-sec MOTOR-DELAY one-shot multivibrator (18:C1), which permits the typewriter motor to reach operating speed before the READY signal is sent to the computer. After the 0.3-sec interval, the MOTOR DELAY one-shot reverts to its normalized state, and the MOTOR DELAY (0) sets T/W OK (typewriter OK signal--8:B3). Since the READY came on, the JOSS Console sends the 74 code to indicate it is active.

2.2.1.3 JOSS Sends 74 Code (Typewriter OK) to Computer - The T/W OK signal is ANDed with RDY ENABLE (1) to generate the READY signal. The RDY ENABLE flip-flop (8:C5) was initially set by the PWR CLR and the absence of hold, paper out, or ready conditions. (The significance of these conditions is described later in this section.) Hence, the RDY ENABLE (1) enables the T/W OK signal to generate the READY signal.

The READY signal drives the lamp driver circuit (8:A3) that illuminates the TYPEWRITER lamp on the control panel. The READY signal transition enabled by SIGNAL PERMIT sets the RDY CHANGE (ready change) flip-flop (18:B7--refer to figure 2-6). The RDY CHANGE signal enables the next OUT SYNC pulse to set the RDY SYNC flip-flop (18:B6) and the SIGNAL RQST flip-flop (18:A7). The SIGNAL RQST signal gates on OSR LOAD (which is the OUT CLOCK AND OUT DONE delayed by 4-μsec) to generate the OSR ← SIGNALS pulse (18:B2).

The OSR ← SIGNALS pulse sets the OUT ENB flip-flop (10:A1) and inserts the 74 code into the OSR. The next OUT CLOCK pulse is enabled by OUT ENB to set the OUT ACT flip-flop (10:C8). The OSR shift pulses shift the OSR register. This operation is the same as previously described when the JOSS Console initially sent the 70 code (power-on) to the computer.

Under the described conditions, the computer is cognizant of the ready and red conditions of the JOSS Console. The computer may initiate one of two operations. It can communicate with the JOSS Console by sending character or machine functions to the typewriter, or it can send a 74 code (or 72, 74, 76, 77-status) to the JOSS Console to switch it to the green state so that the typewriter can transmit data to the computer.

2.3 COMPUTER TO JOSS CONSOLE COMMUNICATIONS (RED OPERATIONS)

During initial operations of the JOSS Console, the PWR CLR pulse resets the green flip-flop (8:B2) to put the console into the red state. In the red state, the typewriter keyboard is locked to prevent the operator from interfering with typewriter communications. The READY (0) and GREEN (0) signals are ANDed (9:B1) to drive the solenoid driver that locks the keyboard.

2.3.1 Characters

For purposes of explanation, assume that the computer is to send a character to the JOSS Console, and that the character is a small z, represented by a 23 code. The start bit of the 23-code on the INC LINE (17:C7) enables the INC CLOCK to set the INC ACT flip-flop (17:A3). The operation that shifts the 23 code into the ISR is similar to that explained previously for transferring the 70 code to the JOSS Console, with the exception that both the 4-μsec and 6-μsec delays are used. After the word is read into the ISR, the INC LAST (1) signal enables the INC CNT0 (1) pulse to generate the INC DONE pulse (17:D1,2). The INC DONE pulse generates the ICB ← (0) pulse (9:B6) to clear the ICB register (9:D2-4). The INC DONE pulse also triggers a 4-μsec delay (9:C1) which generates the ICB LOAD pulse. The ICB LOAD pulse transfers the contents of the ISR into the ICB register. The ICB SAMPLE pulse is generated but has no effect since the input code was neither a long function nor a page-go.

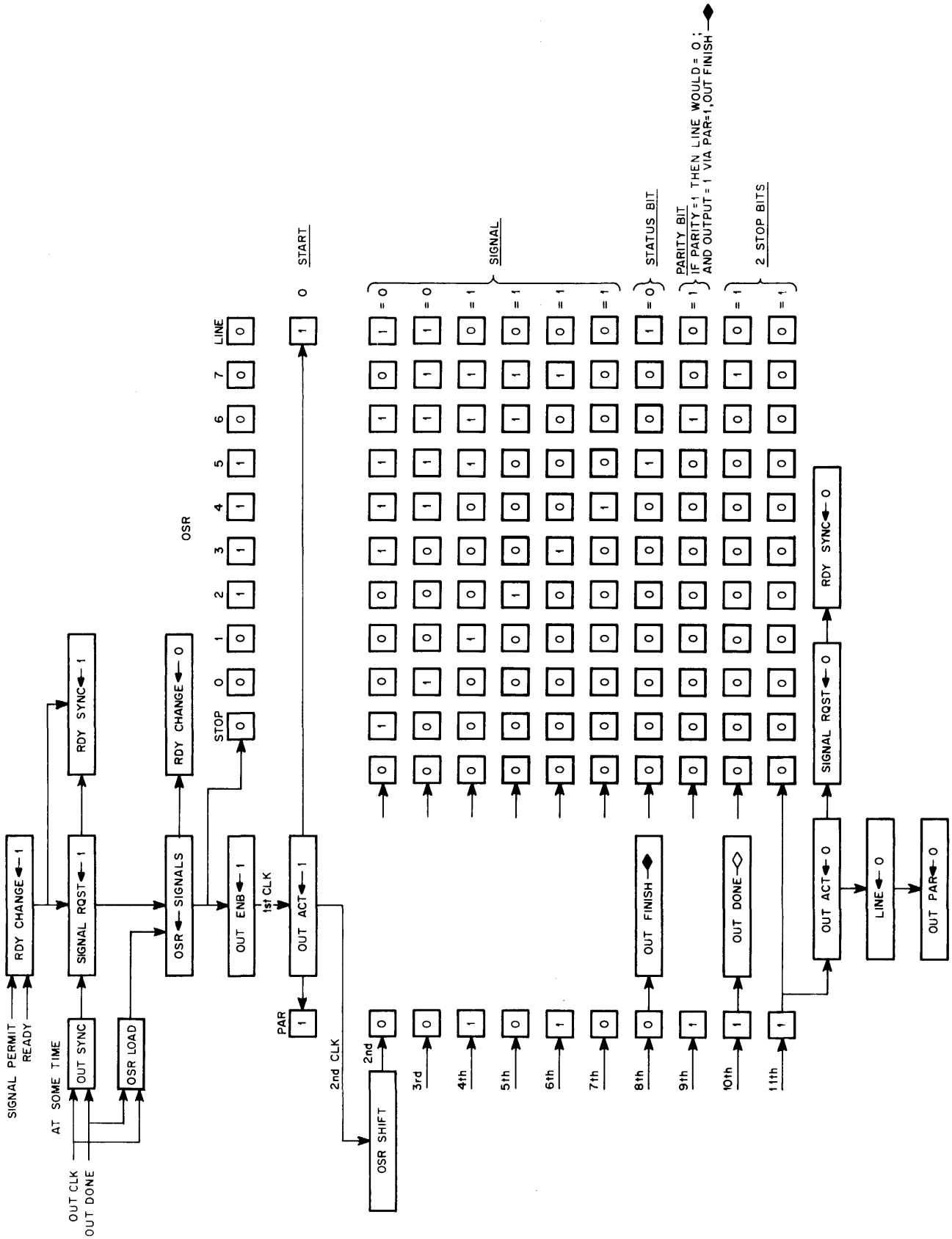


Figure 2-6 Flow, JOSS 74 Code to PDP-6

The ENERGIZE signal is ANDed with the function (ICB1(0)+ICB2(0)) and ~FEEDBACK at 9:B7 to enable one input to each of the solenoid drivers for R2A, R2, R1, R5, T1, T2, and CK. The R2A, R2, etc. designations are solenoids in the typewriter which, when energized by the proper code initiate the typing of the designated character. The JOSS codes for characters typed on the typewriter are codes 00_8 through 54_8; hence, the ICB1(0)+ICB2(0) inputs assure that the character solenoids are enabled only for characters. The additional gating with ~ FEEDBACK assures that the typewriter has completed the last function or character in the required amount of time. When the typewriter types the specified character, cam contacts C1 and C2 open after a delay. When C2 opens, it enables the Schmitt trigger (9:C1,2) to generate the FEEDBACK signal. The FEEDBACK signal resets the ENERGIZE flip-flop (9:D1) which removes the DRIVE signal from the solenoid drivers. The JOSS Console, which had been receiving the next character or machine function in the ISR register, is now ready to gate this into the ICB.

2.3.2 Machine Functions

Typewriter operational functions include lower case (LC), upper case (UC), carriage return (CR), back space (BSP), space (SP), tabulator (TAB), and page go (PG). The CR, PG, and TAB are special cases (long functions - LONG FCN) and require acknowledgment from the JOSS Console to notify the computer that the functions are complete.

As an example of the non-long functions, the back space (code 61) will be explained. The INC LINE (17:C7) initiates the input operation by enabling the INC CLOCK to set the INC ACT flip-flop. The operation of shifting the input into the ISR and then transferring the ISR contents into the ICB is the same as previously described for input operations. After the ICB is loaded, the R151 module (9:C8) decodes its contents to produce the BSP signal. The BSP and DRIVE signal are ANDed (9:A3) to enable the BSP solenoid driver, which energizes the BSP solenoid in the typewriter. The C5 cam contact in the typewriter opens after a delay permitting the C2 through C6 functions (9:C1) to generate the FEEDBACK signal. The FEEDBACK signal resets the ENERGIZE flip-flop to remove the drive source for the BSP solenoid driver. The JOSS Console which has been receiving the next character or function in the ISR register, is now ready to gate this into the ICB.

When a long machine function occurs, it is decoded by the R151 module to energize its respective solenoid driver as explained for the BSP machine function. However, the CR, PG, or TAB signal generates the LONG FCN signal (9:C4) which resets the T/W OK flip-flop (8:B3). The READY signal (18:C3) then becomes inhibited. The computer is cognizant of the not-ready condition and waits for a typewriter OK signal (74 code).

When the typewriter has finished the long machine function, the INTLK signal (11:B8) sets the T/W OK flip-flop (8:B3). The T/W OK and the RDY ENABLE (1) signal are ANDed to generate the READY signal (18:C3).

The READY signal is enabled by the SIGNAL PERMIT signal to set the RDY CHANGE flip-flop which in turn enables the OUT SYNC pulse to set the RDY SYNC flip-flop. The RDY CHANGE (1) signal also enables the OUT SYNC pulse to set the SIGNAL RQST flip-flop. This initiates the output transfer operation previously explained to transfer the 74 code to the computer.

2.3.3 Signals

The signals sent from the computer to the JOSS Console are SYS ON (system on), SYS OFF, IN ON (interrupt on), IN OFF, SG (system green), SR (system red), BEEP (energizes the audible alarm), and NOP (no operation). Operation of the computer sending signals to the JOSS Console was explained previously in section 2.2.1.2. The operation of the other signals is similar and therefore will not be explained. Note that the computer can send signals to the JOSS Console regardless of the status of the console.

2.3.4 Computer Request Status

The computer requests the status of the JOSS Console by inserting a 1 into the status bit position of the word sent to the JOSS Console. After the computer has loaded the ISR, if status bit ISR1 is 0v, the ISR1(0) signal enables the ISR SAMPLE pulse to set the STATUS BIT flip-flop (18:A1). The STATUS BIT (1) signal enables the OUT SYNC pulse to set the STATUS RQST flip-flop (18:A8). The STATUS RQST signal enables the 4-μsec-delayed OUT CLOCK pulse (OSR LOAD) to generate the STATUS SEND pulse (18:B3). The STATUS SEND pulse is delayed 1 μsec to generate the OSR STATUS pulse (10:B,C4). The OSR STATUS pulse sets the OUT ENB flip-flop (10:A1) and loads the status code into the OSR. Note that a single code is not allocated for a single status; rather, the status of each individual bit of OSR3 through OSR7 specifies the status of INC PAR ERR, GREEN/RED, READY, INT ON/OFF, and SYS ON/OFF, respectively. Note also that STATUS SEND (10:C3) sets OSR1 through OSR7 so that the OSR STATUS pulse can insert the proper code into the OSR register. Once OSR is loaded and the OUT ENB flip-flop is set, the operation that follows is similar to previously explained output operations.

2.3.5 Parity Error

When a parity error occurs during an input transfer operation, the INC PARITY flip-flop (17:A1) is in the 0 state when INC DONE occurs. INC PARITY (0) signal enables the INC DONE to set the INC PARITY ERROR flip-flop (17:A1). The INC PARITY (1) signal inhibits the ICB LOAD and ISR SAMPLE pulse (9:B, C1); consequently, the ISR register is disabled and input operation stops at this point. The INC PARITY signal also enables the OUT SYNC pulse to set the STATUS RQST flip-flop (18:A8) which

initiates a status transfer to the computer. Since parity is part of the status message, the computer knows that a parity error occurred on the last transfer. The computer can then take the necessary corrective action. The OSR STATUS pulse resets the INC PARITY ERROR flip-flop.

2.3.6 Interrupts

While the system is in red operation, the operator at the JOSS Console can initiate an interrupt by activating the INTERRUPT switch on the control panel. When depressed, the INTERRUPT button generates the INT SW signal (18:D8) which sets the INT KEY flip-flop (18:B8). The INT KEY signal enables the OUT SYNC pulse to set the INT SYNC and the SIGNAL RQST flip-flops. Setting the SIGNAL RQST initiates a signal transfer to the computer. The interrupt code (72) is inserted into the OSR, and the OSR contents are then transferred serially as explained previously. Note that the operator can initiate and transfer an interrupt code to the flip-flop regardless of the condition of the INT ON/OFF flip-flop. This is not a logical interrupt as recognized by the computer. It is an interrupt which is recognized by the software.

2.4 JOSS CONSOLE TO COMPUTER COMMUNICATIONS (GREEN OPERATIONS)

When GREEN flip-flop is in the set state, the operator at the JOSS Console can type characters or execute machine functions. The characters or machine functions are encoded and transferred serially to the computer.

2.4.1 Characters

When the operator types a character on the typewriter, the particular character code is shown in the selective closing of contacts R2A, R2, R1, R5, T1, and T2 in the typewriter. These contacts close and through switch filters enable gates which insert the character codes into the OCB (output control buffer) register (11:A).

The C1 cam in the typewriter closes to signify that a character has been typed. This generates the C1 CAM signal (11:A1). The C1 CAM signal is enabled by OCB FULL (0) and GREEN (1) to generate the OCB ←CHAR pulse, which strobes the input character (generated by R2A, R2, etc.) into the OCB register. The OCB ←CHAR pulse also sets the OCB FULL flip-flop (11:A8), which signifies that a character (or machine function) is in the OCB register and is ready to be sent to the computer.

The OCB FULL signal enables the OUT SYNC pulse to set the CHAR RQST (character request) flip-flop (18:A6). The CHAR RQST signal enables the 4-μsec-delayed OUT CLOCK to generate the OSR ←CHAR pulse (18:B2) which inserts the OCB contents into the OSR register (10:A). The OSR ←CHAR

pulse also resets the OCB to make it ready for the next input and sets the OUT ENB flip-flop (10:A1) to initiate the transfer of the OSR contents to the computer. The operation of transferring the OSR contents was explained previously.

2.4.2 Functions

For this example, assume that the space function is executed on the typewriter. When the space key is depressed, the contact closure applies +10v to the SF (switch filter, 11:D7) which provides a 0v SP KEY signal. The SP KEY signal generates the FCN KEY signal (11:D1), which in turn generates the FCN ENB (provided of course that the OCB is not full). The opening of the C5 cam in the C2-C6 chain produces the FEEDBACK signal (9:C3) which when enabled by ENERGIZED (0), PAGE GO (0), triggers a 10-msec delay (11:C2) to generate the OCB ← FCNS signal. The ENERGIZED (0) input inhibits OCB ← FCNS when the computer is typing a character on the typewriter, and the PAGE GO (0) signal inhibits the OCB ← FCNS during page-go operation.

After the 10-msec delay, the OCB FCNS gates the SP KEY code (code 63) into the OCB ← register. Not that the OCB FCNS signal unconditionally sets OCB0 and OCB1 and that OCB2 remains reset; this is because all function codes start with octal 6 as the most significant digit. (In a similar manner the LC KEY, UC KEY, BS KEY, and TAB KEY insert their codes into the OCB register.) The OCB ← FCNS pulse also sets the OCB FULL flip-flop, and the OCB FULL signal enables the OUT SYNC pulse to set the CHAR RQST flip-flop (18:A6) and this initiates the transfer to the computer. Note that the LC, UC, SP, BS, and TAB functions are all similar and the PG GO and CR (long functions) differ somewhat.

When the CR key (11:D4) is depressed, it energizes a Schmitt trigger to produce a CR KEY signal. The CR KEY signal resets the GREEN flip-flop (18:B2) to put the system into a red status. The CR KEY also sets the CR HOLD flip-flop (11:D3). The CR HOLD (1) signal enables the next OUT CLOCK to trigger the 2-μsec delay (11:C4). The 2-μsec output pulse is enabled by OCB FULL (0) to generate the OCB FCNS pulse, which inserts the carriage-return code into the OCB register and sets the OCB FULL signal to set CHAR RQST (18:A6) and initiate a character transfer. The OCB FCNS also resets the CR HOLD flip-flop.

When the operator depresses the PAGE key, a Schmitt trigger (11:D5) generates the PAGE KEY signal. The PAGE KEY signal triggers the 2-μsec one-shot multivibrator (11:C2) generating PG KEY DLY, which in turn generates the OCB ← FCNS. The PG KEY DLY also resets the GREEN flip-flop, putting the system into the red state. The OCB ← FCNS signal inserts the PAGE KEY code into the OCB and sets the OCB FULL flip-flop. OCB FULL enables the OUT SYNC pulse to se the CHAR RQST flip-flop and thus initiate the transfer.

The PG KEY DLY sets the PAGE GO flip-flop (9:B2) to drive the PG solenoid driver, which in turn energizes the PG solenoid to advance the paper in the typewriter. When the top of the next page is reached, the typewriter initiates a carriage return and the PAGE GO flip-flop is reset by the ensuing FEEDBACK signal.

2.5 CR HOLD FLIP-FLOP

The logic treats the CARR RTN (CR) function differently from all other characters or functions. The CR function sets a flip-flop to initiate its code. When a CR is initiated from the keyboard, CR HOLD is set, disabling all other function codes, and on the next available OUT CLK pulse allows the CR code to be read into the OCB buffers, thus disabling all character codes.

Since CR code always places the console into RED operation and signifies that an end-of-line transmission has occurred, no other information should be transmitted after this. Since CR KEY sets CR HOLD and prevents anything else from happening, a carr rtn takes precedent over a character or function struck simultaneously with it. This insures that the computer is aware of the change of state from green to red.

2.6 TYPEWRITER HOLD

When the operator wishes to put the typewriter in a not-ready condition, he depresses the typewriter OFF switch. The typewriter ON/OFF switch is labeled S1 on drawing SD-D-616-0-30; S1 has two sets of contacts. One set of contacts applies ac off-line power to the typewriter motor. The other set closes the HOLD contacts (8:D4), or READY (8:D5), which enables the SF (switch filter) to reset, or set, the RDY ENB flip-flop (8:C5). With RDY ENABLE reset, the READY signal is inhibited. The RDY ENABLE (0) generates a positive pulse (18:C6) which sets the RDY CHANGE flip-flop. The RDY CHANGE signal enables the next OUT SYNC pulse to insert the not-ready 75 code into the OSR and set the SIGNAL RQST flip-flop. The SIGNAL RQST initiates the transfer of the 75 code to the computer so that it is cognizant of the JOSS Console not-ready condition.

When the operator pushes the typewriter ON/OFF switch to the ON position, the typewriter is put back into ready operation (providing T/W OK is a 1). This sends a READY signal to the computer so that it knows of the ready condition. As the ON switch is activated, the C1 contacts of S1 generate a momentary 10v READY signal (8:D5), which sets the RDY ENABLE flip-flop. The RDY ENABLE signal generates READY (18:C3) which intitates a signal transfer to the computer to transfer the 74 (ready) code to the computer.

2.7 POWER-OFF SEQUENCE

When the POWER ON/POWER OFF switch on the control box is switched to the POWER OFF position, the POWER OFF contacts (8:A6) trigger the Schmitt trigger to generate the POWER DOWN signal. The POWER DOWN signal triggers the POWER OFF DLY (500 msec) one-shot multivibrator (8:C8) and sets the PWR OFF flip-flop, which generates the TURN OFF signal (8:B8). The TURN OFF signal enables the next OUT SYNC pulse to set the SIGNAL RQST flip-flop, which in turn enables the 4-μsec-delayed OUT CLOCK (OSR LOAD) to generate the OSR SIGNAL pulse (18:B2). The OSR SIGNAL pulse loads the OSR register with the 71 code (turn off) and sets the OUT ENB flip-flop (10:A1) to initiate the 71-code transfer to the computer. The OSR SIGNAL pulse (8:C2), enabled by TURN OFF, resets the SYS (system) flip-flop; this inhibits the SIGNAL PERMIT (18:D4) which prevents any further signal transfers, removes power from the typewriter (8:D2), and resets PWR OFF flip-flop, which terminates the TURN OFF signal so that it does not set the SIGNAL RQST flip-flop on the next OUT SYNC pulse.

After the 500 msec PWR OFF DLY, the PWR OFF DLY (0) signal resets the POWER ON flip-flop (18:B, C3) and triggers the PWR CLR DLY one-shot multivibrator (18:B, C5). The POWER ON (1) signal goes to 0v and disables SD (18:C4), which de-energizes RY1 (drawing PS-C-616-0-29) to remove power from the console. However, before console power is completely off, the PWR CLR DLY signal enables the 10-kc clock (18:B, C5) to generate PWR CLR pulses, which prevent any system operation while shutting off the console.

2.8 CHAR RQST/SIGNAL RQST/STATUS RQST

These three flip-flops are set whenever output action is initiated. However, they are gated with OSR LOAD so that a priority order is set up between them. This priority is set so that a CHAR RQST always is acknowledged first, a SIGNAL RQST second, and a STATUS RQST third.

CHAPTER 3
OPERATION

3.1 INTRODUCTION

Operation of the JOSS Console is controlled by the control box shown in figure 3-1. The controls and indicators of the JOSS Console control box are explained in table 3-1.

Figure 3-1 JOSS Console and Indicators

TABLE 3-1 OPERATING CONTROLS

Control or Indicator	Function
POWER ON/POWER OFF	Applies power to the JOSS Console/Removes power from the JOSS Console.
CONSOLE POWER	Indicates that the POWER ON control has successfully applied power to the JOSS Console.
JOSS SYSTEM	Indicates that the computer has acknowledged the TURN-ON signal; enables console functions, including typewriter.
TYPEWRITER	Indicates that on-line power has been applied to the typewriter.
Red lamp	Indicates that computer has control of the JOSS Console.
Green lamp	Indicates that user has control of the JOSS Console..
INTERRUPT	When illuminated, indicates that the software has acknowledged the interrupt. Indicates sending an interrupt signal to the computer. (red mode only)

3.2 OPERATING PROCEDURES

Operating procedures for the JOSS Console are explained in narrative form as follows: Depress the POWER ON button. This applies power to the JOSS Console and also sends a signal to the computer that indicates power application. If power has been successfully applied to the JOSS Console, the CONSOLE POWER lamp illuminates. If the computer successfully receives the power-on signal and is ready to communicate with the JOSS Console, the computer sends a system-on signal to the JOSS Console. The system-on signal illuminates the JOSS SYSTEM lamp. At the same time, the system-on signal applied to the typewriter initiates a short delay to permit the typewriter motor to reach operating speed. After the delay, the TYPEWRITER lamp lights to signify that the typewriter is operational. A signal goes to the computer which indicates the typewriter ready condition.

The JOSS Console is fully operational only when the JOSS SYSTEM, CONSOLE POWER, and the TYPEWRITER lamps are illuminated. Application of power places the JOSS Console in red mode of operation (computer has control); the typewriter keyboard is locked to the operator to prevent communication from the JOSS Console to the computer. The computer communicates to the JOSS Console by sending encoded signals which, when decoded, type or execute the specified machine function on the typewriter.

In addition to sending typewriter characters or functions, the computer can send signals to the JOSS Console to perform certain functions. These signals and their functions are explained in table 3-2.

TABLE 3-2 COMPUTER SIGNALS

Signals	Function
System on	Turns on "JOSS system" to notify operator that the computer is ready to communicate.
System off	Turns off "JOSS system" to notify operator that the computer is not ready to communicate.
Interrupt on	Illuminates the INTERRUPT lamp to indicate that the computer has acknowledged the interrupt.
Interrupt off	Turns off the INTERRUPT lamp.
System green	Illuminates the green lamp to indicate that the user has control of the JOSS Console. Each time the JOSS Console switches from red to green a short, soft beep occurs.
System red	Illuminates the red lamp to indicate that the computer has control of the JOSS Console.
Beep	Sounds the audible alarm. This is a long, loud beep, distinguishable from switching to green beep.
NOP	No operation (usually sent with STATUS REQUEST bit).

In the red mode (red lamp lit), the INTERRUPT button may be depressed to initiate a software interrupt. The computer interrupt must be enabled in the computer before the JOSS Console interrupt is processed. The programmer must make sure that the INTERRUPT lamp on the control box is turned on or off by the computer to signify the condition of the computer interrupt.

When the computer puts the console into green mode (illuminates the green lamp), the user has control of the JOSS Console. The keyboard is unlocked and the operator may communicate by typing on the keyboard.

The characters typed or the machine functions executed are encoded and sent to the computer. When the carriage-return or page-go machine functions are executed, the JOSS Console reverts to the red mode. The operator must wait for the computer to switch the JOSS Console back to green mode before resuming typing.

With console power removed the typewriter may be used in an off-line mode by depressing the typewriter ON control, which applies ac off-line power. If the typewriter OFF control is depressed, ac off-line power is removed. The only requirement for off-line use is that the rear-panel circuit breaker be on and the external ac source be turned on. While in off-line mode, if console power is applied, the typewriter is turned off (to await SYS ON).

CHAPTER 4
MAINTENANCE

4.1 MARGINAL CHECK FACILITY

Maintenance procedures for the JOSS Console consist of normal shop practices performed for normal digital circuits. In addition, the JOSS Console incorporates a marginal check facility. The marginal check facility permits the normal +10v or −15v, which is applied to the modules, to be varied over a small range. This aggravates borderline conditions within the logic to reveal observable faults. These conditions can be corrected during scheduled preventive maintenance to forestall possible future equipment failures. The check can also be used as a troubleshooting aid to locate marginal or intermittent components, such as deteriorating transistors. The checks are performed by operating the equipment logic circuits from an external, adjustable power source, such as the DEC Type 734 Variable Power Supply.

Marginal check terminals are provided on color-coded connectors which are connected in common to all racks. An external power supply can be connected to the 8-pin Jones plug (figure 4-1) which connects the external power supply to the marginal check terminals in each rack. The color coding of the terminals from top to bottom is as follows:

Orange, +10 vdc marginal-check supply

Red, +10 vdc internal supply

Black, ground

Blue, −15 vdc internal supply

Green, −15 vdc marginal-check supply

Pin 1 of the Jones plug connects to the orange terminal, pin 3 to the green terminal, and pin 7 to the black terminal. Two single-pole single-throw switches at the end of each rack of logic allow selection of either the normal internal power supply or the external marginal-check power supply for distribution to the logic. The top switch selects the +10v supply routed to all modules in that rack. In the down position the fixed internal +10v supply connected to the red terminal is supplied to the modules, and in the up position the marginal-check voltage supplied to the orange terminal is supplied to all modules in that rack. The bottom switch selects the −15v supply to be routed to all modules. In the down position the fixed −15v output of the internal power supply, received at the blue terminal, is supplied to the modules; while in the up position, the marginal-check voltage, connected to the green terminal, is supplied to all modules.

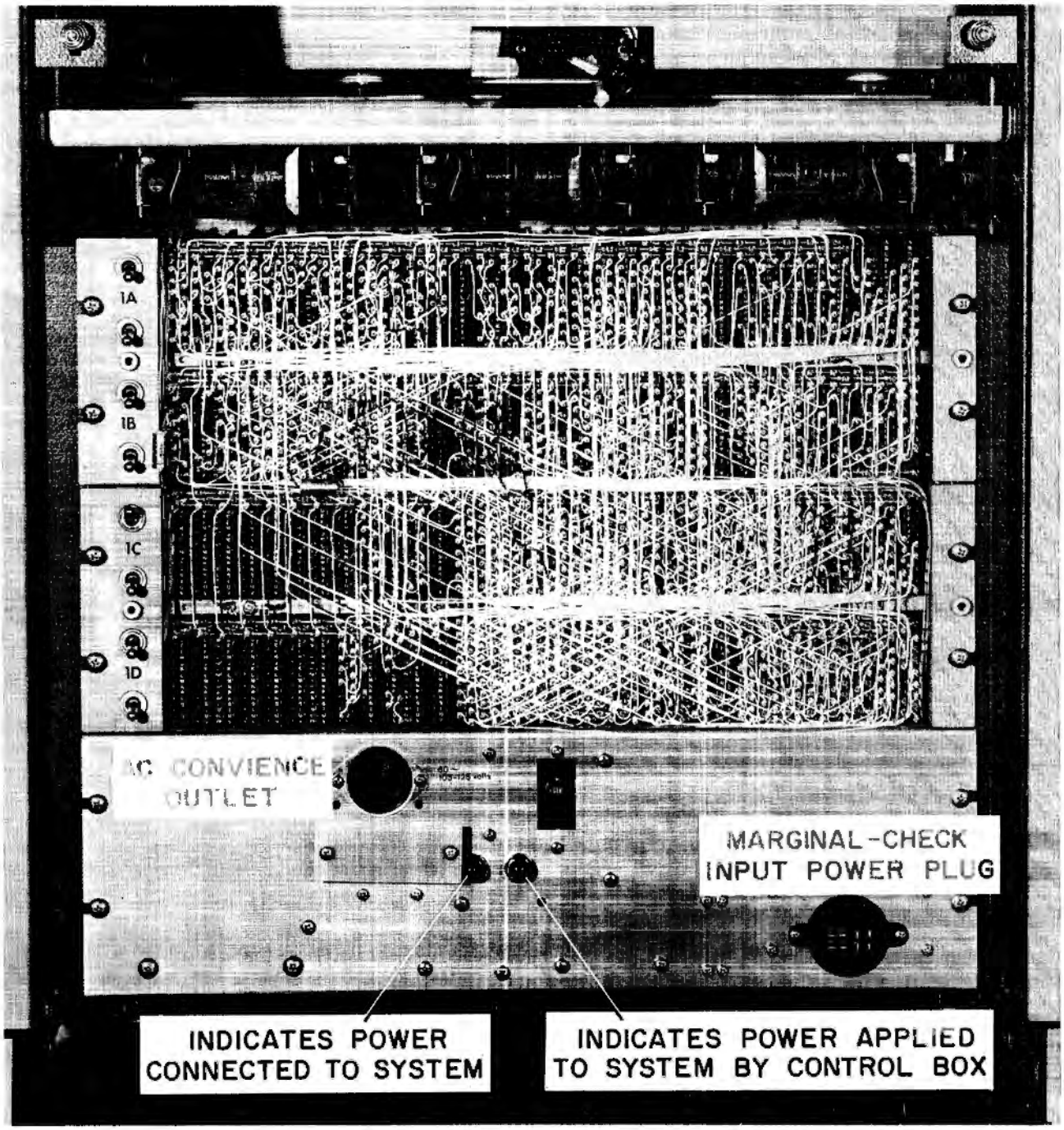


Figure 4-1 Power Indicators and Marginal Check Facilities

4.1.1 Marginal Check Procedure

To perform marginal checks, proceed as follows:

a. Connect the external power supply to the Jones plug with the variable +10v connected to pin 1, ground connected to pin 7, and the variable −15v connected to pin 3.

b. Energize the morginal-check power supply and adjust the +10v output to a nominal +10 vdc.

c. Set the top switch on the rack to be checked to the up position.

d. Start equipment operation in a routine which fully utilizes the circuits in the rack to be tested.

e. Lower the +10v marginal-check power supply until normal system operation is interrupted. Record the marginal-check voltage. At this point marginal modules may be replaced.

f. Start equipment operation. Then increase the +10v marginal-check supply until normal operation is interrupted, at which point record the marginal-check voltage. Marginal modules can again be located and replaced.

g. Stop operation and return the top switch to the down position.

h. Repeat steps b through g for the bottom switch on the logic rack being checked.

i. Repeat steps b through h for each rack or logic to be checked.

j. De-energize and/or disconnect the external marginal-check power supply.

CHAPTER 5
INSTALLATION AND INTERFACE

5.1 INTRODUCTION

The JOSS Console is a self-contained unit and the only installation requirements are input/output signal connections to the 10-pin Cinch-Jones plug (figure 5-1) and the external ac power connection. A normal 115-vac, 60-cps output can supply the ac input power for the JOSS Console. The input/output Cinch-Jones plug can accommodate three inputs and two outputs for Data Phone communication, or one TWP input and one TWP output for DEC 630 Data Communication Channel. Pin 2 and pin 10 of the external socket should be jumpered when connection is to DEC 630. Pins and uses are listed below:

Pin	Use
1	Output GND (630)--PROT. GND (D.P.)
2	Received Data (D.P.)
3	Data Terminal Ready (D.P.)
4	Output Data (630)
5	Clear to Send (D.P.)
6	Input Data (630)
7	Data Set Ready (D.P.)
8	Transmit Data (D.P.)
9	Input GND (630)--SIG. GND (D.P.)
10	GND

5.2 630 INTERFACE

Output signals from the console to the 630 are logic levels of ground and −3v and are transmitted via a single twisted pair (telephone pair) wire. These signals are driven by a R650 Bus Driver which is designed to avoid ringing on exceptionally long lines.

Input signals to the console from the 630 are logic levels of ground and −3v and are received via a single twisted pair (telephone pair) wire. These signals go directly into the input of a gatable inverter (R111) with the input SIGNAL line terminated. This input SIGNAL line terminator accomplishes two things: first, it damps out ringing on the line; second, it keeps the R111 output from turning on when the INPUT/OUTPUT connector is open or connected to other than a 630.

On input or output operations the first bit of any code is called the start bit and must be at −3v to initiate action.

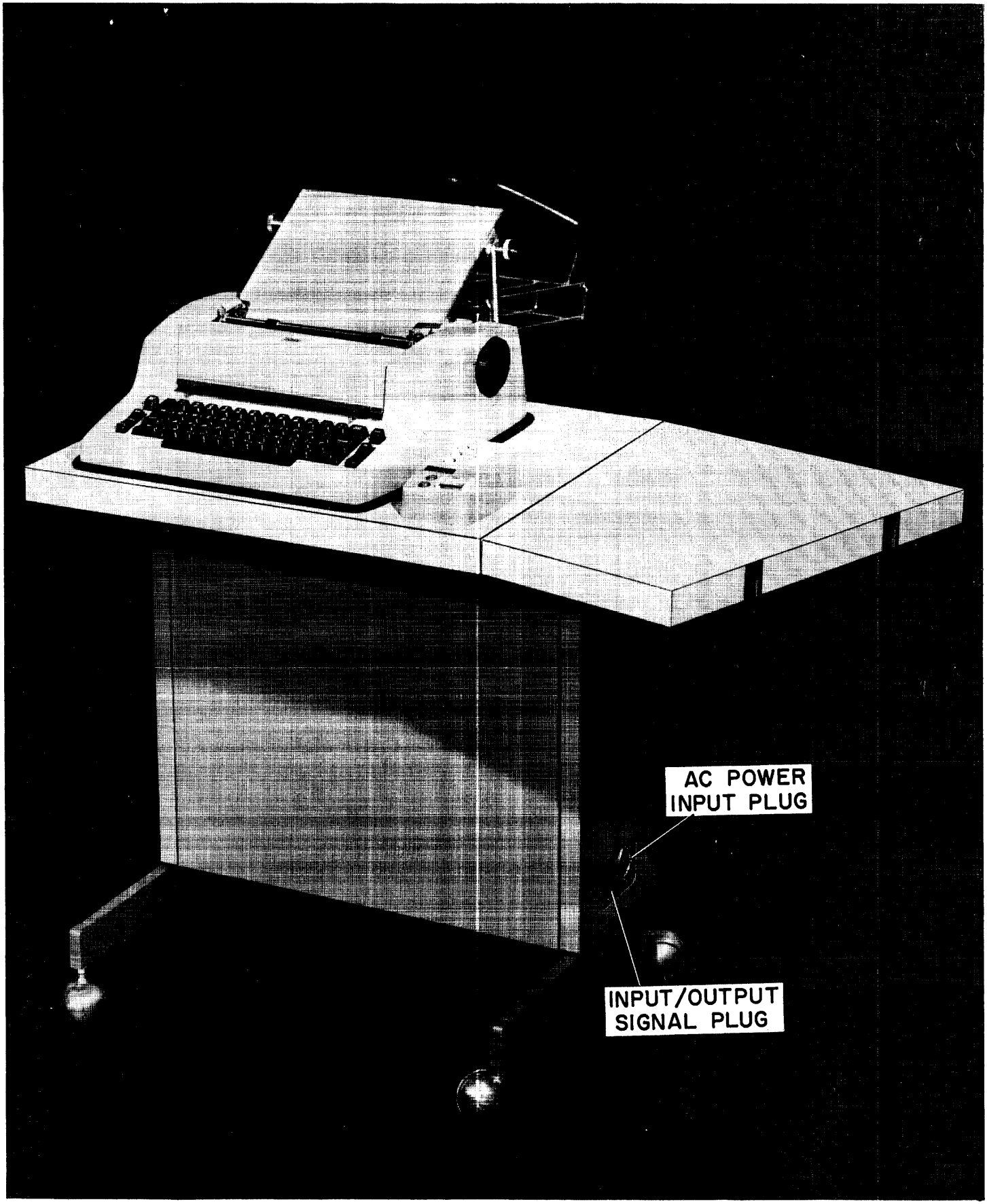


Figure 5-1 Power and I/O Signal Connections

5.3 DATA PHONE INTERFACE

Output signal from the console to the data phone are logic levels of +6v and −6v with all signals to the data phone transmitted on a single wire. Logic levels of ground and −3v go to W602 Level Converters which convert ground inputs to −6v and −3v inputs to +6v. Two signals are required for output operation to the data phone terminal:

TRANSMIT DATA	This is the actual line used to transmit output signal codes.
READY	This level is generated when power has been applied to the console and allows the data phone to enter and remain in the data mode.

Input signals to the console from the data phone are logic levels of +6v and −6v with all signals from the data phone received on a single wire. Logic levels of +6v and −6v go to W510 Level Converters which convert +6v inputs to −3v and −6v inputs to ground. Three signals are required for input operation to the console:

DATA SET READY	This indicates that data could be sent to the console.
RECEIVED DATA	This is the actual line used to receive input signal codes.
CLEAR TO SEND	This line causes DLY GO to change from ground to −3v when the data phone is ready to transmit or receive data. DLY GO going from ground to −3v allows the PWR CLR DLY to time out (500 msec) and thus start output and input operations. Should the data phone not be ready, DLY GO would remain at ground and thus constantly enable PWR CLR DLY.

CHAPTER 6
ENGINEERING DRAWINGS

6.1 INTRODUCTION

This chapter contains reduced copies of DEC block schematics, circuit schematics, and other engineering drawings necessary for understanding and maintaining this equipment. Only those drawings which are essential and are not available in the referenced pertinent documents are included.

6.2 DRAWING NUMBERS

DEC engineering drawing numbers contain five groups of information, separated by hyphens. A drawing number such as BS-D-9999-1-5 consists of the following information reading from left to right: a 2- or 3-letter code specifying the type of drawing (BS); a 1-letter code specifying the original size of the drawing (D); the type number of the equipment (9999); the manufacturing series of the equipment (1); and the drawing number within a particular series (5). The drawing type codes are:

BS, block schematic or logic diagram
CS, circuit schematic
RS, replacement schematic
SD, system diagram
UML, utilization module list
WD, wiring diagram

6.3 CIRCUIT SYMBOLS

The block schematics of DEC equipment are multipurpose drawings that combine signal flow, logical function, circuit type and physical location, wiring, and other pertinent information. Individual circuits are shown in block or semiblock form, using special symbols that define the circuit operation. These symbols are similar to those appearing in both the FLIP CHIP Module Catalog and the System Module Catalog, but are often simplified.

6.4 LOGIC SIGNAL SYMBOLS

DEC standard logic signal symbols are shown at the input of most circuits to specify the enabling conditions required to produce a desired output. These symbols represent either standard DEC logic levels, standard DEC pulses, standard FLIP CHIP pulses, or level transitions.

6.4.1 Logic Levels

The standard DEC logic level is either at ground (0 to −0.3v) or at −3v (−2.5 to −3.5v). Logic signals generally have mnemonic names which indicate the condition represented by assertion of

the signal. An open diamond (——◇) indicates that the signal is a DEC logic level and that ground represents assertion; a solid diamond (——◆) indicates that the signal is also a DEC logic level and that −3v represents assertion. All logic signals applied to the conditioning level inputs of capacitor-diode gates of diode-capacitor-diode gates must be present for a specified length of time (depending on the module used) before an input pulse will trigger operation of the gate.

6.4.2 FLIP CHIP Standard Pulses

FLIP CHIP circuit operation uses two types of pulses, R series and B series. The pulse produced by R-series modules starts at −3v, goes to ground (−0.2v) for 100 nsec, then returns to −3v. This pulse is shown in figure 6-1.

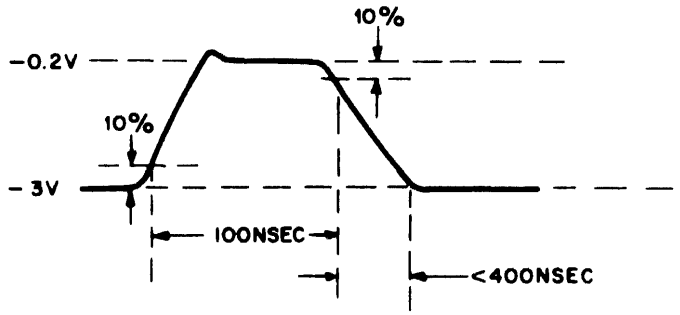


Figure 6-1 FLIP CHIP R-Series Pulse

The B-series negative pulse is 2.5v in amplitude and 40 nsec in duration and is shown in figure 6-2. If this pulse is applied to the base of an inverter, the inverter output will be a narrow pulse, similar in shape to the R-series standard pulse. The B-series positive pulse, which goes from ground to +2.5v, is the inverse of the B-series negative pulse.

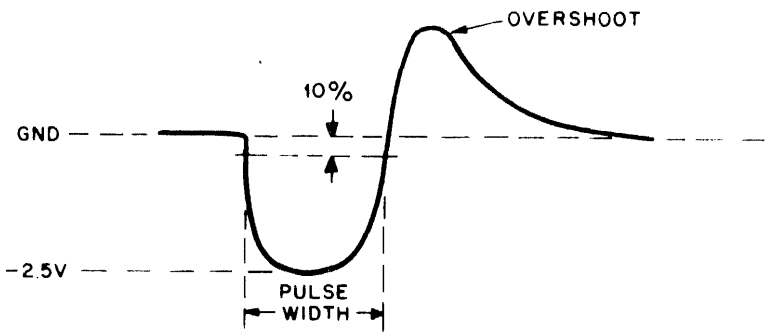


Figure 6-2 Standard Negative Pulse

6.5 COORDINATE SYSTEM

Each engineering logic drawing is divided into 32 zones (4 horizontal and 8 vertical) by marginal map coordinates. Figure references in the text are usually followed by a letter and a digit specifying the zone in which the referenced circuit is located. Physical reference to a drawing area such as "lower left" or "upper center" may also be used.

6.6 MODULE IDENTIFICATION

Two designations appear in or near each circuit symbol or inside the dotted line surrounding multiple circuit symbols shown on engineering drawings. The upper designation consists of four characters which specify the module type. Modules are identified by this designation in the Digital System Module Catalog while FLIP CHIP modules are described in the FLIP CHIP Module Catalog. Modules not described in the catalogs are described in this manual or in other referenced pertinent documents.

The lower designation is the module location code. The leftmost character of this designation is a number indicating the cabinet in which the module is located. The next character is a letter indicating the mounting panel in which the module is located. The last character consists of one or two numbers specifying the module location within the mounting panel. As an example, the designation 1A22 indicates that this module is mounted in location 22 of mounting panel A in cabinet 1. Terminal J of this module is designated at 1A22J.

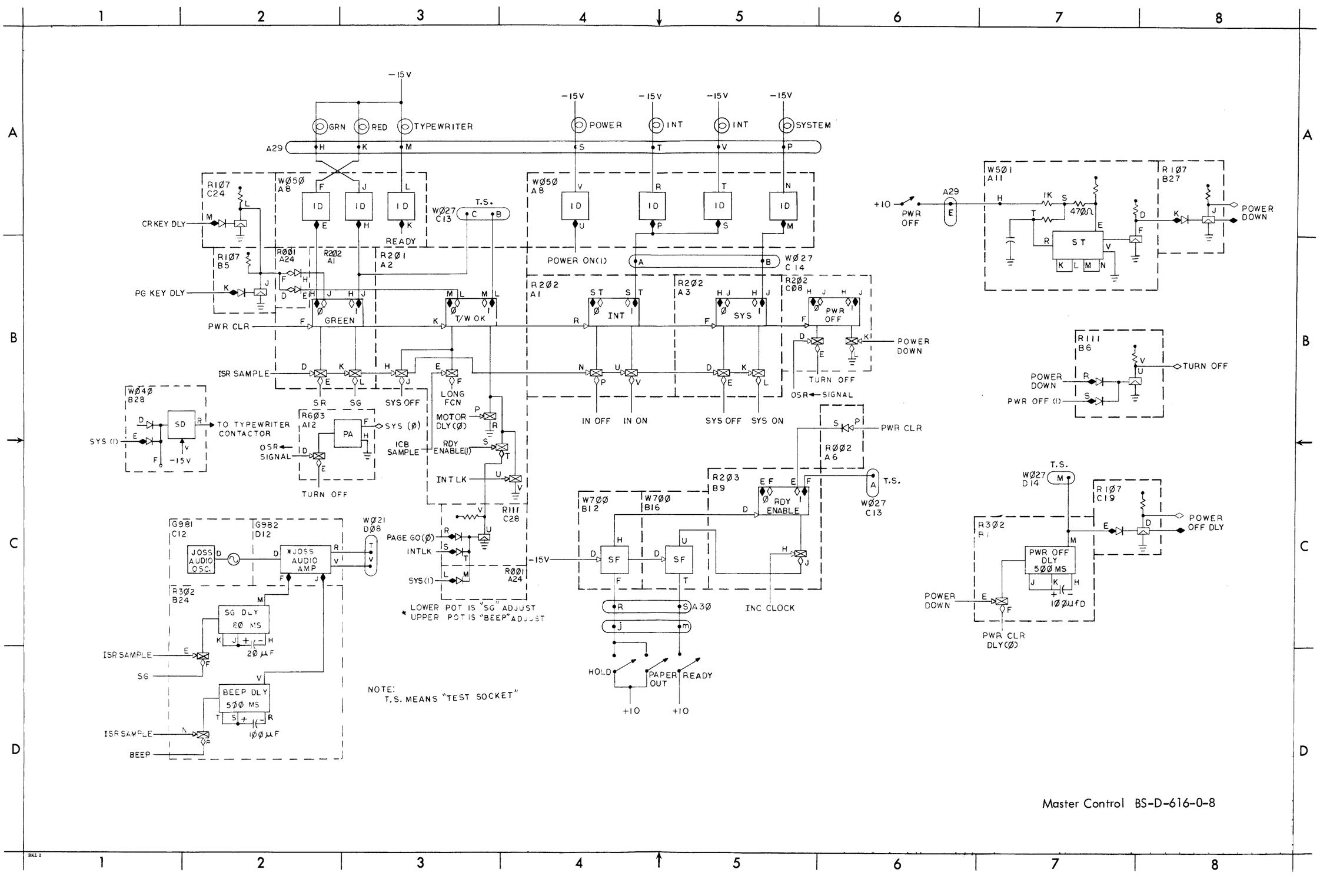
Module mounting panels which can accommodate more than one row of modules may be used in the construction of certain equipment. For this equipment, a letter is assigned to each row of modules within a mounting panel. When a particular device is contained within one cabinet, the number 1 may be omitted from the reference designations appearing on the associated drawing for that device.

6.7 SEMICONDUCTOR SUBSTITUTION

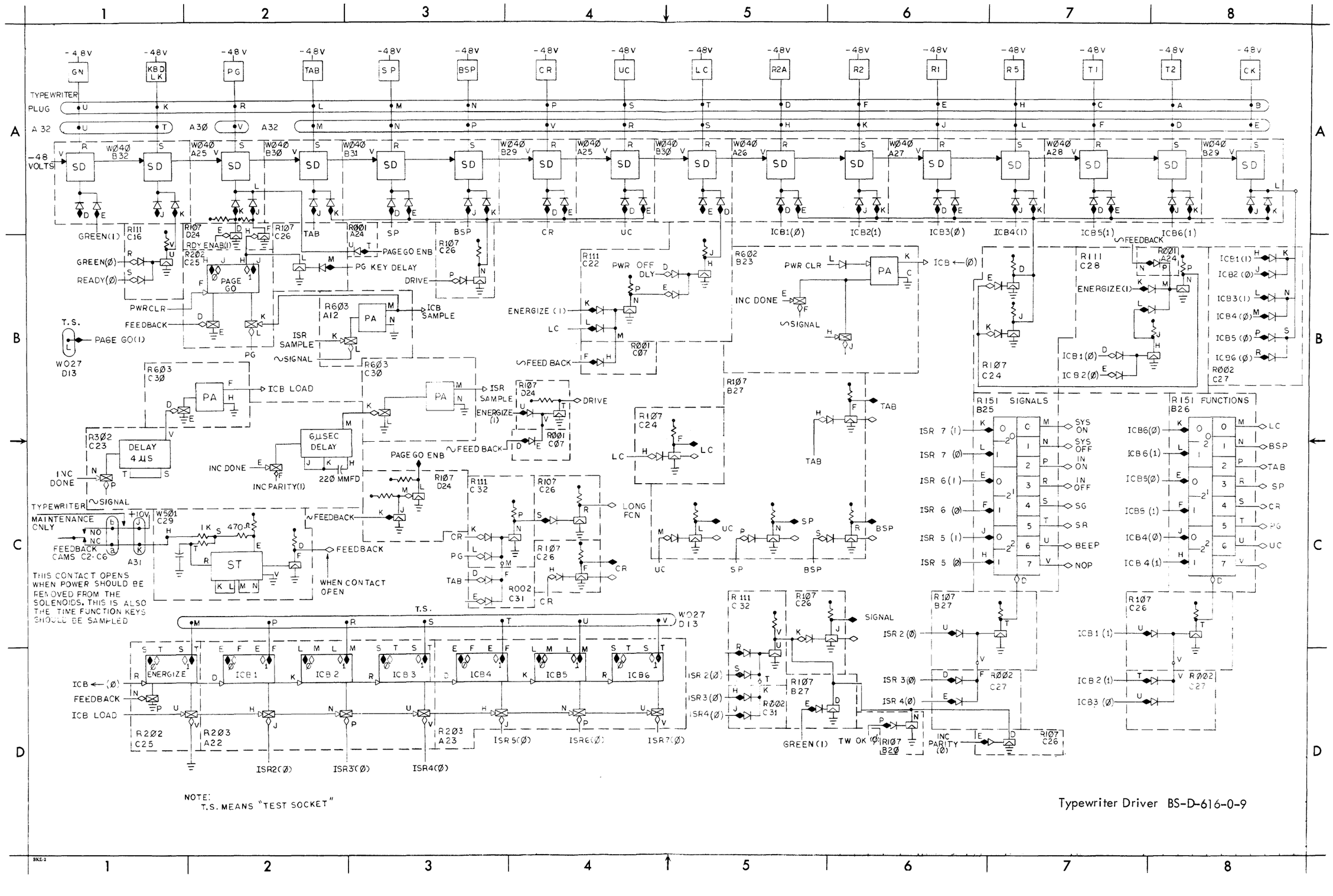
Standard EIA components specified in table 6-1 can replace most DEC semiconductors in modules of the JOSS Console, shown on the RS drawings for the console. Exact replacement is recommended for semiconductors not listed.

TABLE 6-1 SEMICONDUCTOR SUBSTITUTION

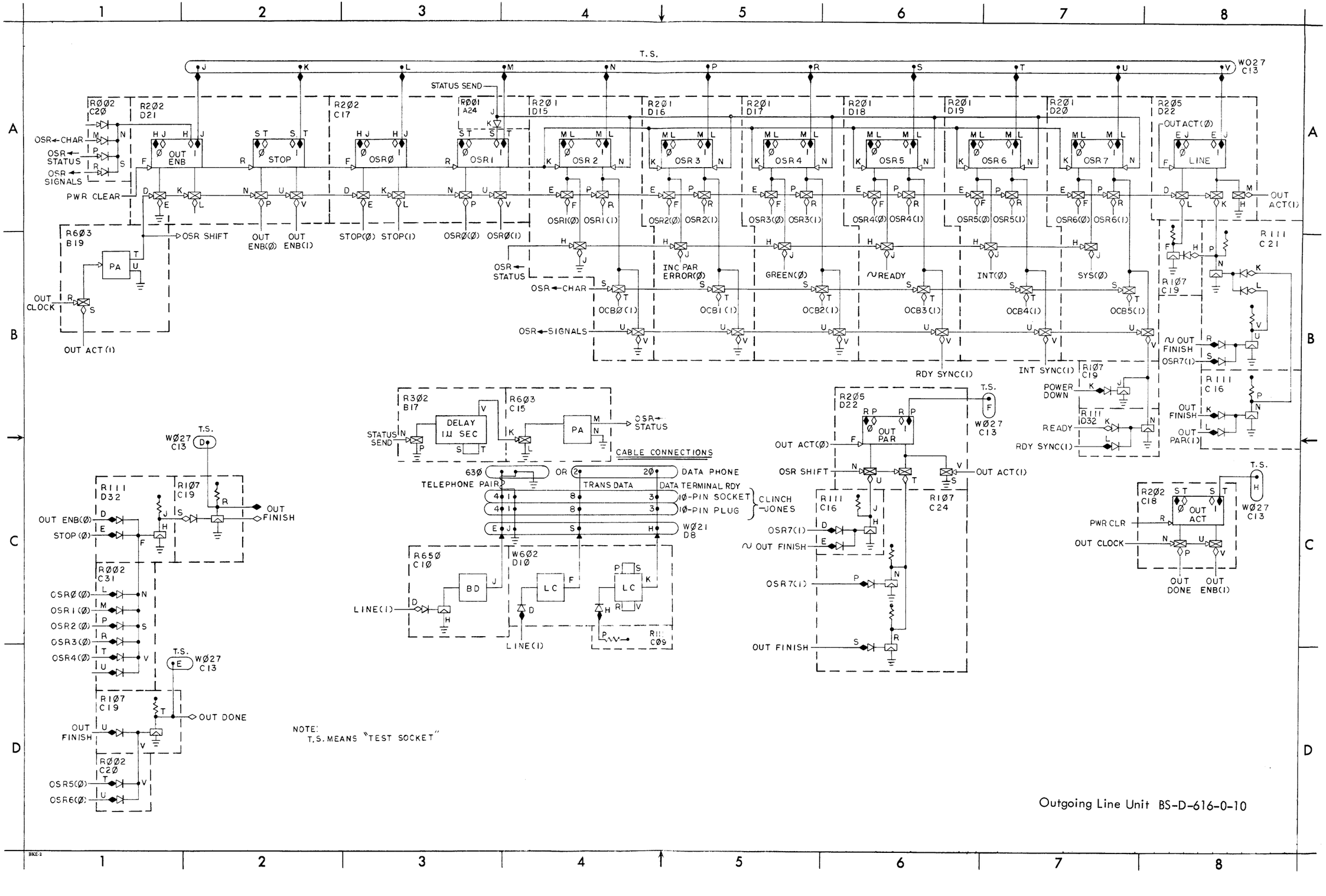
DEC	EIA	DEC	EIA
D-662	IN645	DEC 3639-0	2N3639
D-664	IN914	DEC 3639-2	2N3639-2
DEC 2219	2N2219	DEC 3790	2N3790
DEC 3009	2N3009	IN748	IN748 3.9v
DEC 3494	2N3494	MR2066	IN4003
DEC 3639	2N3639	4JXIC741	2N527



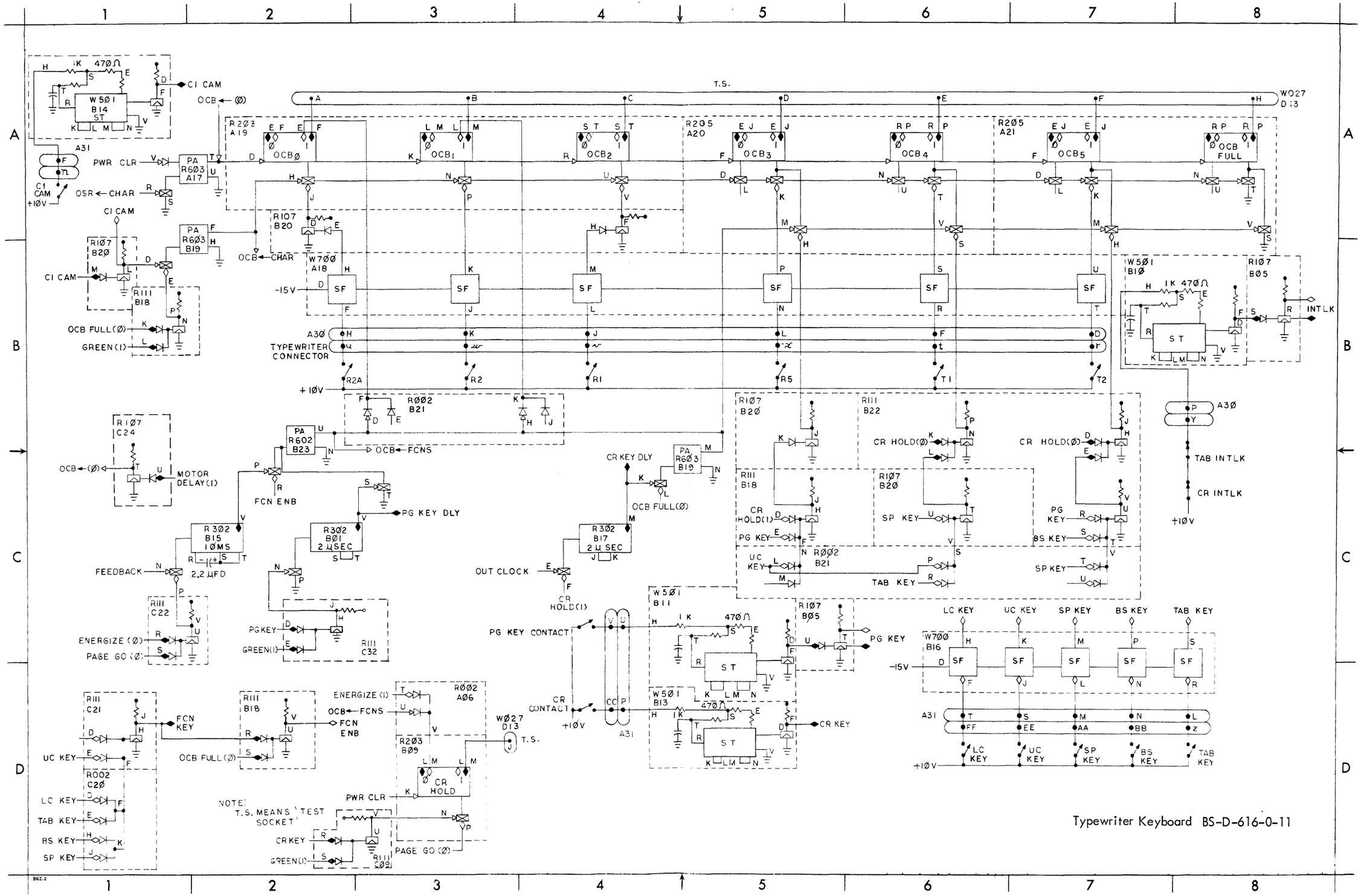
Master Control BS-D-616-0-8



Typewriter Driver BS-D-616-0-9

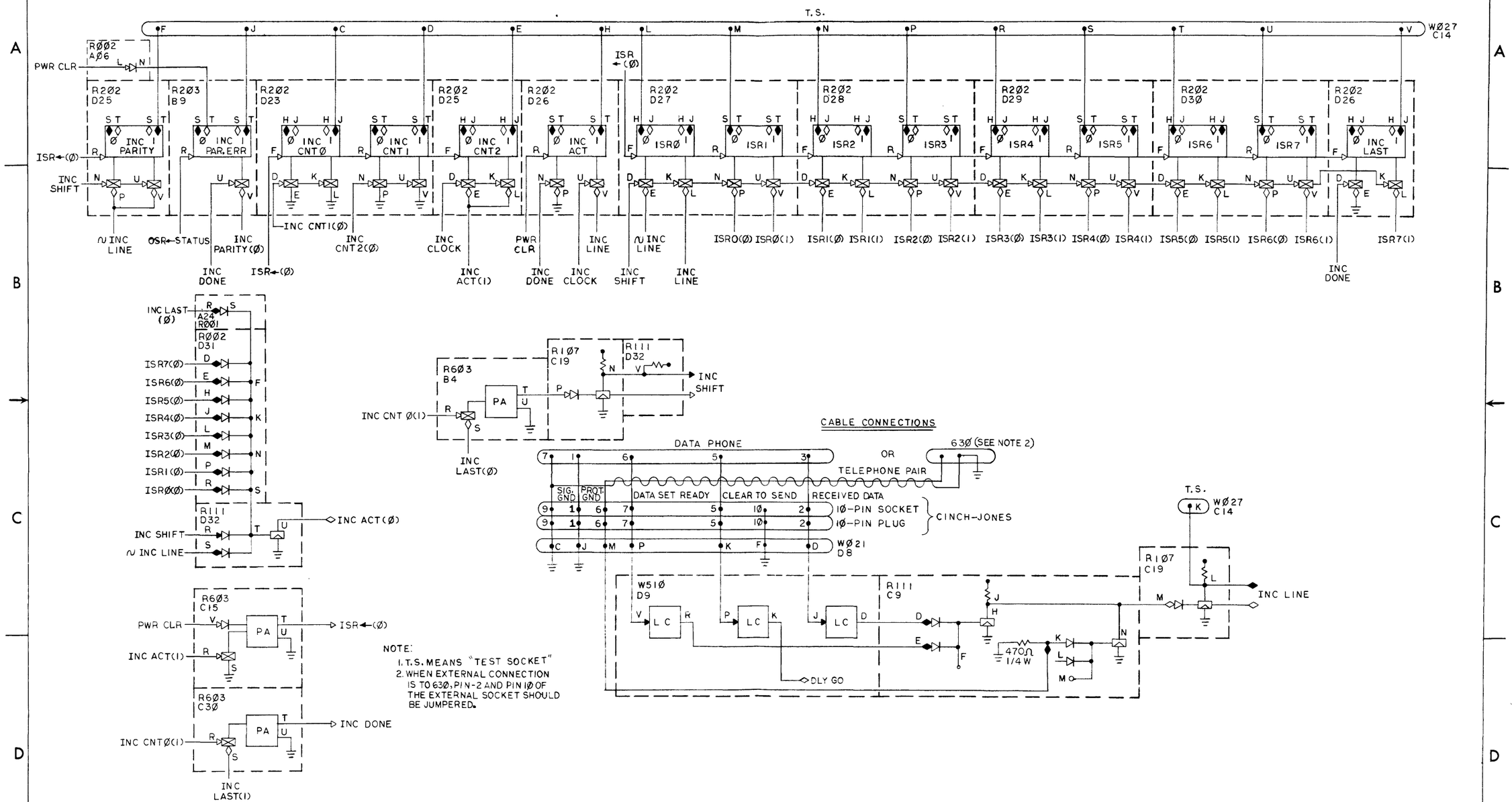


Outgoing Line Unit BS-D-616-0-10

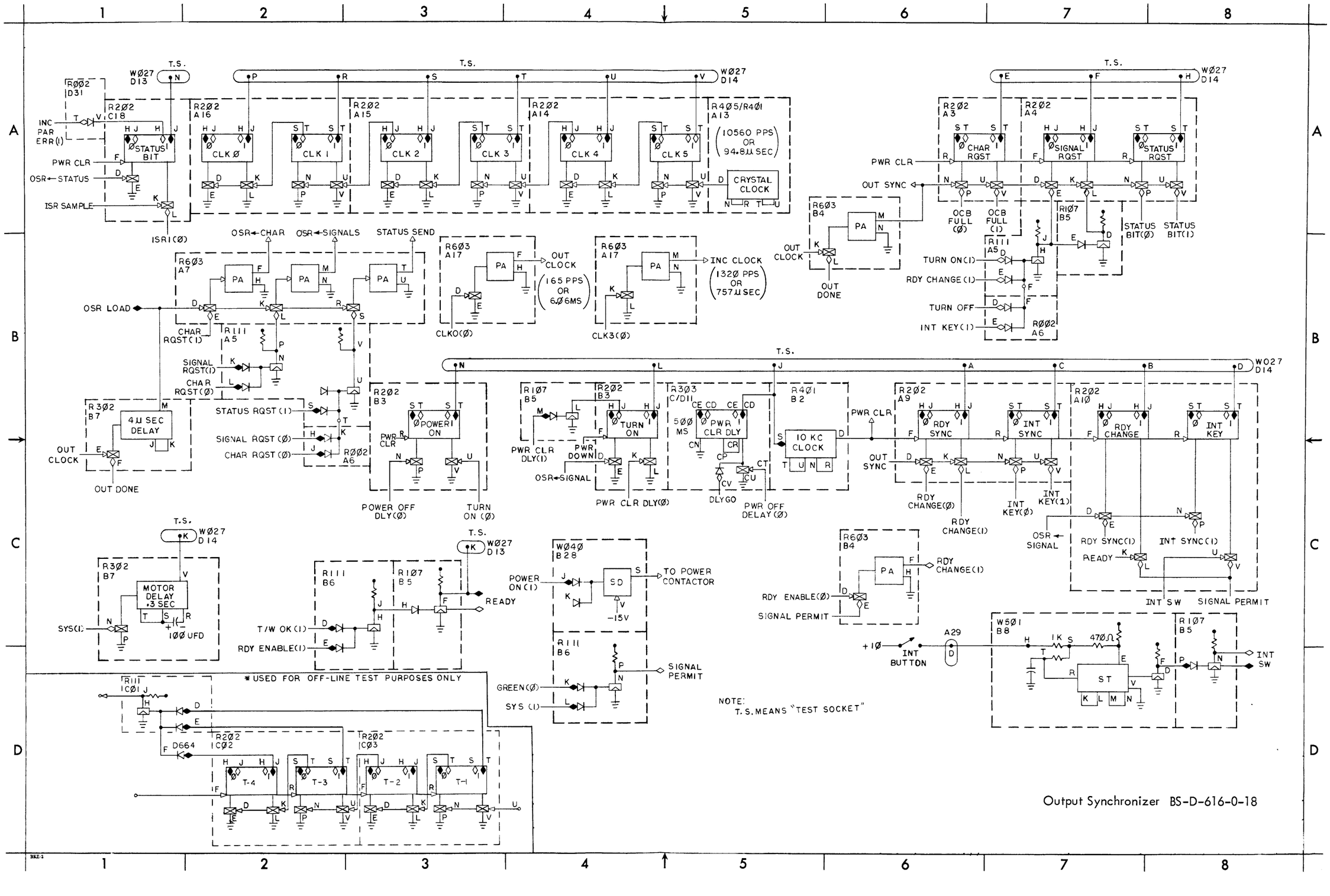


Typewriter Keyboard BS-D-616-0-11

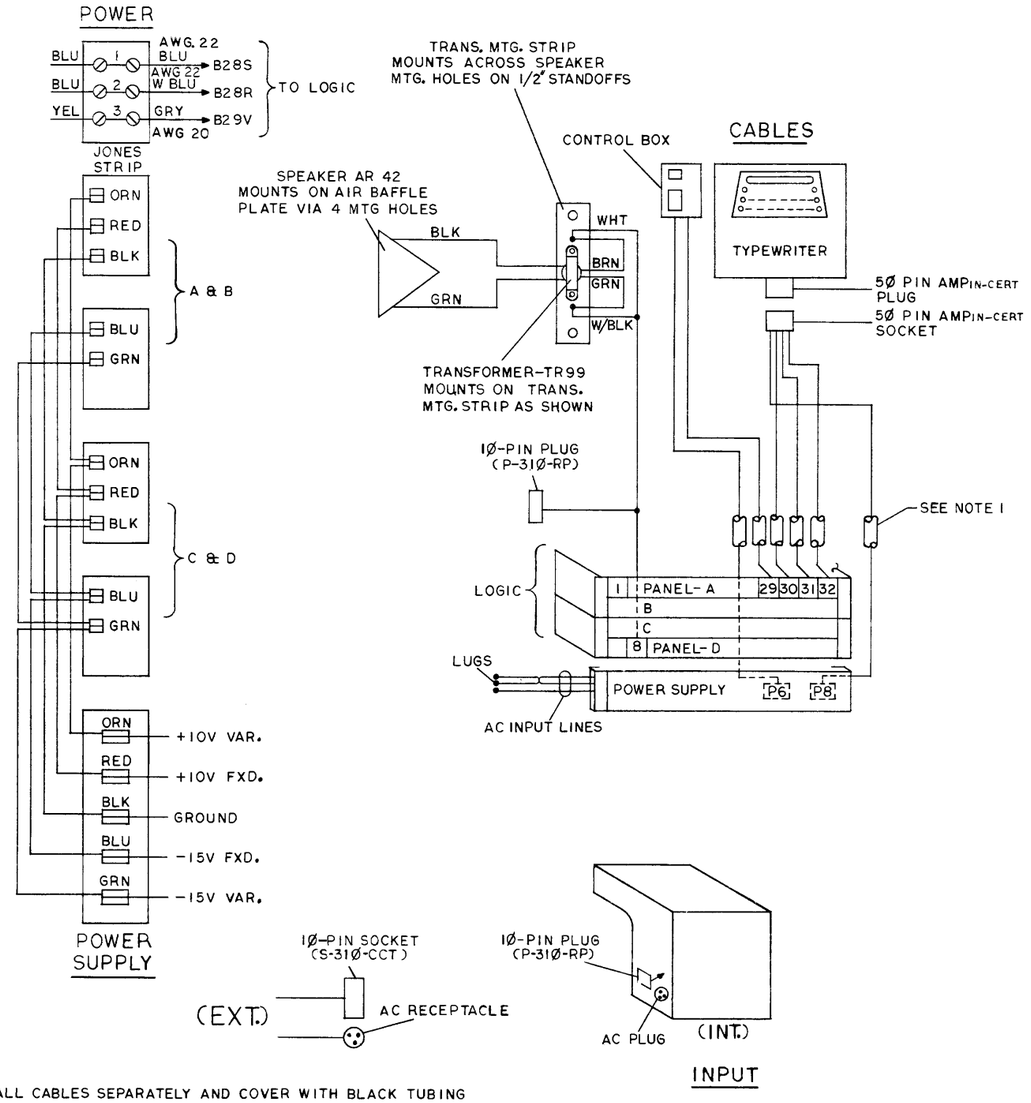
Incoming Line Unit BS-D-616-0-17



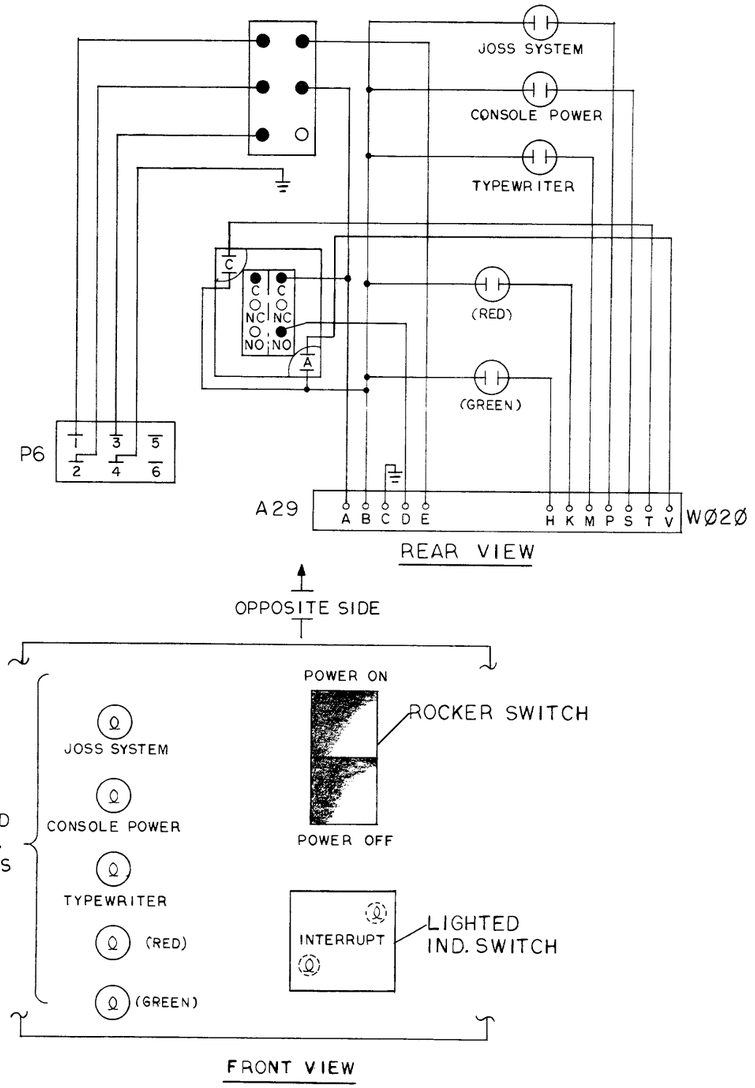
Incoming Line Unit BS-D-616-0-17



Output Synchronizer BS-D-616-0-18

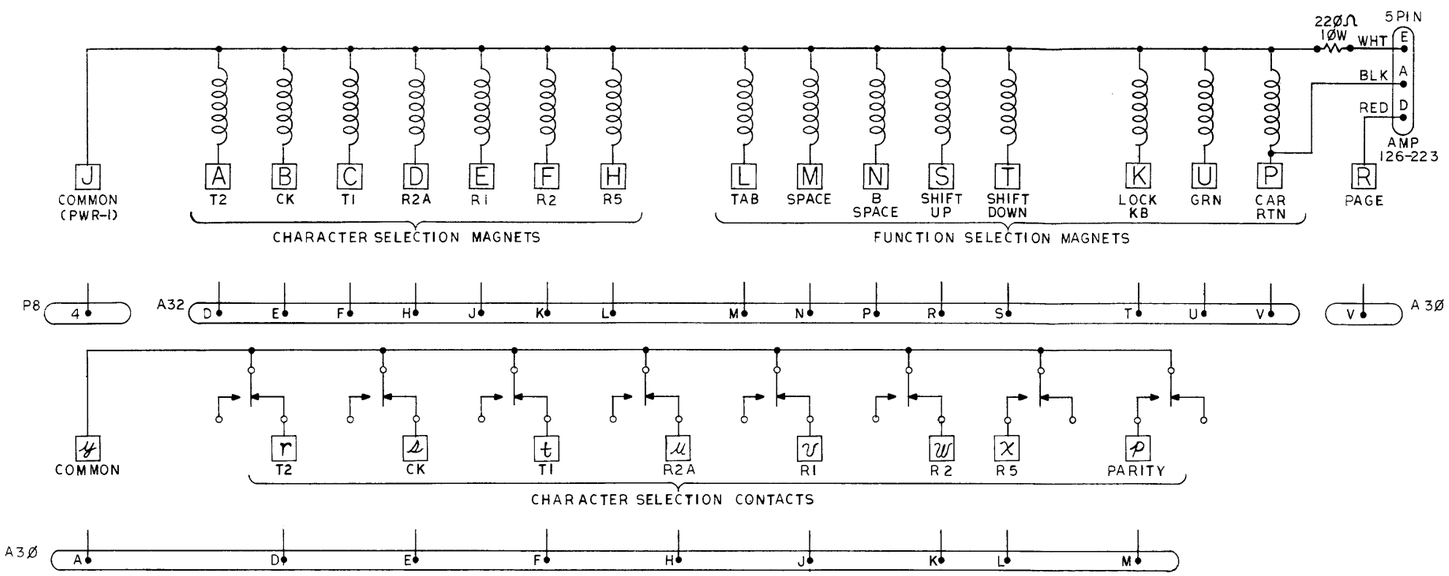


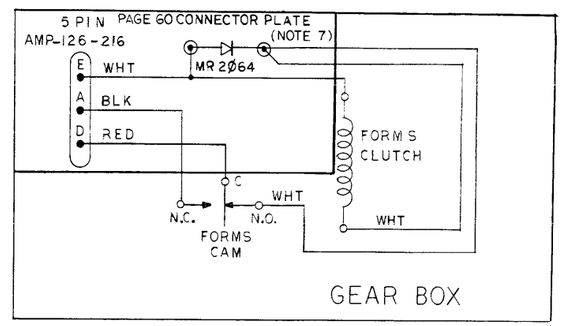
NOTES:
1. RUN ALL CABLES SEPARATELY AND COVER WITH BLACK TUBING
2. 50 PIN AMPIN-CERT SOCKET IS: 200277-4; PINS ARE 201328-1
3. SEE ASSOCIATED CABLE SCHEDULES FOR CONNECTIONS
4. ON WØ2Ø (A29-A32) REMOVE RESISTORS AND ADD SOLID-WIRE JUMPERS

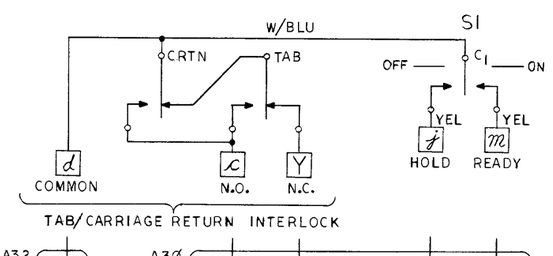


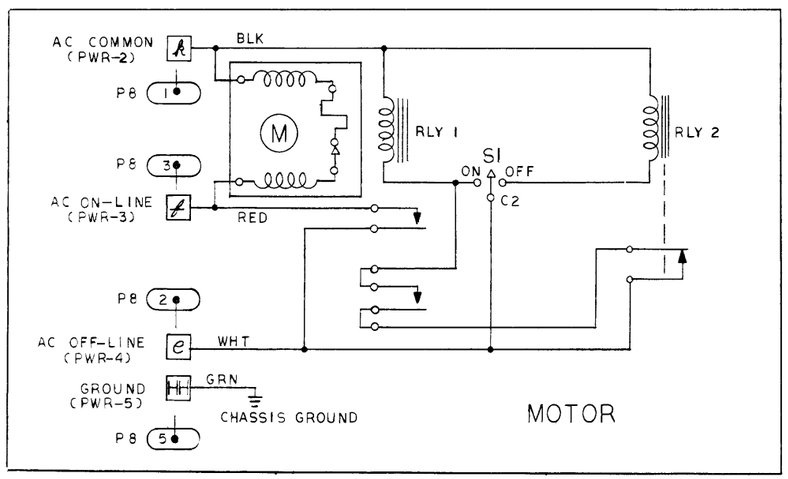
Console Connection, JOSS Console SD-D-616-0-23

Selectric I/O Typewriter (Sheet 1) SD-D-616-0-30





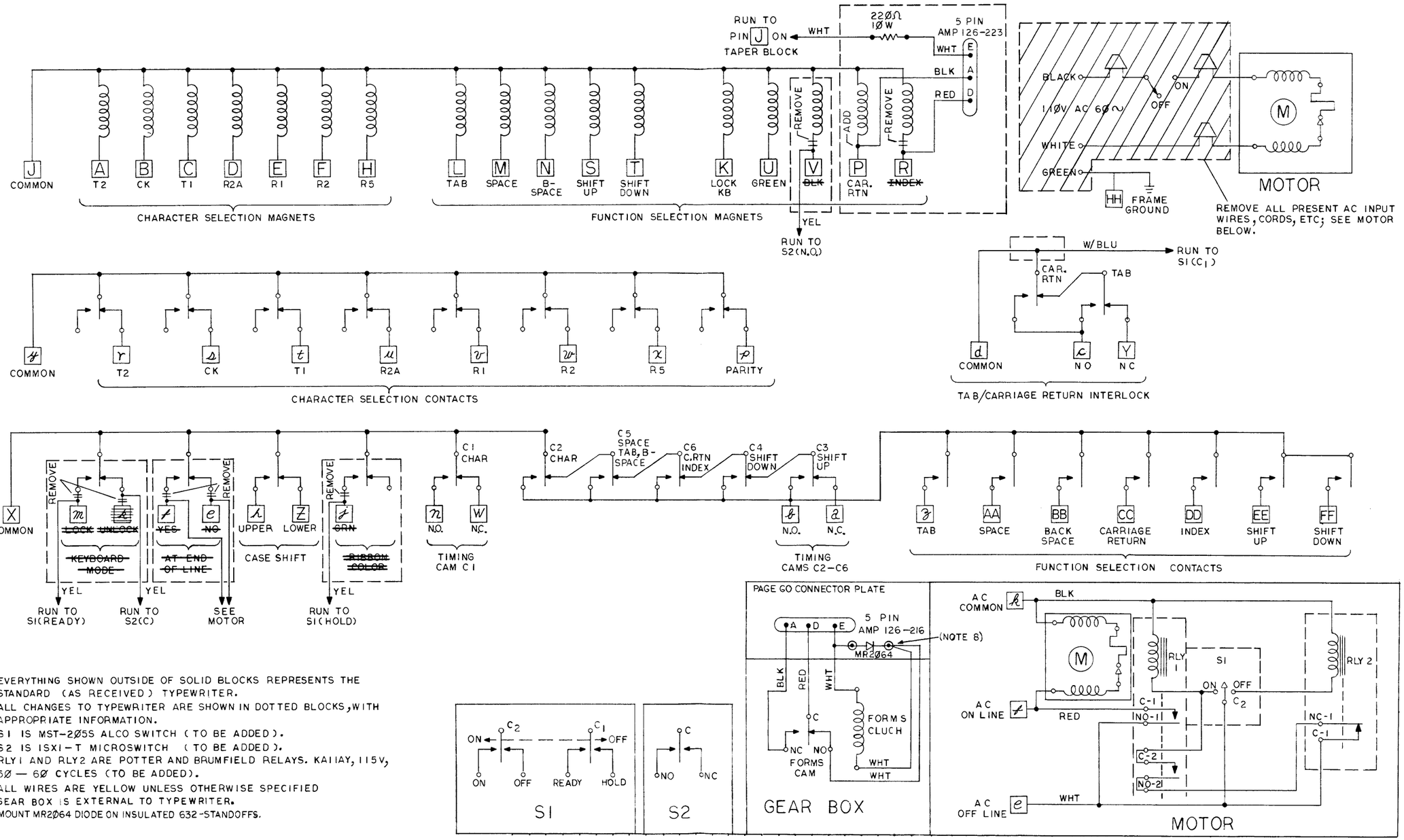




NOTES:

1. S1 IS MST-2Ø5S ALCO SWITCH.
2. S2 IS ISXI-T MICRO SWITCH.
3. RLY 1 AND RLY 2 ARE POTTER & BRUMFIELD RELAYS, KAIIAY, 115V, 5Ø–6Ø CYCLES.
4. ALL WIRES ARE YEL UNLESS OTHERWISE SPECIFIED.
5. SEE A-616-Ø-2Ø (4 PGS) FOR A3Ø, A31, A32 & P8 CABLES.
6. CABLES SHOWN ARE CONNECTED TO TERMINALS, TOWARDS WHICH WIRES ARE POINTING
7. MOUNT MR2Ø64 DIODE ON INSULATED 632-STANDOFFS.

Selectric I/O Typewriter (Sheet 2) SD-D-616-0-30



NOTES:

1. EVERYTHING SHOWN OUTSIDE OF SOLID BLOCKS REPRESENTS THE STANDARD (AS RECEIVED) TYPEWRITER.
2. ALL CHANGES TO TYPEWRITER ARE SHOWN IN DOTTED BLOCKS, WITH APPROPRIATE INFORMATION.
3. S1 IS MST-2Ø5S ALCO SWITCH (TO BE ADDED).
4. S2 IS ISX1-T MICROSWITCH (TO BE ADDED).
5. RLY1 AND RLY2 ARE POTTER AND BRUMFIELD RELAYS. KA11AY, 115V, 5Ø — 6Ø CYCLES (TO BE ADDED).
6. ALL WIRES ARE YELLOW UNLESS OTHERWISE SPECIFIED
7. GEAR BOX IS EXTERNAL TO TYPEWRITER.
8. MOUNT MR2Ø64 DIODE ON INSULATED 632-STANDOFFS.

Selectric I/O Typewriter (Sheet 2) SD-D-616-0-30

JACK ☐ PLUG ☒ FEMALE ☒ MALE ☐	LOCATION, LENGTH, ROUTE 1½ FT. FROM REAR OF WØ2Ø TO AMP SOCKET. USE SINGLE TEFLON WIRE AND WRAP IN BLACK TUBING. REMOVE RESISTORS FROM WØ2Ø AND ADD SOLID WIRE JUMPERS.			
COLOR	PIN WØ2Ø	PIN	NAME	REMARKS
BLK	A	y	+10V ------- CHAR. SEL	
	B			NO CONNECTION
	C			NO CONNECTION
BRN	D	r	T2 ------- CONTACTS	
RED	E	s	CK	
ORN	F	t	T1	
YEL	H	u	R2A	
GRN	J	v	R1	
BLU	K	w	R2	
VIO	L	x	R5	
GRY	M	p	PARITY	
WHT	N	c	INTLK, N.O.	
W/BLK	P	Y	INTLK. N.C.	
W/BRN	R	j	HOLD	
W/RED	S	m	READY ------- CONTACTS	
	T			NO CONNECTION
	U			NO CONNECTION
W/ORN	V	R	PAGE -------- MAGNET	
	* ADD DECAL		A30 ON REAR OF WØ2Ø	

JACK ☐ PLUG ☒ FEMALE ☒ MALE ☐	LOCATION, LENGTH, ROUTE 1½ FT. FROM REAR OF WØ2Ø TO AMP SOCKET. USE SINGLE TEFLON WIRE AND WRAP IN BLACK TUBING. REMOVE RESISTORS FROM WØ2Ø AND ADD SOLID WIRE JUMPERS.			
COLOR	PIN WØ2Ø	PIN	NAME	REMARKS
BLK	A	X	+10V FUNCTION	
	B			NO CONNECTION
	C			NO CONNECTION
BRN	D	h	U.C. MODE --- CONTACTS	
RED	E	Z	L.C. MODE	
ORN	F	n	C1 N.O.	
YEL	H	W	C1 N.C.	
GRN	J	b	C2 - C6 N.O.	
BLU	K	a	C2 - C6 N.C.	
VIO	L	z	TAB	
GRY	M	AA	SPACE	
WHT	N	BB	BACK SPACE	
W/BLK	P	CC	CAR. RET.	
W/BRN	R	DD	INDEX	
W/RED	S	EE	SHIFT UP	
W/ORN	T	FF	SHIFT DOWN	
W/YEL	U	V	PAGE ------- CONTACTS	
	V			NO CONNECTION
	* ADD DECAL		A31 ON REAR OF WØ2Ø	

JACK ☐ PLUG ☒ FEMALE ☒ MALE ☐	LOCATION, LENGTH, ROUTE 1½ FT. FROM REAR OF WØ2Ø TO AMP SOCKET. USE SINGLE TEFLON WIRE AND WRAP IN BLACK TUBING. REMOVE RESISTORS FROM WØ2Ø AND ADD SOLID WIRE JUMPERS.			
COLOR	PIN WØ2Ø	PIN	NAME	REMARKS
BLK	A	d	+10V (CR,TAB,INTLK) CNTK	
	B			NO CONNECTION
	C			NO CONNECTION
BRN	D	A	T2 ---------- MAGNET	
RED	E	B	CK	
ORN	F	C	T1	
YEL	H	D	R2A	
GRN	J	E	R1	
BLU	K	F	R2	
VIO	L	H	R5	
GRY	M	L	TAB	
WHT	N	M	SP	
W/BLK	P	N	BSP	
W/BRN	R	S	UC	
W/RED	S	T	LC	
W/ORN	T	K	KBDLK	
W/YEL	U	U	GN	
W/GRN	V	P	CR ---------- MAGNET	
	* ADD DECAL		A32 ON REAR OF WØ2Ø	

JACK ☐ PLUG ☒ FEMALE ☒ MALE ☐	LOCATION, LENGTH, ROUTE 2½ FT. FROM 8-PIN JONES PLUG TO AMP SOCKET. USE SINGLE AWG #20 WIRE. WRAP IN BLACK TUBING. (FROM 5Ø PIN AMP TO POWER SUPPLY)

COLOR	8-PIN PIN P-3Ø8-CCT	PIN	NAME	REMARKS
BLK	1	k	AC COMMON	
WHT	2	e	AC OFF-LINE	
RED	3	f	AC ON-LINE	
GRY	4	J	COMMON (-48V)	
GRN	5	HH	FRAME GROUND	
	6			NO CONNECTION
	7			NO CONNECTION
	8			NO CONNECTION

JACK ☐ PLUG ☒ FEMALE ☐ MALE ☒	LOCATION, LENGTH, ROUTE 3 FT. USE SINGLE TEFLON WIRE AND WRAP IN BLACK TUBING. ALLOW 4" OF WIRE TO EXTEND BEYOND TUBING. REMOVE RESISTORS FROM WØ2Ø AND ADD SOLID WIRE JUMPERS.			
COLOR	BOX PIN LOCATION	PIN	NAME	REMARKS
BLK	INT/PWR OFF	A	+10	
BRN	ALL LAMPS	B	-15V	
RED	BOX GND	C	GND	
ORN	INT (N.O.)	D	INTERRUP BUTTON	
YEL	PWR OFF	E	POWER-OFF SWITCH	
		F		NO CONNECTION
GRN	LAMP	H	RED LAMP	
		J		NO CONNECTION
BLU	LAMP	K	GREEN LAMP	
		L		NO CONNECTION
VIO	LAMP	M	T/W LAMP	
		N		NO CONNECTION
GRY	LAMP	P	SYSTEM LAMP	
		R		NO CONNECTION
WHT	LAMP	S	POWER LAMP	
W/BLK	LAMP	T	INT. LAMP	
		U		NO CONNECTION
W/BRN	LAMP	V	INT. LAMP	
	* ADD DECAL A29 ON REAR OF WØ2Ø			

Control Box Logic (Sheet 1) CL-A-616-0-21

JACK ☐ PLUG ☒ FEMALE ☐ MALE ☒	LOCATION, LENGTH, ROUTE 4 FT. USE SINGLE TEFLON AWG.#18 WIRE. COVER WITH BLK TUBING AND ALLOW 4" OF WIRE TO EXTEND (FROM CONTROL BOX TO POWER SUPPLY)

COLOR	BOX PIN LOCATION	PIN	NAME	REMARKS
WHT	PWR OFF	1	OFF	
RED	PWR COM	2	COMMON	
BLK	PWR ON	3	ON	
GRN	BOX GND	4	GROUND	
		5		NO CONNECTION
		6		NO CONNECTION

JACK [X] PLUG [ ] FEMALE [ ] MALE [X]	LOCATION, LENGTH, ROUTE 1 FT. FROM REAR OF WØ21 USE SINGLE TEFLON WIRE AND WRAP IN BLACK TUBING. (FROM LOGIC TO SIDE) ALLOW 2 1/2" OF WIRE TO EXTEND BEYOND TUBING

COLOR	PIN WØ21	PIN	NAME	REMARKS
	A			NOT USED ON WØ21
	B			NOT USED ON WØ21
BLK	C	9	GROUND 7	
BRN	D	2	INC LINE 3	
RED	E	4	LINE (1)	
W/BRN	F	1Ø	GROUND	
ORN	H	3	PWR CLR DLY (Ø) 2Ø	
YEL	J	1	GROUND 1	
GRN	K	5	INC LINE 5	
	L			NO CONNECTION
BLU	M	6	INC LINE	
	N			NO CONNECTION
VIO	P	7	INC LINE 6	
	R			NO CONNECTION
GRY	S	8	LINE (1) 2	
WHT	T		TRANS. PRIMARY	PRIMARY BROWN
	U			NO CONNECTION
W/BLK	V		TRANS. PRIMARY	PRIMARY GREEN
	* ADD DECAL	D8	ON REAR OF WØ21.	
NOTE: WHEN MAKING CABLE			LEAVE P-31Ø-RP CONNECTIONS HANGING.	

COLOR	NAME	PIN	PIN	REMARKS
	RES	A11H	A11S	1K ¼w
	RES	A11S	A11E	470 OHM ¼w
(TANT)	CAP	BØ1K(+)	BØ1H (-)	100 ufd
	RES	CØ9K	GND	470 OHM ¼w
	RES	C29H	C29S	1K ¼w
	RES	C29S	C29E	470 OHM ¼w
(TANT)	CAP	BØ7S(+)	BØ7R(-)	1ØØ ufd
	RES	BØ8H	BØ8S	1K ¼w
	RES	BØ8S	BØ8E	47Ø OHM ¼w
(TANT)	CAP	B15S(+)	B15R(-)	2.2 ufd
	RES	B14H	B14S	1K ¼w
	RES	B14S	B14E	47Ø OHM ¼w
	RES	B11H	B11S	1K ¼w
	RES	B11S	B11E	47Ø OHM ¼w
	RES	B13H	B13S	1K ¼w
	RES	B13S	B13E	47Ø ¼w
	RES	B1ØH	B1ØS	1K ¼w
	RES	B1ØS	B1ØE	47Ø OHM ¼w
	DIODE	CØ2J	CØ1F	D664
(TANT)	CAP	B24J(+)	B24H(-)	2Ø ufd
(TANT)	CAP	B24S(+)	B24R(-)	1ØØ ufd
	CAP	C23H	C23J	220 MMFD

External Components List CL-A-616-0-31

COLOR	NAME	PIN	PIN	REMARKS
BLUE	RDY ENABLE(1)	C13A	BØ9F	
BLUE	T/W OK(1)	C13B	AØ2L	
BLUE	GREEN(1)	C13C	AØ1J	
WHITE	OUT FINISH ◄—	C13D	C19R	
WHITE	~OUT DONE ◄—	C13E	C19T	
BLUE	OUT PAR(1)	C13F	D22P	
BLUE	OUT ACT(1)	C13H	C18T	
BLUE	OUT ENB(1)	C13J	D21J	
BLUE	STOP(1)	C13K	D21T	
BLUE	OSR 0(1)	C13L	C17J	
BLUE	OSR 1(1)	C13M	C17T	
BLUE	OSR 2(1)	C13N	D15L	
BLUE	OSR 3(1)	C13P	D16L	
BLUE	OSR 4(1)	C13R	D17L	
BLUE	OSR 5(1)	C13S	D18L	
BLUE	OSR 6(1)	C13T	D19L	
BLUE	OSR 7(1)	C13U	D2ØL	
BLUE	LINE(1)	C13V	D22J	
NOTE:				
1.	ALL CONNECTORS ARE WØ27.			

Inter-Panel Connector Typewriter Console (Sheet 1) WS-A-616-0-13

COLOR	NAME	PIN	PIN	REMARKS
BLUE	INT(1)	C14A	AØ1T	
BLUE	SYS(1)	C14B	AØ3J	
BLUE	INC CNT 0(1)	C14C	D23J	
BLUE	INC CNT 1(1)	C14D	D23T	
BLUE	INC CNT 2(1)	C14E	D25J	
BLUE	INC PARITY(1)	C14F	D25T	
BLUE	INC ACT(1)	C14H	D26T	
BLUE	INC PAR ERR(1)	C14J	B9T	
WHT	INC LINE—◆	C14K	C19L	
YEL	ISR 0(Ø)	C14L	D27H	
YEL	ISR 1(Ø)	C14M	D27S	
YEL	ISR 2(Ø)	C14N	D28H	
YEL	ISR 3(Ø)	C14P	D28S	
YEL	ISR 4(Ø)	C14R	D29H	
YEL	ISR 5(Ø)	C14S	D29S	
YEL	IST 6(Ø)	C14T	D3ØH	
YEL	ISR 7(Ø)	C14U	D3ØS	
BLUE	INC LAST(1)	C14V	D26J	

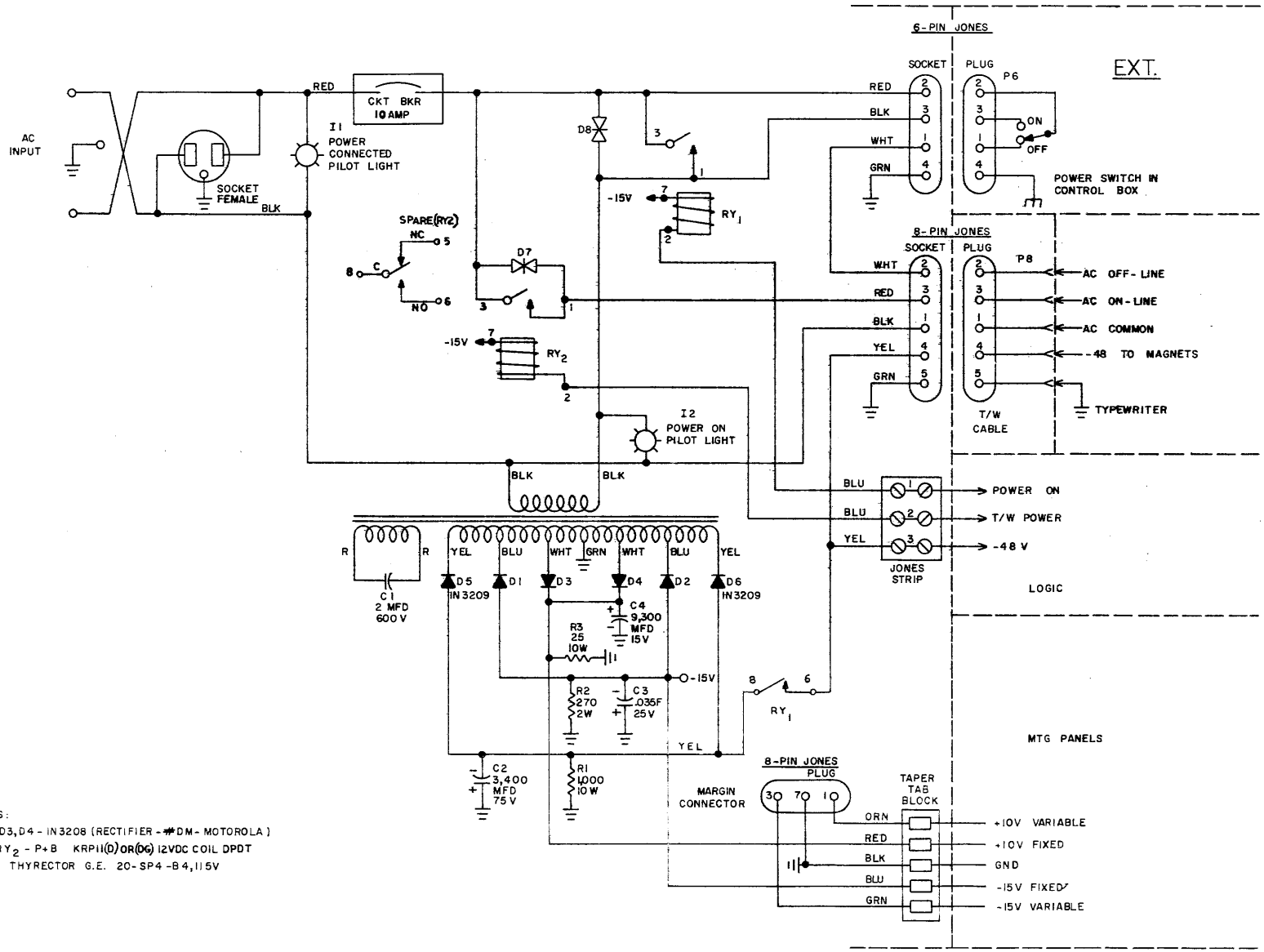
Inter-Panel Connector Typewriter Console (Sheet 2) WS-A-616-0-13

COLOR	NAME	PIN	PIN	REMARKS
BLUE	OCB 0(1)	D13A	A19F	
BLUE	OCB 1(1)	D13B	A19M	
BLUE	OCB 2(1)	D13C	A19T	
BLUE	OCB 3(1)	D13D	A2ØJ	
BLUE	OCB 4(1)	D13E	A2ØP	
BLUE	OCB 5(1)	D13F	A21J	
BLUE	OCB FULL(1)	D13H	A21P	
BLUE	CR HOLD(1)	D13J	BØ9M	
WHITE	READY——◆	D13K	BØ5F	
BLUE	PAGE GO(1)	D13L	C25J	
BLUE	ENERGIZE(1)	D13M	C25T	
BLUE	STATUS BIT(1)	D13N	C18J	
BLUE	ICB 1(1)	D13P	A22F	
BLUE	ICB 2(1)	D13R	A22M	
BLUE	ICB 3(1)	D13S	A22T	
BLUE	ICB 4(1)	D13T	A23F	
BLUE	ICB 5(1)	D13U	A23M	
BLUE	ICB 6(1)	D13V	A23T	

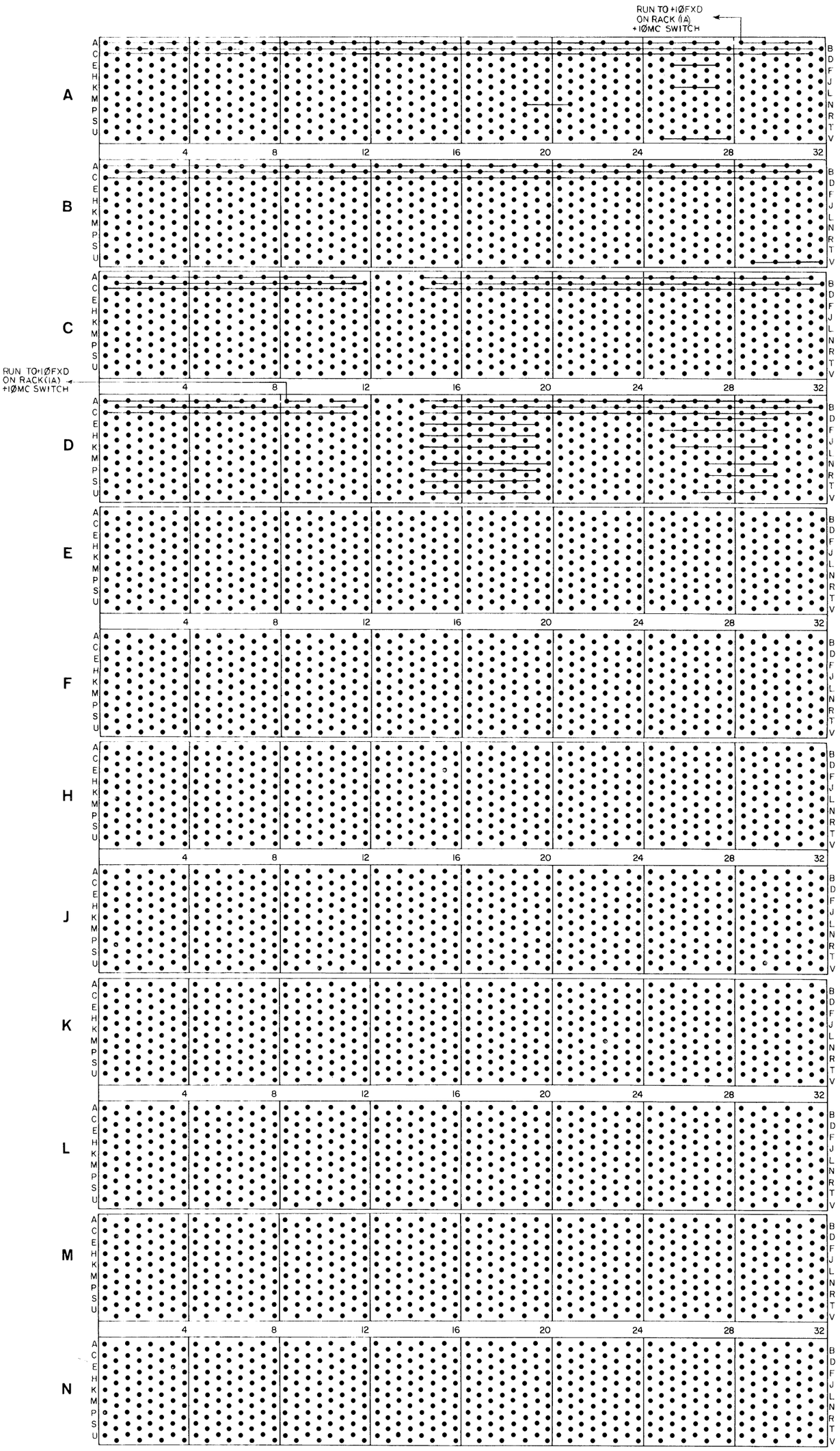
Inter-Panel Connector Typewriter Console (Sheet 3) WS-A-616-0-13

COLOR	NAME	PIN	PIN	REMARKS
BLUE	RDY SYNC(1)	D14A	AØ9J	
BLUE	RDY CHANGE(1)	D14B	A1ØJ	
BLUE	INT SYNC(1)	D14C	AØ9T	
BLUE	INT KEY(1)	D14D	A1ØT	
BLUE	CHAR RQST(1)	D14E	AØ3T	
BLUE	SIGNAL RQST(1)	D14F	AØ4J	
BLUE	STATUS RQST(1)	D14H	AØ4T	
WHITE	PWR CLR DLY—◆	D14J	D11S	
WHITE	MOTOR DLY—◆	D14K	BØ7V	
BLUE	TURN ON(1)	D14L	BØ3J	
WHITE	PWR OFF DLY—◆	D14M	BØ1M	
BLUE	POWER ON(1)	D14N	BØ3T	
BLUE	CLK 0(1)	D14P	A16J	
BLUE	CLK 1(1)	D14R	A16T	
BLUE	CLK 2(1)	D14S	A15J	
BLUE	CLK 3(1)	D14T	A15T	
BLUE	CLK 4(1)	D14U	A14J	
BLUE	CLK 5(1)	D14V	A14T	

Inter-Panel Connector Typewriter Console (Sheet 4) WS-A-616-0-13



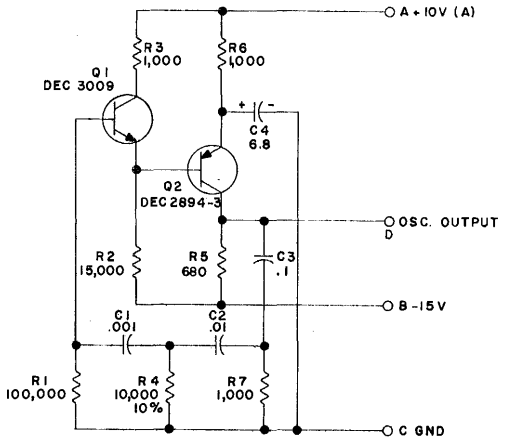
NOTES:
D1,D2,D3,D4 - IN3208 (RECTIFIER - #DM- MOTOROLA)
RY_1 + RY_2 - P+B KRP11(D) OR(DG) 12VDC COIL DPDT
D7,D8 THYRECTOR G.E. 20-SP4-B4,115V



Bus Schedule WD-D-616-0-15

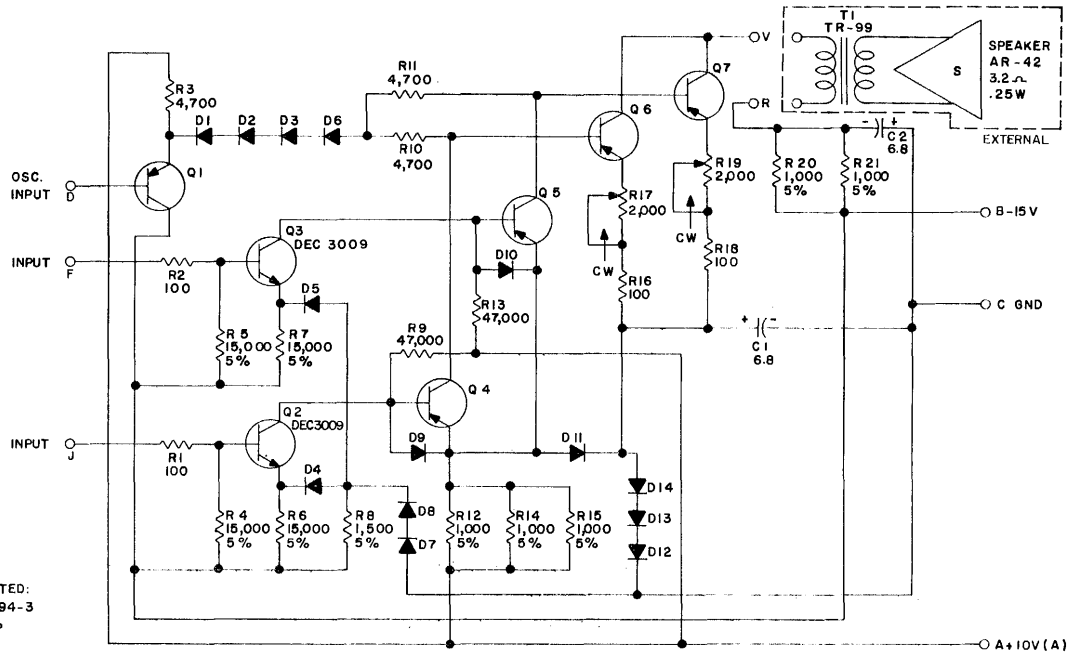
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
	R202	R201	R202	R202	R111	R002	R603	W050	R202	R202	W501	R603	R405	R202	R202	R202	R603	W700	R203	R205	R205	R203	R203	R001	W040	W040	W040	W040				
A	GREEN	T/W OK	SYS	SIGNAL RQST	SIGNAL RQST	SIGNAL RQST STATUS SEND	OSR ← CHAR	RED GREEN READY	RDY SYNC	RDY CHANGE	POWER DOWN	SYS	CRYSTAL CLOCK	CLK 4	CLK 2	CLK 0	OUT CLOCK	R2A R2	OCB_0	OCB_3	OCB_5	ICB 1	ICB 4	TW OK GREEN OSP(1)	UC	R2A	R1	T1	CONTROL BOX CONN. A29	T/W CONN. A A30	T/W CONN. B A31	T/W CONN. C A32
	INT		CHAR RQST	STATUS RQST	OSR ← SIGNALS STATUS SEND	INC PAR ERR(Ø) RDY ENABLE CR HOLD	OSR ← SIGNALS STATUS SEND	SYSTEM INT INT POWER	INT SYNC	INT KEY		ICB SAMPLE		CLK 5	CLK 3	CLK 1	INC CLOCK OCB ← (Ø)	R1 R5 T1 T2	OCB_1 OCB_2	OCB_4	OCB FULL	ICB 2 ICB 3	ICB 5 ICB 6	TW OK FEEDBACK INC ACT PAGE SO	PG	R2	R5	T2				
	R302	R401	R202	R603	R107	R111	R302	W501	R203	W501	W501	W700	W501	W501	R302	W700	R302	R111	R603	R107	R002	R111	R602	R302	R151	R151	R107	W040	W040	W040	W040	W040
B	PWR OFF DLY PG KEY DLY	PWR CLR	TURN ON PWR ON	RDY CHANGE OUT SYNC INC SHIFT	SIGNAL RQST READY PG KEY DLY TURN ON INT SW INT LK PG KEY	READY SIGNAL PERMIT TURN OFF	OSR LOAD MOTOR DELAY	INT SW	RDY ENABLE CR HOLD INC PAR ERR	INT LK	PG KEY	RDY ENABLE	CR KEY	C1 CAM	OCB ← FCNS	LC KEY UC KEY SP KEY BS KEY TAB KEY RDY ENABLE	OCB ← FCNS OSR ← STATUS	OCB 3 OCB CHAR FCN ENB	OCB ← CHAR OCB FCNS OSR SHIFT	OCB 0 OCB 2 OCB 3 C1 CAM SIGNAL OCB 4	OCB 0 OCB 1 OCB 3 OCB 4 OCB 5	OCB 5 OCB 4 OCB 5	ICB ← (Ø) OCB ← FNCS	SG DLY BEEP DLY	SIGNALS	FUNCTIONS	SIGNAL TAB POWER DOWN UC SP BSP FUNCTIONS	T/W ON PWR ON	CR CK	LC TAB	SP BSP	GN KBD LK
	*	*	*				R001	R202	R111	R650	R303	G981			R603	R111	R202	R202	R107	R002	R111	R111	R302	R107	R202	R107	R002	R111	W501	R603	R002	R111
C	← (R111)	RESERVED FOR TESTER (R202)	→ (R202)				DRIVE LC	PWR OFF	INC LINE DATA TERMINAL READY CR HOLD	LINE(1)	POWER CLEAR DELAY	JOSS AUDIO OSC.	INDIC. CI3	INDIC. CI4	OSR ← STATUS ISR (Ø)	OUT PAR LINE KBD LK	OSR 0 OSR 1	STATUS BIT OUT ACT	PWR OFF DLY LINE OSR 7 INC LINE INC SHIFT OUT FINISH OUT DONE	FCN KEY FCN KEY OUT ENB OUT ENB OUT DONE	FCN KEY LINE LINE	LC OCB ← FCNS	ICB SAMPLE ICB LOAD	R5 LC R5 CR KEY DLY OUT PAR OUT PAR OCB ← Ø	PAGE GO ENERGIZE	SIGNAL CR SIGNAL PAGE GO ENERGIZE LONG FCN SIGNALS	FUNCTIONS CK CK CK SIGNALS	R5 R5 TW OK	FEEDBACK	ICB LOAD ISR SAMPLE INC DONE	LONG FCN SIGNAL OUT FINISH OUT FINISH OUT FINISH	P6 KEY DLY LONG FCN SIGNAL
								W510		W602		G982			R201	R201	R201	R201	R201	R201	R202	R205	R202	RIØ7	R202	R202	R202	R202	R202	R202	R002	R111
D								LINE CONNECTOR D8	REC DATA CLEAR TO SEND D.S. READY	XMT DATA D.T. READY		JOSS AUDIO AMP	INDIC. DI3	INDIC. DI4	OSR 2	OSR 3	OSR 4	OSR 5	OSR 6	OSR 7	OUT ENB STOP	LINE OUT PAR	INC CNT_0 INC CNT 1	RDY ENABLE PAGE SO PAGE GO ENB PAGE GO ENB DRIVE	INC CNT 2 INC PARITY	INC LAST INC ACT	ISR 0 ISR 1	ISR 2 ISR 3	ISR 4 ISR 5	ISR 6 ISR 7	INC ACT(Ø) INC ACT(Ø) INC ACT(Ø) INC ACT(Ø) STATUS BIT(1)	OUT FINISH CSR 7 INC ACT INC SHIFT

* FOR OFF-LINE PURPOSES ONLY. NOT INCLUDED AS PART OF UML MODULES.



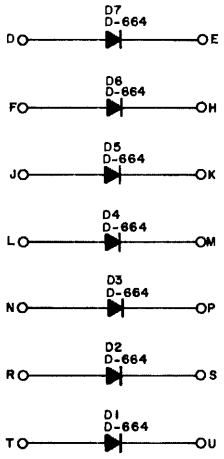
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W, 5%
CAPACITORS ARE MFD

JOSS Audio Oscillator RS-B-G981

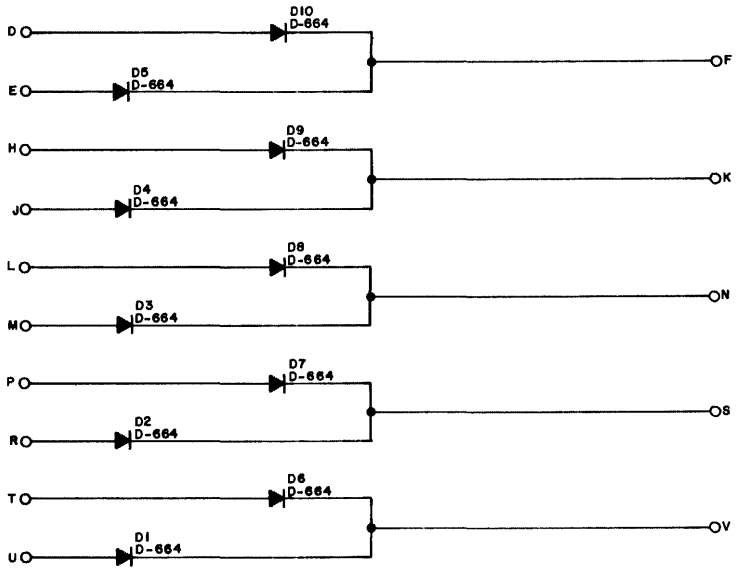


UNLESS OTHERWISE INDICATED:
TRANSISTORS ARE DEC 2894-3
RESISTORS ARE 1/4W, 10%
CAPACITORS ARE MFD
DIODES ARE D-662
T1 IS PRI IMP -- 500Ω, SEC IMP -- 3.2Ω
R17, R19 ARE BOURNS POT

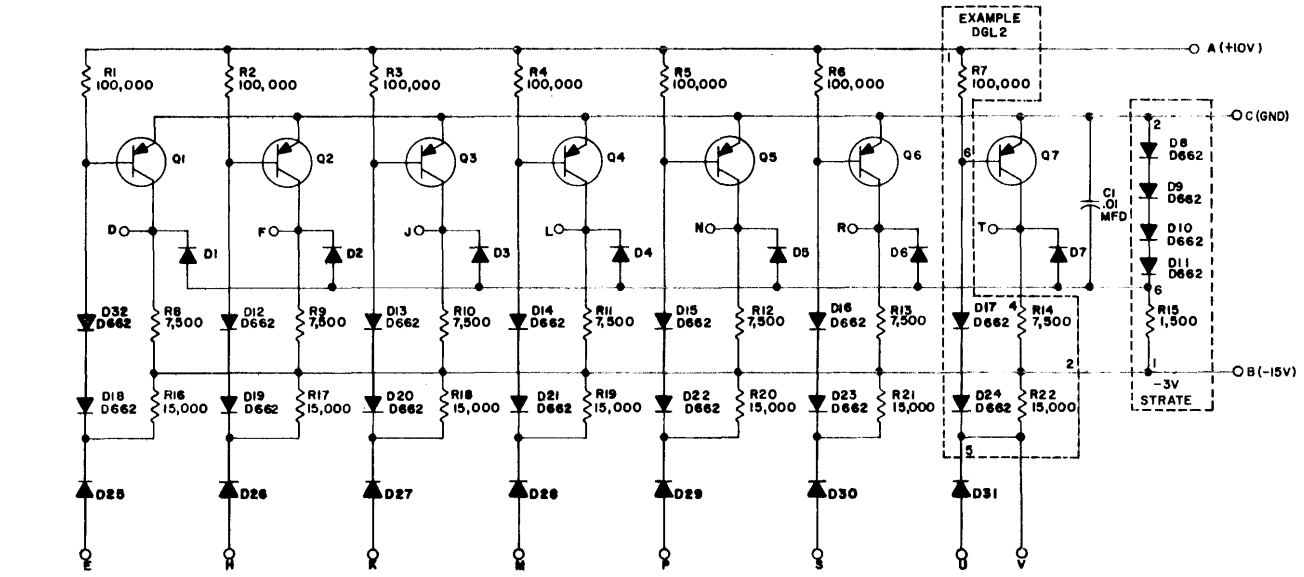
JOSS Audio Amplifier RS-B-G982



Diode Network RS-B-R001

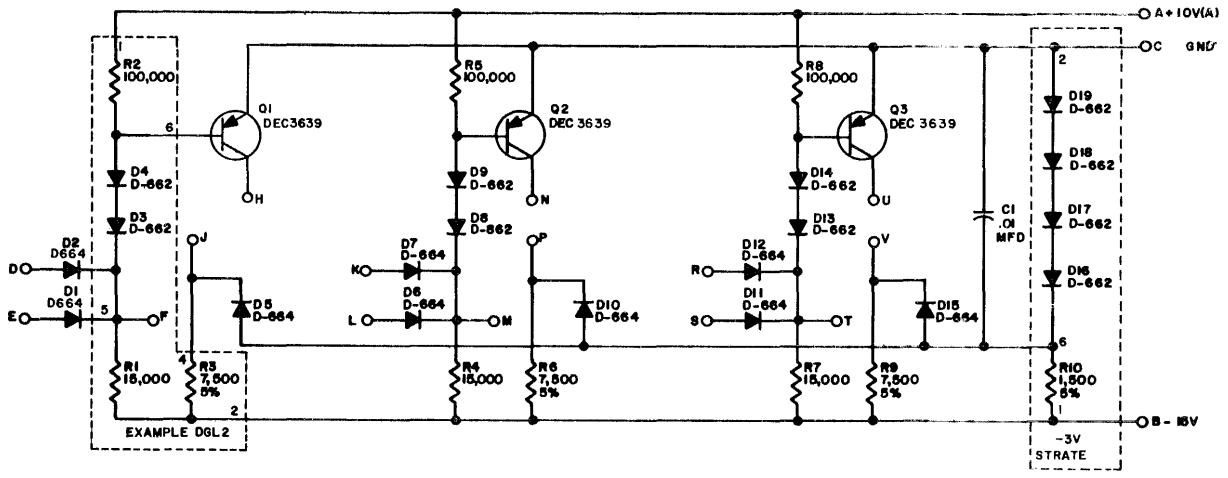


Diode Cluster RS-B-R002



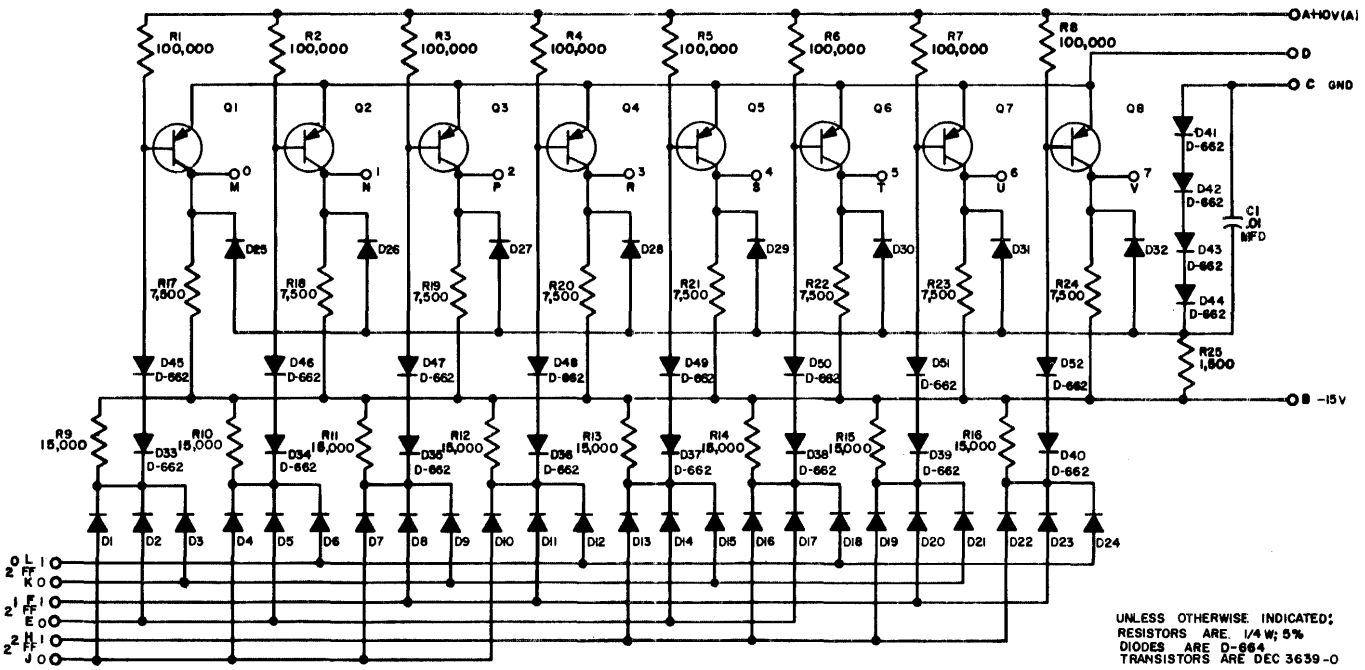
UNLESS OTHERWISE INDICATED;
RESISTORS ARE 1/4W, 5%
DIODES ARE D-664
TRANSISTORS ARE DEC 3639-0
PRINTED CIRCUIT REV. FOR
DGL BOARD IS SIA

Inverter RS-B-R107

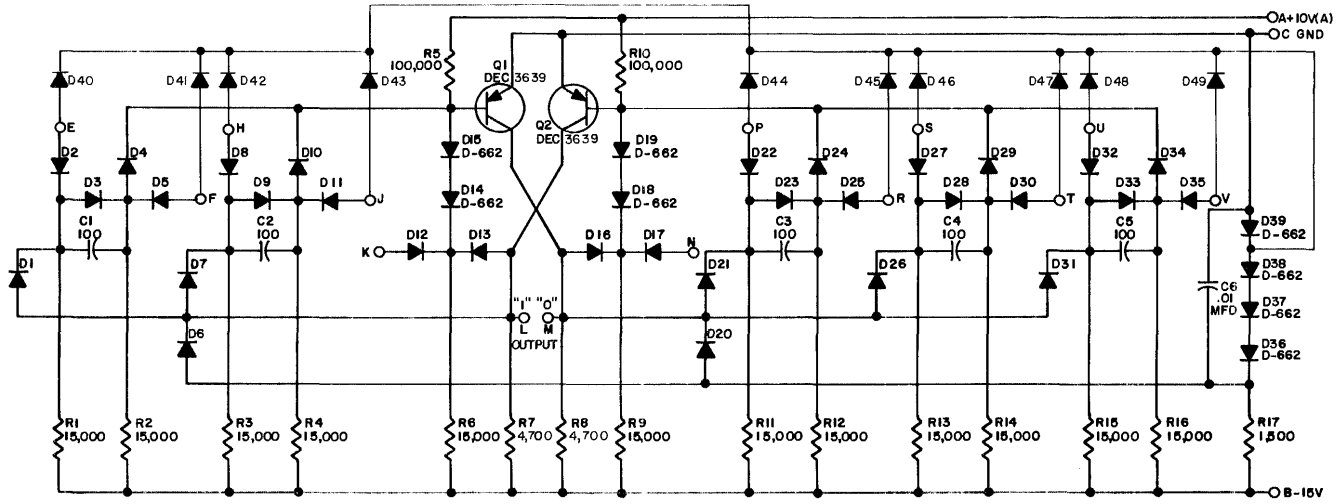


UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 5%
PRINTED CIRCUIT REV. FOR
DGL BOARD IS SIA

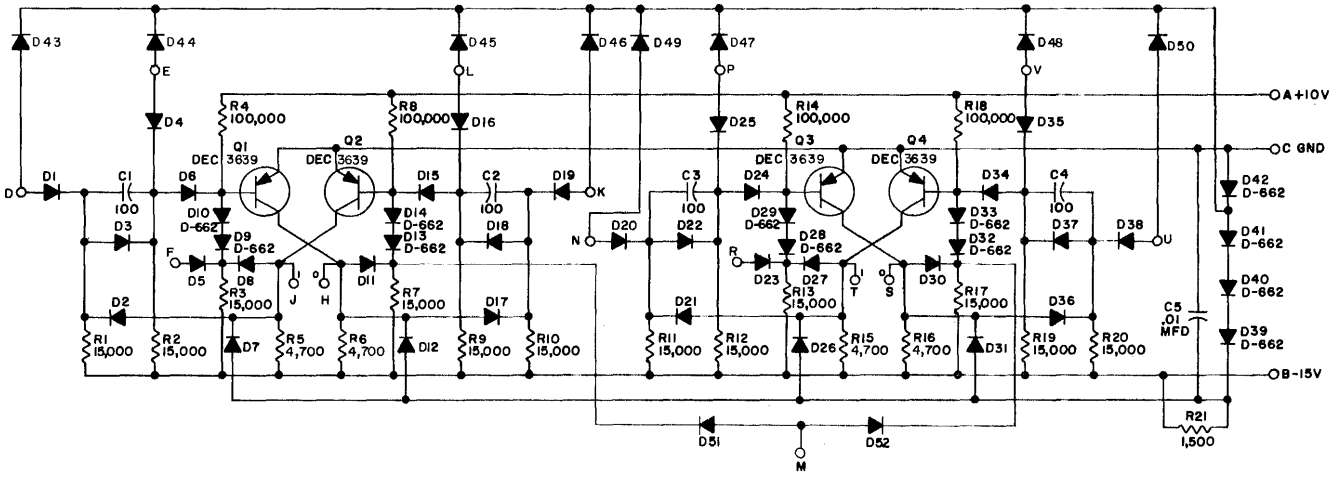
Diode Gate RS-B-R111



Binary-to-Octal Decoder RS-B-R151

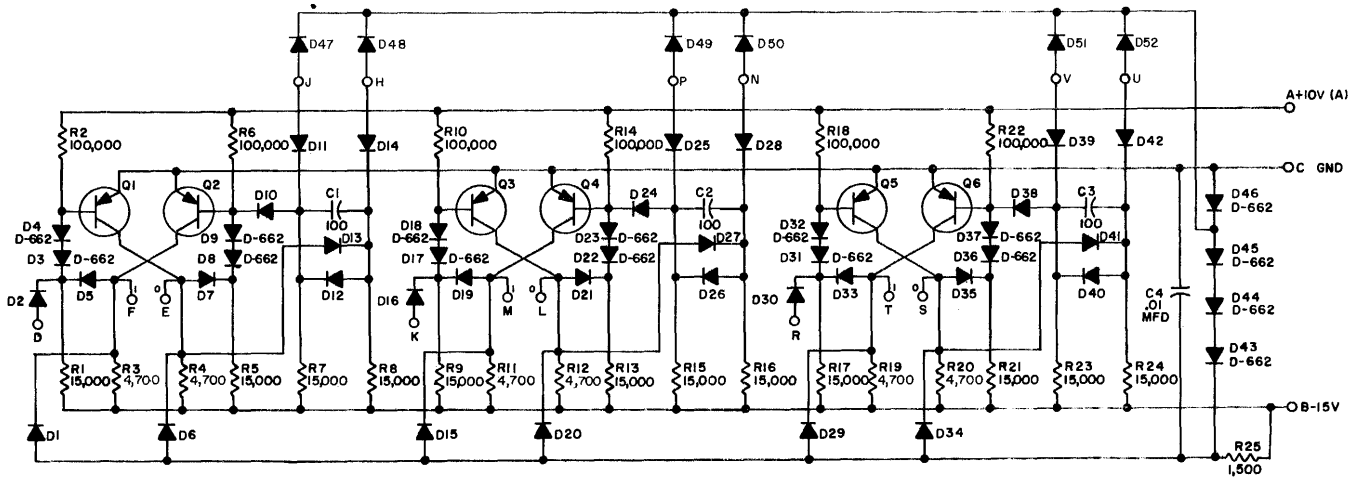


Flip-Flop RS-B-R201



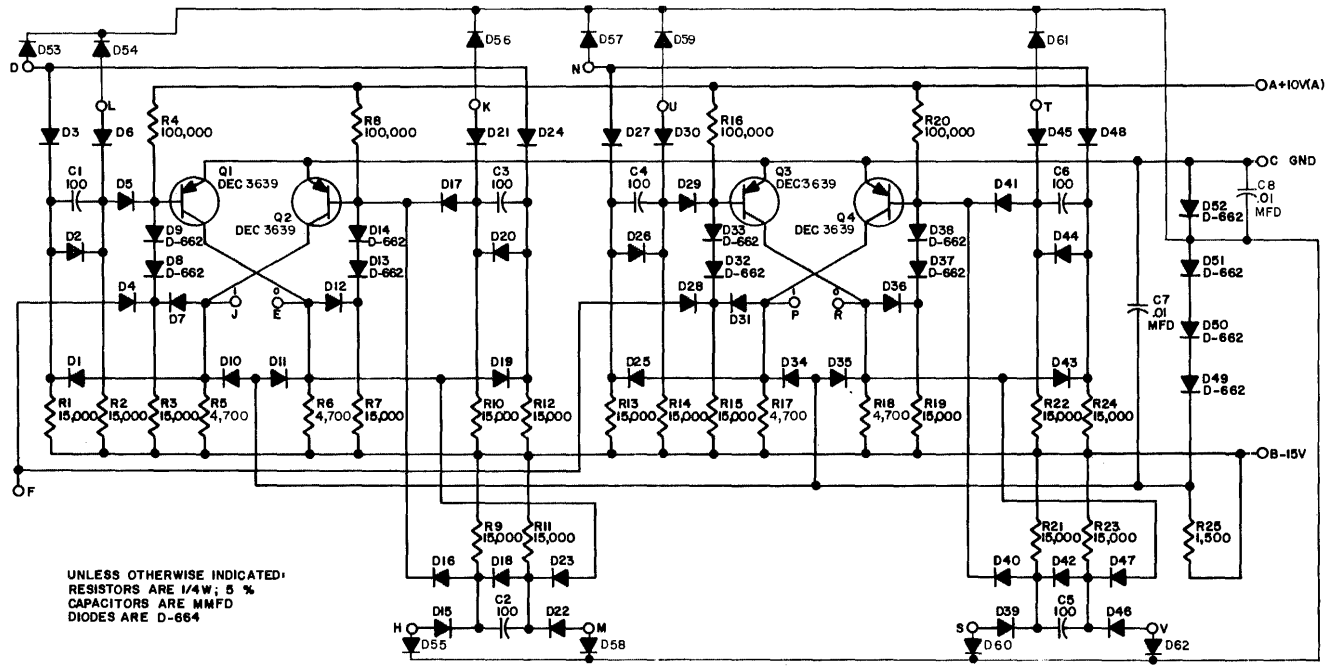
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W, 5%
CAPACITORS ARE MMFD
DIODES ARE D-664

Dual Flip-Flop RS-B-R202

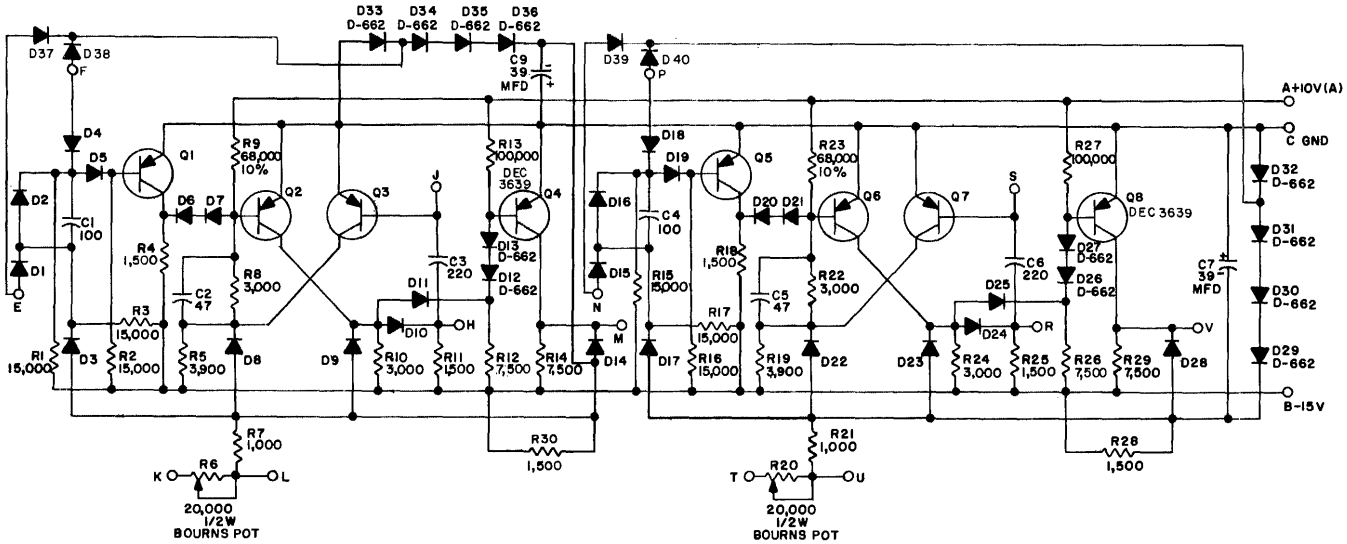


UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W, 5%
CAPACITORS ARE MMFD
DIODES ARE D-664
TRANSISTORS ARE DEC 3639

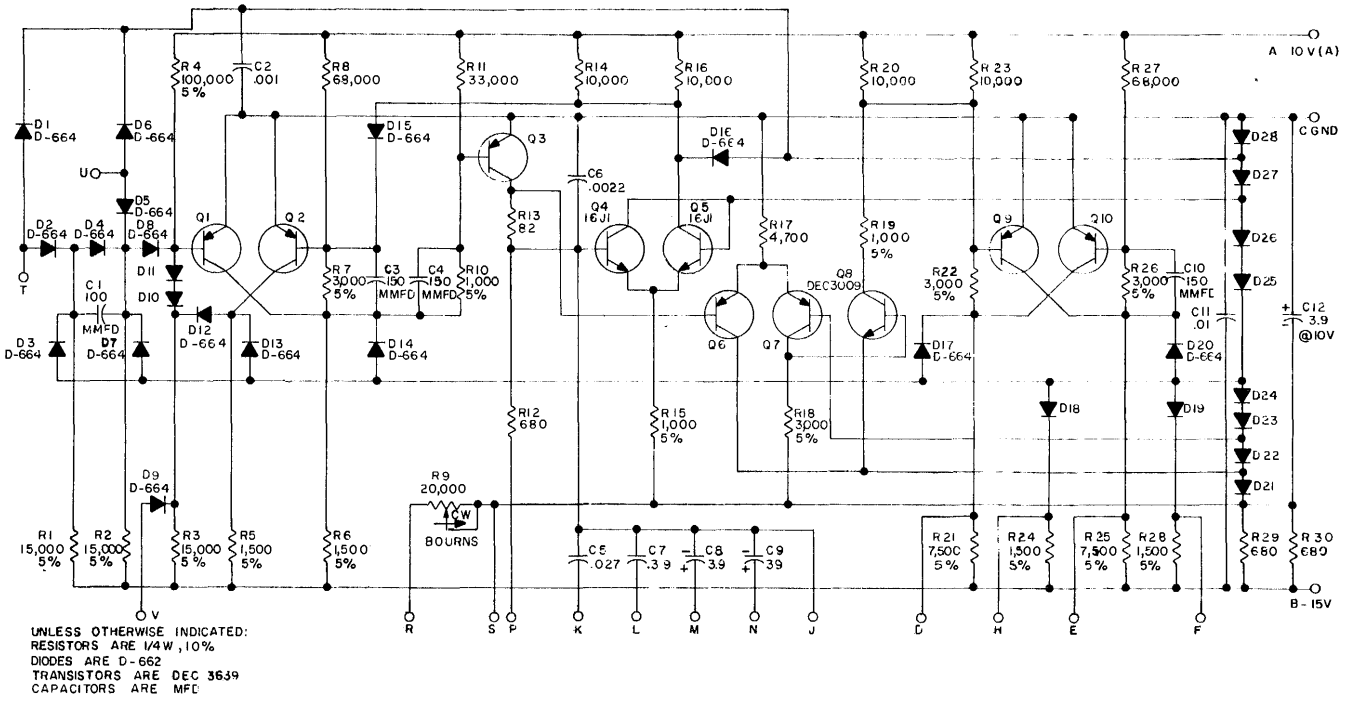
Triple Flip-Flop RS-B-R203



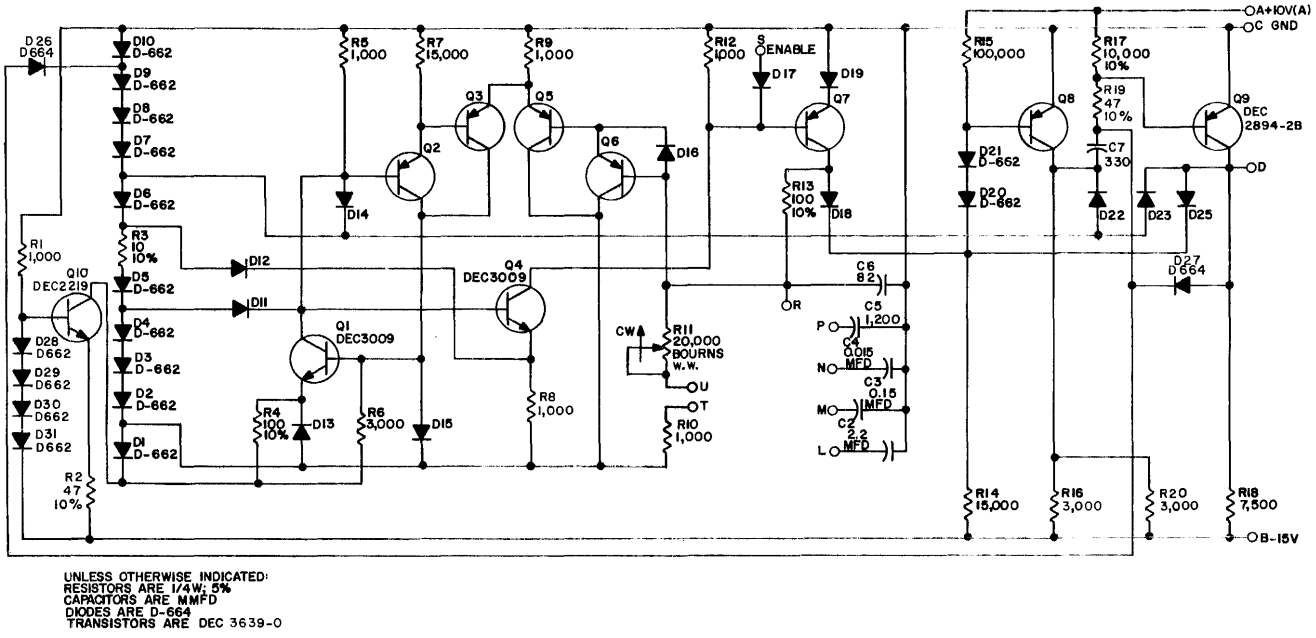
Dual Flip-Flop RS-B-R205



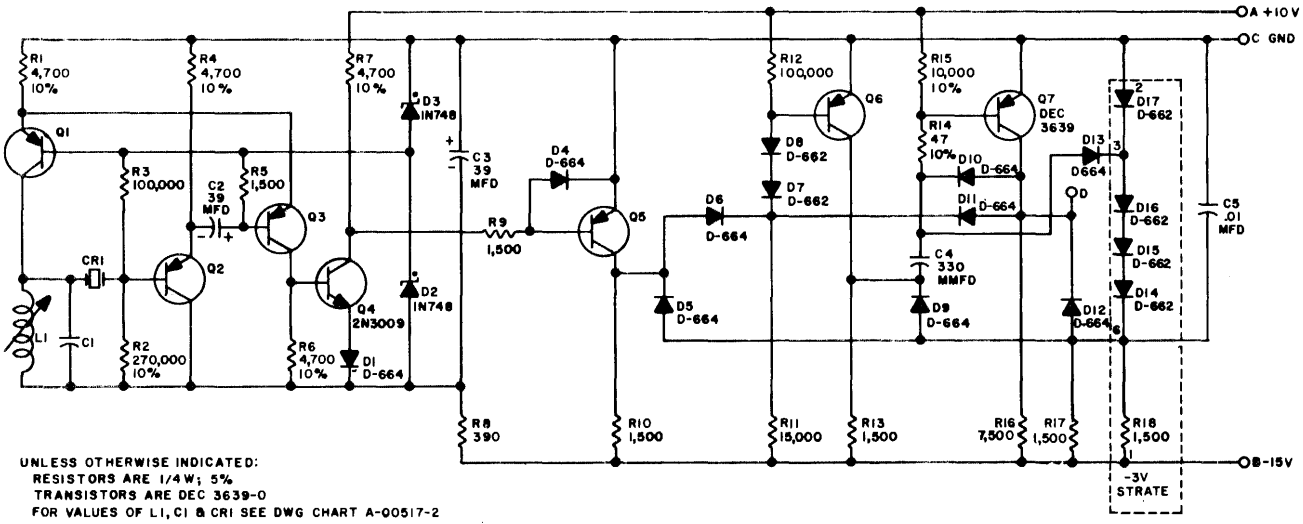
Delay RS-B-R302



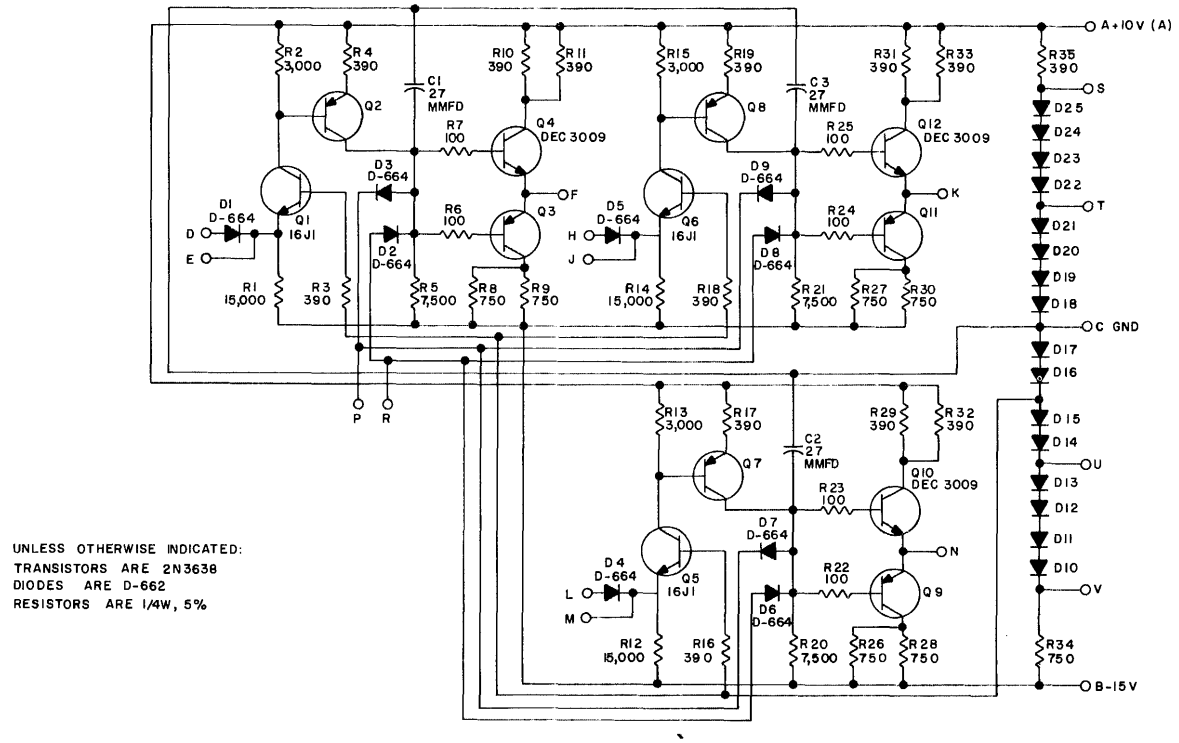
Integrating One Shot RS-B-R303



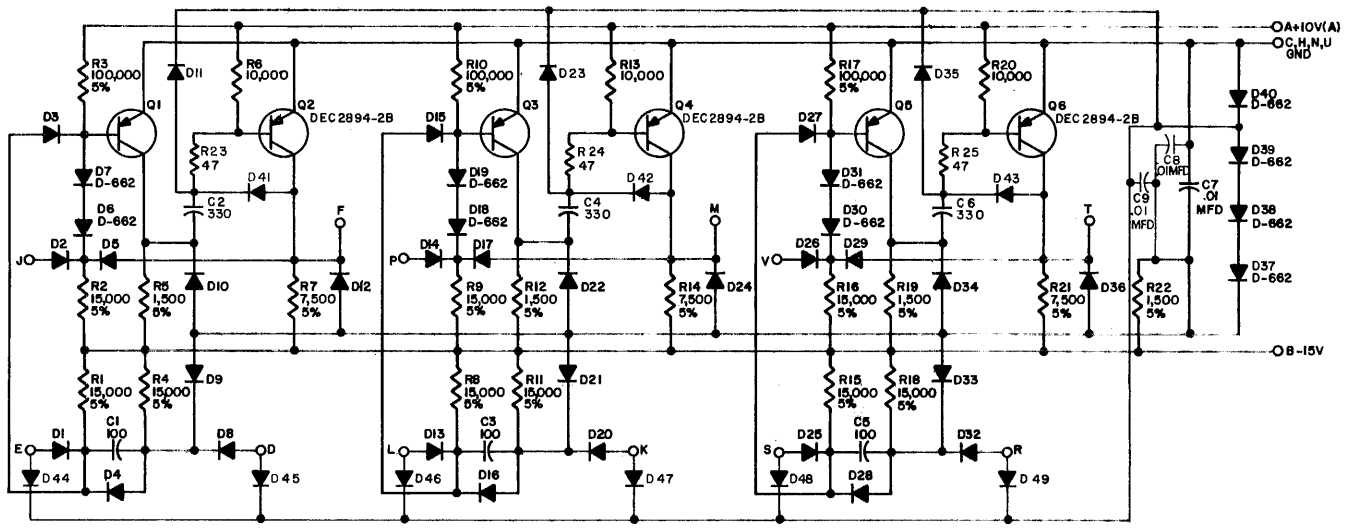
Clock RS-B-R401



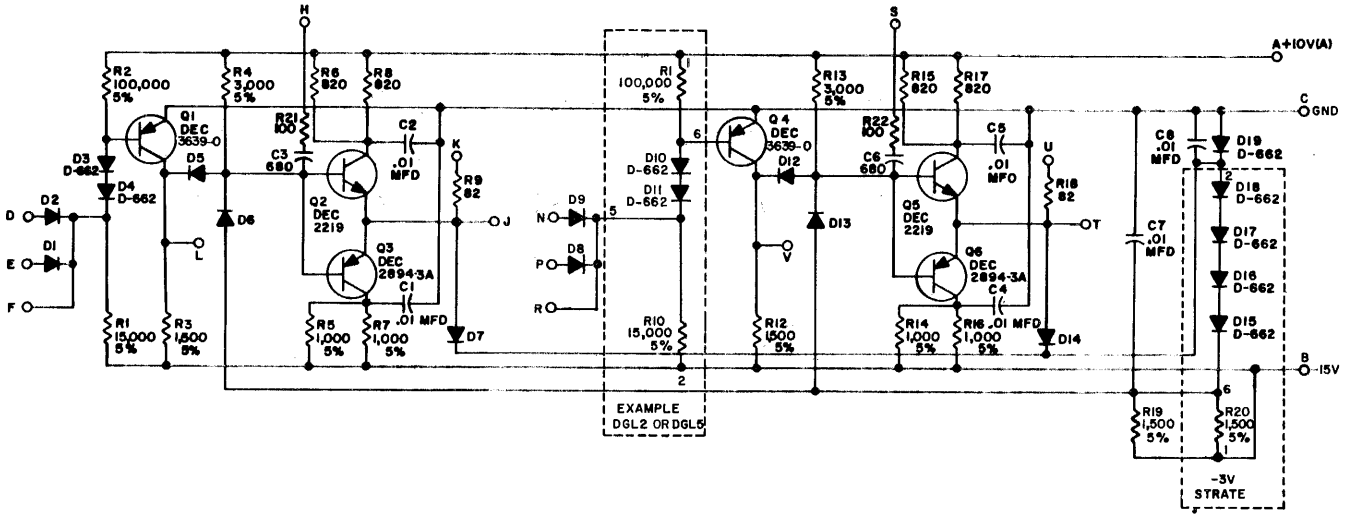
Crystal Clock RS-B-R405



Pulse Amplifier RS-B-R602

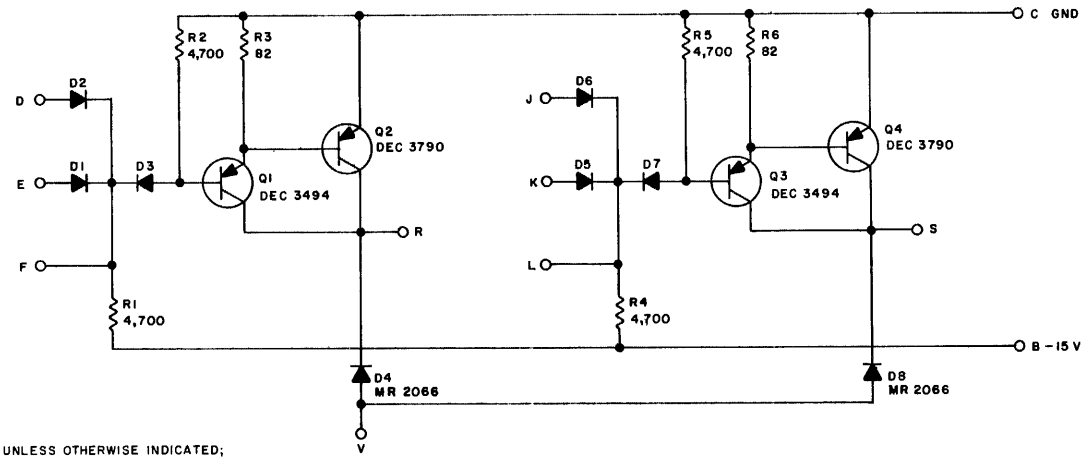


Pulse Amplifier RS-B-R603



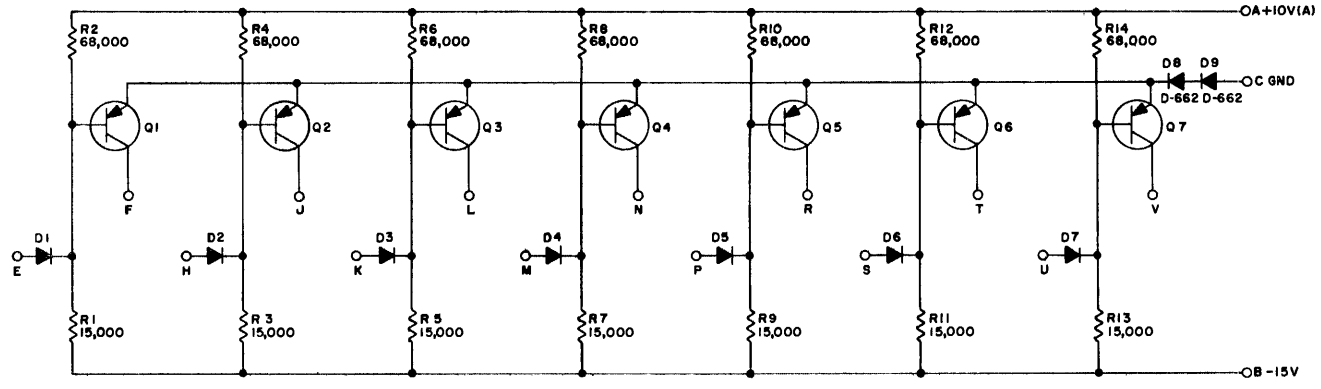
UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W, 10%
CAPACITORS ARE MMFD
DIODES ARE D-664
PRINTED CIRCUIT REV. FOR DGL BOARD IS SJA

Bus Driver RS-B-R650



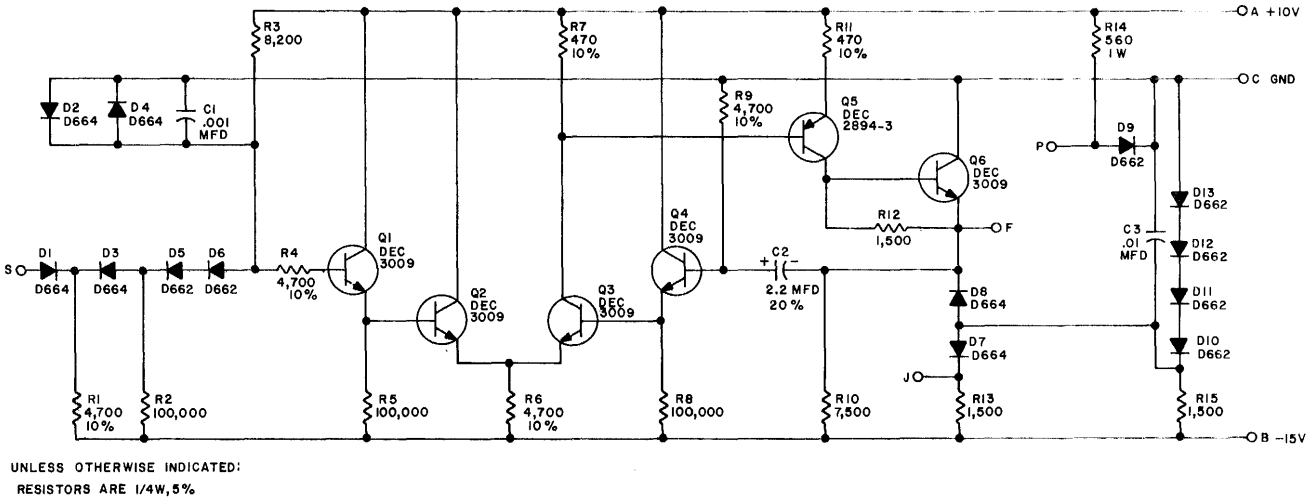
UNLESS OTHERWISE INDICATED;
RESISTORS ARE 1/4W; 10%
DIODES ARE D-664

Solenoid Driver RS-B-W040

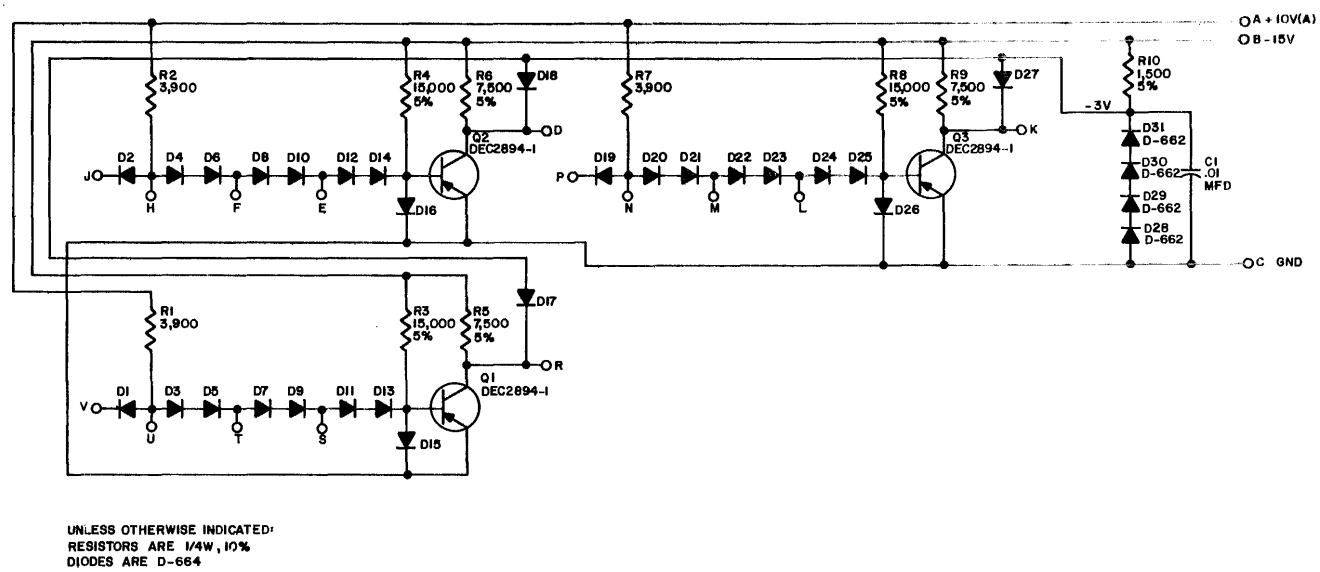


UNLESS OTHERWISE INDICATED;
RESISTORS ARE 1/4W; 10%
DIODES ARE D-664
TRANSISTORS ARE DEC 3638AB OR DEC 6534B

Indicator Driver RS-B-W050



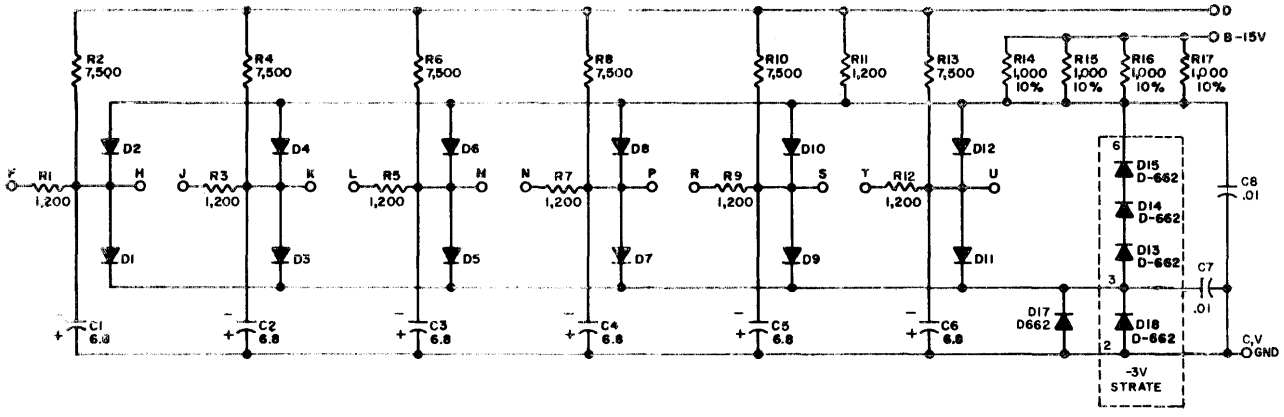
Initial Transient Detector RS-B-W504



Positive Level Converter RS-B-W510

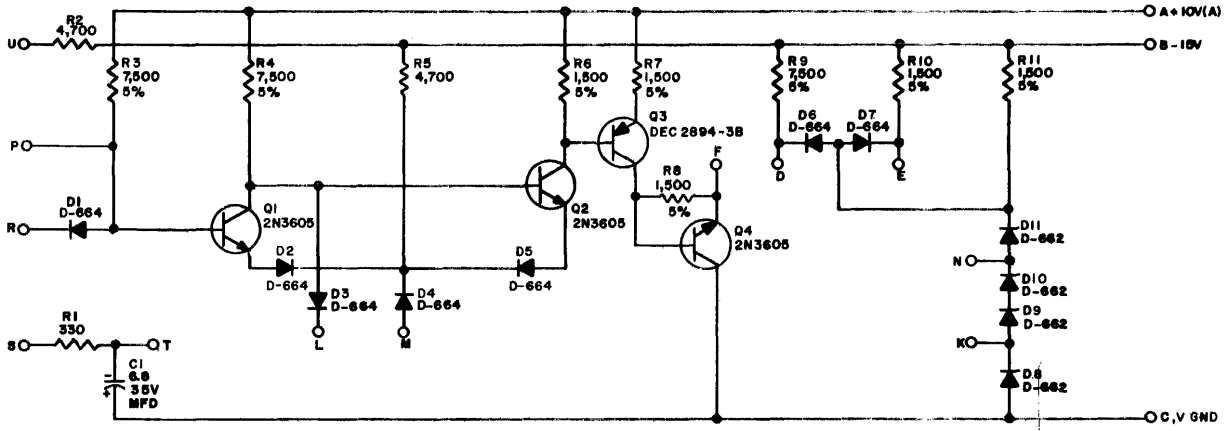
(To be supplied)

Bipolar Level Amplifier RS-B-W602



UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4 W, 5%
CAPACITORS ARE MFD
DIODES ARE D-664

Switch Filter RS-B-W700



UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%

Schmitt Trigger RS-B-W501

EQUIPMENT
CORPORATION

MAYNARD, MASSACHUSETTS

PRINTED IN U.S.A.

1-3/66

H-646

TYPE
646
LINE PRINTER

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

TYPE
646
LINE PRINTER

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

COPY NO.

PREFACE

This manual contains information on the theory of operation, installation, operation, and maintenance of the Line Printer Control Type 646. The line printer control is a program-controlled device which accepts data and controls from the PDP-6 general-purpose digital computing system via the I/O bus and sends printable characters and function signals to the 646 Line Printer at the printer acceptance rate. The control also informs the PDP-6 program of the line printer status. Chapter 1 of this manual presents information of a general nature. Chapter 2 explains the theory of operation for the control and its interface with the line printer. Chapters 3, 4, and 5 present information and procedures which enable personnel to install, operate, and maintain the control.

CONTENTS

CONTENTS (continued)

ILLUSTRATIONS

ENGINEERING DRAWINGS

ENGINEERING DRAWINGS (continued)

TABLES

CHAPTER 1

INTRODUCTION AND DESCRIPTION

PURPOSE OF EQUIPMENT

The Line Printer Control Type 646 is the program-controlled buffer interface between the Programmed Data Processor-6 (PDP-6) general-purpose digital computing system and the 646 Line Printer (series 5 ANelex line printer). Under program control, the line printer control accepts data from the I/O bus or transmits status indications to the PDP-6. Each data entry may include characters to be printed, control characters defining printout format, or illegal characters (codes not assigned to the line printer). Printable characters and format control signals are sent to the line printer at its acceptance rate and with printer-required signal characteristics. Illegal character codes are ignored. The PDP-6 program is free during each data transfer cycle to perform other routines. By means of the PDP-6 priority interrupt system, the line printer informs the program that its status should be interrogated. Status indications generated by the line printer consist of: done (ready for data entry), busy (line printer performing a program-directed function), printer error (line printer is off line or interface cable connection is not complete), and line overflow error (120-character capacity of line printer character buffer has been exceeded prior to the program sending a printout command through the line printer control).

DESCRIPTION

Functional Description

The Line Printer Control Type 646 receives signals from and sends signals to the PDP-6 system via the I/O bus. An interface cable conveys signals between the control and the line printer. The line printer control application diagram, Figure 1-1, shows the signal flow between the three.

All input/output devices in the PDP-6 system share the I/O bus. For this reason, a unique device selection code accompanies each program instruction to specify which device is to

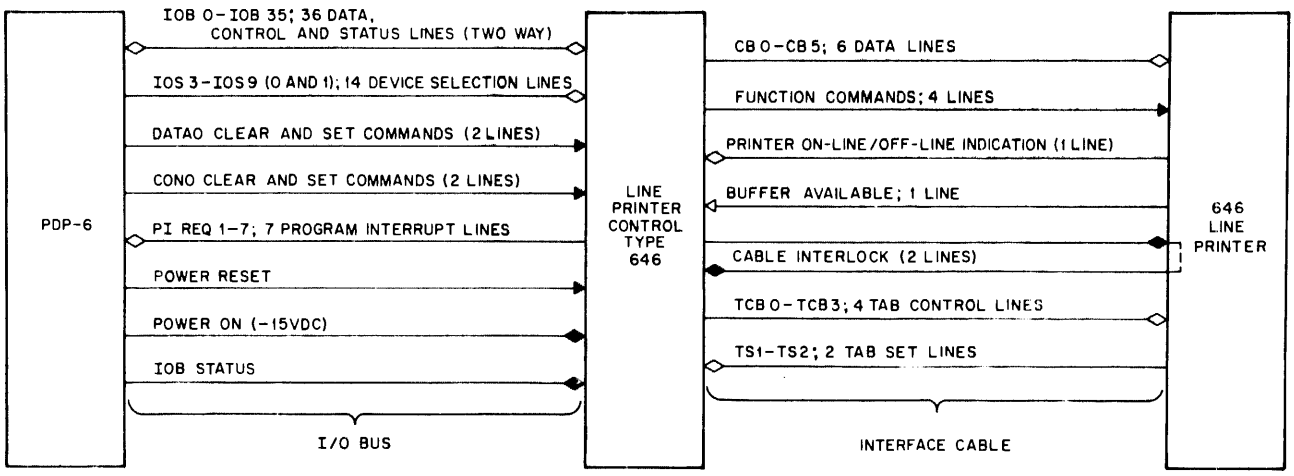


Figure 1-1 Line Printer Control, Application Diagram

respond. The device selection code is represented by seven pairs of complementary signals, with each pair containing one bit of the 7-bit code.

The input/output devices, or their respective controls, decode the 14 levels to determine if their code is present. The line printer code is 124_8. When the line printer control decodes a 001 010 1 from the IOS3-IOS9 lines, the program instruction, CONO (conditions out) or DATAO (data out), is accepted. Either instruction consists of a CLEAR pulse and a SET pulse. The CLEAR pulse readies the line printer control for the instruction; the SET pulse accomplishes the instruction.

A CONO instruction CLEAR pulse clears the line printer control. The following SET pulse enters the done and error priority interrupts in the control register; and if I/O bit 7 is 1, initiates clearing of the line printer control and the line printer character buffer. The control enters a busy status until the line printer sends a buffer available indication, which resets the busy flag and sets the done flag. The done interrupt is generated when the done flag is set. The error interrupt is generated if the line printer is off line, the interface cable is not connected, or if the program has overfilled the line printer character buffer. The IOB STATUS signal occurs with a conditioned CONI instruction (CONSO or CONSZ) to sample the line printer control status indications into the PDP-6 through the I/O bus.

A DATAO CLEAR pulse clears the character acceptance portion of the line printer control. The subsequent DATAO SET pulse samples five 7-bit ASCII-coded characters present on IOB0 through IOB35 lines into the line printer control. The done flag is reset and the busy flag is set. The five characters are processed in turn by the line printer control. Printable characters are sequentially sent to the line printer for entry in its character buffer. Control characters defining printout formatting are decoded within the line printer control to create signals which are sent to the line printer to accomplish the required function. The five-character entry may contain character codes not assigned to the line printer. The line printer control will detect these illegal codes and inhibit their passage to the line printer. Unless the five characters contained a STOP control character, completion of the data transfer cycle generates a done interrupt by setting the done flag. The program recognizes this as a request for another five-character loading. If a STOP control character is decoded, the busy flag is reset but the done flag is not set to generate the interrupt.

Once the line printer is on line and characters are loaded into its character buffer, the line printer cannot go off line until the characters are printed out or cleared out of the buffer. A program command can initiate the printout or clearing operation. Depression of the MANUAL PRINT control on the line printer also initiates the printout. If the program transfers more than 120 printable characters without a printout command or clearout command, the line printer locks in the busy state to prohibit further character transfers (these characters would be lost). A carriage return control character or a CONO program instruction must be sent to return the line printer control to operation.

Eleven control character inputs to the line printer control initiate operations within the control and the line printer. Two (horizontal tab and carriage return) control the horizontal format of the printout. Eight (form feed, line feed, vertical tab and DC0-DC4) control the vertical format of the printout. One (stop) halts line printer control operation. A summary of these characters and their respective functions follows.

Horizontal Tab - Performance of this control character is analogous to typewriter tabs. Upon decoding a horizontal tab control, the line printer loads space characters in the line printer character buffer until the tab setting is reached. The next printable character enters the

character buffer at this printing point. The tab setting is preselected by the line printer tab selection switch to be one of five increments (8, 10, 12, 16, or 20 columns).

Carriage Return - A decoded carriage return control character clears the line printer control column counter to enter the next character at column 1 in the line printer character buffer. This clearing returns the printout point to the extreme left-hand column. A carriage return can also initiate printout of characters stored in the buffer but cannot initiate a paper advance.

Form Feed - A form feed control character initiates a print and space command to the line printer. Characters previously entered in the buffer are printed out. The paper then advances to the top of the next page. Unless a carriage return occurs, the next character enters the character buffer at column N plus 1, where N equals the column count of the previous printout.

Line Feed - A line feed control character initiates a print and space command. After stored characters are printed out, the paper advances 20 lines. The next character entry in the buffer is at column N plus 1 (see Form Feed).

Vertical Tab - A vertical tab control character initiates the printout of stored characters and advances the paper 20 lines. The next character entry in the buffer is at column N plus 1 (see Form Feed).

DC0-DC4 - Five vertical tab control characters can initiate printout of stored characters. After the printout, the paper advances 30, 2, 3, 6, or 10 lines, respectively. The next character entry in the buffer is at column N plus 1 (see Form Feed).

Stop - Decoding of this control character halts line printer control and thus line printer operation.

There are 66 printing lines per page. A vertical format tape contained in the line printer ensures that the top two and the bottom two remain blank to leave 62 usable printing lines. Refer to the ANelex instruction manual for a description of the vertical format control.

Physical Description

The Line Printer Control Type 646 consists of 45 DEC system modules installed in three module panels of a standard computer cabinet. (See engineering drawing UML-D-646-0-11 in Appendix 1.) Usually, the control is not installed in a separate cabinet but in one housing additional input/output device controls or system logic for the PDP-6 general-purpose digital computing system. The actual location of the line printer control modules is indeterminate as the complexity of a purchased system determines space availability. The reader is advised to refer to engineering documentation accompanying the system for location and wiring information.

The standard computer cabinet has front and rear split doors. The front door permits access to the connector wiring for the module panels. The rear door and an internal door provide access to the cabinet power supply, the modules, and the eight I/O bus cables. The PDP-6 system I/O bus is serially connected to each input/output device or its control. Four of the eight I/O bus cables interface the line printer control with the I/O bus system. The other four leave the line printer control interface to connect some subsequent input/output device with the I/O bus. One Type 1032 Terminating Package (47 ohms) must be installed in connectors number 4 (Figure 3-1) if the line printer control is the last I/O device connected to the I/O bus cables.

Electrical Description

Standard DEC power supplies installed in the computer cabinet operate from line inputs of 115 vac to provide the module-required voltages of -15 vdc and +10 (A) vdc and +10 (B) vdc. Refer to the PDP-6 System Manual for descriptions of these power supplies and the loading imposed upon them by the line printer control modules.

Line Printer Description

The following reference data listing summarizes the principal characteristics of the 646 Line Printer (series 5 ANelex printer).

Physical Characteristics

Height:	52 to 57 in., adjustable
Length:	56 in.
Depth:	30 in.

Weight:	1350 lbs

Printing Characteristics

Width:	120 columns (characters) max. at 10/in.
Lines per inch:	6 max.
Lines per minute:	300 max.
Load character in buffer:	10 μsec
Paper advance:	18 msec/line
Printout of stored characters:	180 msec

Paper Media

Single copy:	20-lb wt
Multiple copy:	Paper 9 to 12 lb wt; high-speed carbon 5 to 7 lb wt
Width:	3 to 19 in.
Feed holes:	Diameter of 0.155 (+0.005; -0.000) in.; 0.500 (+0.010) in. center-to-center vertically; 0.250 (+0.010) in. center of holes to vertical edge of paper.

Ribbon Requirements

Multilith ribbon and roll assembly:	ANelex Part No. 66209

Power Requirements

Input:	115 volts, 60-cycle, 3KVA (obtained independently of the PDP-6 system)

Symbols and Terminology

Readers who are unfamiliar with the DEC conventions of symbols and terminology to represent logical functions are advised to refer to the symbols and terminology discussion in Appendix 1. Reference to the DEC System Modules Handbook (C-100) will amplify this discussion.

Pertinent Documents

Discussions in the subsequent sections of this manual assume that the reader has read and is familiar with the content of the following documents.

System Modules Handbook (C-100)
PDP-6 Maintenance Manual (F-67)
PDP-6 Handbook (F-65)
PDP-6 Installation Manual (F-68)
Series 5 ANelex Printer Instruction Manual

Abbreviations

The following lists those abbreviations used throughout this manual and their meaning.

ASCII	American Standard Code for Information Interchange
I/O	Input/Output
CONO	Conditions Out
DATAO	Data Out
PDP-6	Programmed Data Processor-6 system
LPT	Line Printer Control
IOB	Input/Output Bus

CHAPTER 2

THEORY OF OPERATION

OVERALL FUNCTIONAL DESCRIPTION

The Line Printer Control Type 646, once data is entered from the PDP-6 system, operates at the 646 Line Printer acceptance rate in transferring printable characters and commands to the printer character buffer or vertical format control section. A CONO instruction (Figure 2-1) enters the priority codes in the control register and with IOB7 1 clears the line printer control and line printer character buffer. The codes also appear in the status register along with status indications. The IOB STATUS interrogation signal (initiated by CONSO, CONSZ, or CONI) samples these indications through the I/O bus into the PDP-6. A DATAO instruction enters five 7-bit characters in the character input register and initiates processing of the first character. Each character is decoded in turn by the character decoding section. If illegal, the character is shifted out of the character input register and nothing is sent to the line printer. If legal, the character (printable or control) appears in the control buffer and a command is sent to the character transfer section to initiate the required action. This character is not shifted out of the character input register until the action is accomplished. The process continues until five characters have been shifted out of the register at which time a word request generates the done interrupt to request another five-character entry. If a stop character appears in an entry, the line printer control halts upon decoding it.

The error interrupt occurs to inform the PDP-6 that the line printer went off line, the interface cable became disconnected, or the program exceeded the 120-character capacity of the line printer buffer before sending a printout command.

The DATA STROBE command samples printable characters into the line printer character buffer. If a horizontal tab function is requested by the program, DATA STROBE samples space characters into the buffer until the tab stop is reached. Selection of the tab column increment is made through use of the line printer tab selection switch. DATA STROBE also enters space characters in the buffer when a carriage return does not occur to return the printout point to column 1. The space characters are entered until the last column of the previous printout is reached. The next character is then processed.

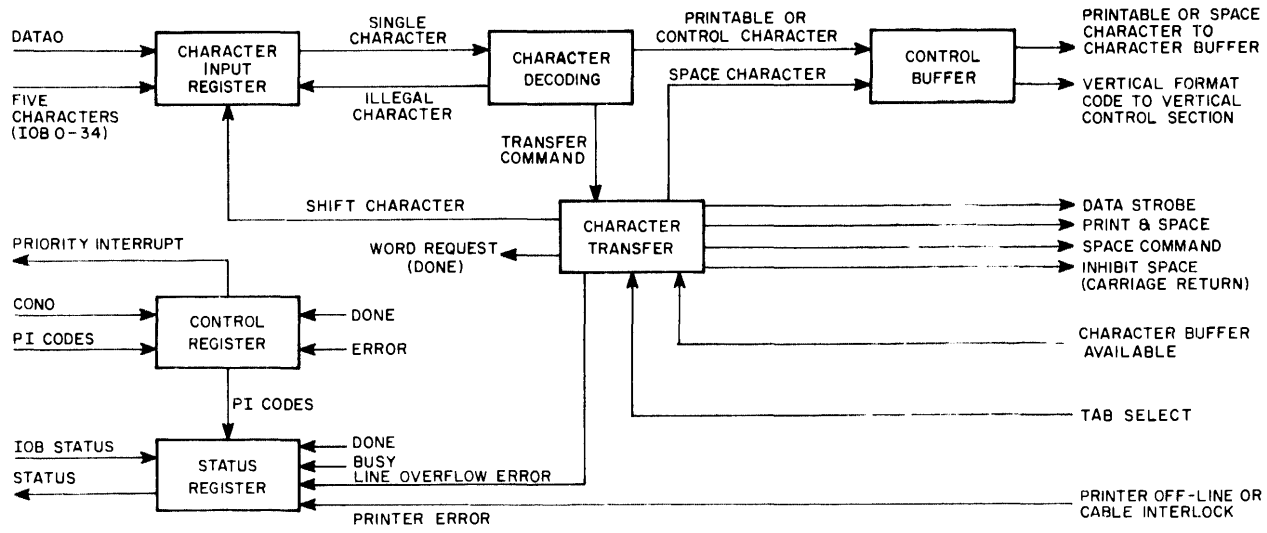


Figure 2-1 Line Printer Control, Block Diagram

The PRINT and SPACE command is used to print out stored characters and to advance the paper to the line specified by the decoded vertical format control character. If a carriage return function initiates PRINT and SPACE, the paper is inhibited from advancing.

The SPACE COMMAND advances the paper to the line specified by the decoded vertical format control character. This command occurs only when there are no characters in the line printer character buffer.

DETAILED DESCRIPTION

Logic circuits for the line printer control are grouped in five major functional sections. They are:

1. the interrupt and status logic,
2. the character input register,
3. the control character decoding circuit,
4. the character transfer circuit, and
5. the counters and tab selection circuit.

An engineering logic drawing for each functional logic section is contained in Appendix 1. Reference is made to these diagrams in the paragraphs following which describe the circuit

functions. These descriptions presume that the reader is familiar with the DEC logic signal and element symbology. Any reader not possessing this knowledge is advised to refer to the symbols and terminology portion of Appendix 1. Line printer control logic drawings refer to I/O bits; PDP-6 system drawings refer to the same as IOB bits.

Interrupt and Status Logic

The interrupt and status logic (diagram 646-0-6, Appendix 1) accepts a program instruction (CONO or DATAO) and accomplishes the required action only when the line printer device selection code (124_8) accompanies the instruction. Line printer selection is made when the IOS3-IOS9 lines provide seven ground level signals, representing 001 010 1, to a Type 4118 Diode Gate which decodes the levels to place LPT SEL at - 3 volts. The LPT SEL negative level goes to four Type 4606 Pulse Amplifiers and to one Type 4113 Diode Gate allowing a program instruction input to affect the required circuits.

A CONO CLR instruction pulse (Figure 2-2) input at pin D of I/O bus 4 generates a 0.4-microsecond pulse for LPT IC CLR. This LPT IC CLR pulse clears the control register (flip-flops PIA0, PIA1, PIA2, PIB0, PIB1, and PIB2) and the DONE, BUSY, and CLEAR flip-flops. The IOB PWR RES (power reset) negative pulse also generates the LPT IC CLR pulse when the system power is turned on or when the I/O RESET key on the computer console is depressed.

Approximately 1 microsecond after the CONO CLR pulse, the CONO SET pulse (Figure 2-2) input to pin E of I/O bus 4 generates the 0.4-microsecond LPT IC SET pulse which enters I/O bits 15, 16, and 17 in the PIA (priority interrupt) protion of the control register and I/O bits 12, 13, and 14 in the PIB portion. The PIA and PIB 3-bit codes reflect the priority interrupt code for the line printer control. The PIA code is not released to the PDP-6 unless the LPT DONE FLAG applies a ground input to pin P of Type 4151 Binary-to-Octal Decoder. The PIB code is not sent to the PDP-6 unless a PRINTER ERROR or a LINE OVERFLOW condition occurs. The LP IC SET pulse sets the BUSY flip-flop if IO-10 is at ground (1) or the DONE flip-flop if IO-11 is at ground (1).

The IOB STATUS signal is negative during a CONSO, CONSZ, or CONI instruction input. Coupled with the LPT SEL level, IOB STATUS samples the information bits in the line printer status register (ten Type 4657 Bus Drivers) into the PDP-6 I/O bus system.

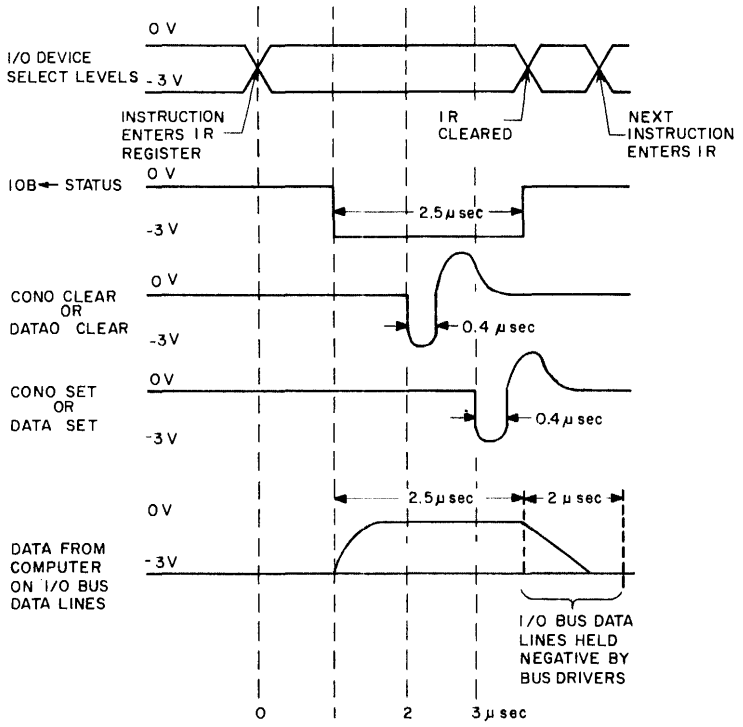


Figure 2-2 PDP-6 I/O Bus Timing Diagram

If I/O bit 7 is 1, the CONO SET pulse sets the CLEAR flip-flop producing a -3 ground transition for the LPT CLEAR signal. This transition is applied to a 4606 Pulse Amplifier to produce the LPT DATA CLR pulse (B-2) which clears the character input register and to a 4606 Pulse Amplifier in the character transfer circuit to initiate clearing of the line printer character buffer. The LPT CLEAR pulse also sets the busy flag. When the line printer indicates completion of character buffer clearing, the busy flag is reset and the done flag is set by the WORD REQUEST input from the character transfer circuit.

A DATAO CLR pulse input (Figure 2-2) at pin B of I/O bus 4 generates the LPT DATA CLR pulse which clears the character input register. Approximately 1 microsecond after DATAO CLR, a DATAO SET pulse generates the LPT DATA SET pulse which sets the busy flag, resets the done flag and samples five 7-bit characters into the character input register through IOB0-IOB34 lines. Thereafter, a WORD REQUEST input at the end of each five-character transfer cycle sets the done flag to generate the done priority interrupt and resets the busy flag. This WORD REQUEST initiation of the done interrupt continues until an EOT control character is entered in the character input register by the PDP-6 program. Decoding of this control character places STOP at a negative level allowing a COLUMN COUNT pulse to reset the busy flag without setting the done flag to generate the done priority interrupt to request additional character inputs.

Character Input Register

The character input register (diagram 646-0-9, Appendix 1) consists of seven Type 4221 Shift Register modules. Five 7-bit ASCII-coded characters can be simultaneously transferred from the PDP-6 system to the register modules through I/O bit lines 0-34. Bit 1 of the first character is stored in the first stage (W input) of module CI-0 (A-3); bit 2 is stored in the first stage (W input) of module CI-1 (A-5); bit 3 is stored in module CI-2; bit 4 is stored in module CI-3; etc. In like fashion, the other four characters are stored in the second, third, fourth, and fifth stages of the seven modules. The sixth, or left-most, stage of each register module does not receive data and, further, is clamped in the 0 state by the constant INHIBIT level resulting from the -15 vdc input at pins E and R of module CI-0 (B-1).

Three signals control operation of the character input register. They are: LPT (line printer) DATA SET, LPT DATA CLR, and SHIFT CHARACTER. The LPT signals are generated in the interrupt and status logic circuit as a result of I/O instructions from the PDP-6 processor. The SHIFT CHARACTER input is developed in the character transfer circuit. A CONO instruction with I/O bit 7 a 1 generates a 0.4-microsecond pulse for LPT DATA CLR, which applied to pin Z of each register module, clears the character input register. A DATAO instruction generates a 0.4-microsecond pulse for the LPT DATA SET input to pin Y of each register module. Any module stage having a ground level (logical 1) at its respective I/O bit input is switched by the LPT DATA SET pulse to the 1 state. All module stages not having an I/O bit input at ground remain in the 0 state. The characters entered in the character input register can be printable characters, control characters (defining machine functions requested of the line printer by the PDP-6), or illegal characters (codes not assigned to the line printer).

The I and O side outputs of the first stage in each register module (pins N and P, respectively) go to the control character decoding circuit. This circuit decodes the bit-content of the character to determine if it is legal or illegal; and if legal, what action is requested. A legal character is one in which bits 1 and 2 complement each other. Conversely, an illegal character has bits 1 and 2 at the same logical level.

After LPT DATA SET enters the five 7-bit characters in the character input register, a 0.4-microsecond pulse is generated in the character transfer circuit for the SHIFT CHARACTER input to pin X of each register module. This pulse shifts the five characters to the right.

Character 1 is dumped out of the register and replaced in the first stage of each module by character 2. If character 1 was legal, the required action (transfer of character code or machine function command to line printer) was accomplished prior to the SHIFT CHARACTER pulse. If the character was illegal, data is not transferred to the line printer and the character is simply shifted out of the character input register.

Thereafter, while the line printer indicates that its character buffer is available for data entries and a printable character or legal control character is in the character 1 position of the character input register, the SHIFT CHARACTER pulse occurs until the five 7-bit characters have been shifted out of the register. During the interim between SHIFT REGISTER pulses, the bit-content of each character is examined to determine its legality and the action required. With the shift of the fifth character out of the register, the done priority interrupt occurs to request an additional five-character entry. A decoded EOT (stop) control character inhibits generation of the done priority interrupt and thus halts operation of the line printer control.

Character Counter and Decoder

The character counter (diagram 606-0-9, Appendix 1) consists of a Type 4215 Complementing Flip-Flop module (C-7, 8), a Type 4606 Pulse Amplifier, and a Type 4606 Pulse Amplifier. The decoder includes two Type 4112 Negative-Diode NORs and a Type 4102 Inverter module. The counter circuit counts SHIFT CHARACTER pulses to produce outputs which the decoder employs to indicate when the five 7-bit characters have been shifted out of the character input register. The counter also, through its CHARACTER COUNT pulse output, initiates the four SHIFT CHARACTER pulses after the first one is initiated by the LPT DATA SET pulse input to the character transfer circuit.

The LPT DATA CLR 0.4-microsecond pulse which clears the character input register, also by triggering the 4606 Pulse Amplifier, clears the character counter. The pulse amplifier delivers a 1.0-microsecond pulse to the flip-flop module, which places the three stages in the 0 state. Thereafter, each SHIFT CHARACTER 0.4-microsecond pulse input to pin U of Pulse Amplifier 4606 produces a 0.4-microsecond pulse output. The counter has a modulus 5 configuration. The first pulse switches flip-flop 5-2 to the 1 state. The second pulse complements 5-2 and switches 5-1 to 1. The third pulse again complements 5-2. Flip-flop 5-1 remains in the 1

state. The fourth pulse complements 5-2 and 5-1 and switches 5-1 to 1. The fifth pulse complements 5-0. The following table summarizes the response of the counter to the pulse inputs.

TABLE 2-1 COUNTER RESPONSES

Flip-flop	5-0	5-1	5-2
LPT DATA CLR	0	0	0
SHIFT CHARACTER - 1	0	0	1
SHIFT CHARACTER - 2	0	1	0
SHIFT CHARACTER - 3	0	1	1
SHIFT CHARACTER - 4	1	0	0
SHIFT CHARACTER - 5	0	0	0

The 0 side outputs of the three flip-flops go to the pin X-Y negative diode NOR and the inverter module. When the character counter represents an all zero condition due to the LPT DATA CLR pulse or the counting of five SHIFT CHARACTER pulses, the WORD TRANSFER COMP (complete) output is at - 3 volts. This level initiates a word request that sets the done flag and clears the busy flag in the interrupt and status logic circuit. While the counter is decoding SHIFT CHARACTER pulses, one of the three inputs to the decoder is 1 (see above). This condition places WORD TRANSFER COMP at ground and $\overline{\text{WORD TRANSFER COMP}}$ at - 3 volts. The latter level enables generation of the SHIFT CHARACTER pulses by the character transfer circuit.

Either a horizontal tab operation (011) input at pin M or a form advance input to pin N of the other negative diode gate also produces the enabling - 3 volt level for $\overline{\text{WORD TRANSFER COMP}}$ and the ground inhibiting level for WORD TRANSFER COMP.

Horizontal Tab Selection

The horizontal tab selection circuit (diagram 646-0-5, Appendix 1) functions in association with the line printer tab selection switch to set up preselected tab column increments when the program specifies a horizontal tab control character. The two-pole, five-position switch

is located within the right-hand door of the line printer on the test panel. The circuit includes a Type 4215 Complementing Flip-Flop module, a Type 4606 Pulse Amplifier, a Type 4114 Diode Gate, and a Type 6102 Inverter module.

A CLEAR COLUMN COUNTER pulse input to the pulse amplifier clears the flip-flops. Thereafter, the flip-flops are cleared by a COUNT COLUMN pulse while the TS-1 and TS-2 inputs to the 4114 Diode Gate are simultaneously at ground levels. The TS-1 and TS-2 inputs come from the tab selection switch in the line printer and become grounded as a result of the switch position and the counting of columns by the 4215 Flip-Flops. The flip-flops are stepped by each transition to ground of the 0 side output of column counter stage CC6 (A-5). These transitions occur when CC6 switches from the 1 state to the 0 state at alternate column counts.

The five tab increments are controlled as follows:

Tab 8 — TS-1 is grounded when TC-1 is switched to the 1 state by the fourth CC6 pulse input, which is equivalent to eight columns. TS-2 is grounded through an internal switch connection.

Tab 10 — TS-1, as for tab 8, grounds at the fourth CC6 pulse input. TS-2 is grounded by switching TC-3 to the 1 state at the fifth CC6 pulse input.

Tab 12 — TS-1 is grounded at the fourth CC6 pulse input. TS-2 is grounded by the switching of TC-2 to the 1 state at the sixth CC6 pulse input.

Tab 16 — TS-1 is grounded at the switching of TC-0 to the 1 state at the eighth CC6 pulse input. TS-2, as before, grounds through an internal switch connection.

Tab 20 — TS-1 grounds at the eighth CC6 pulse input. TS-2 grounds at the tenth CC6 pulse input.

Character Transfer

The character transfer circuit (diagram 646-0-8, Appendix 1) synchronizes operations within the line printer control and generates command signals which initiate functions requested of the line printer by the PDP-6 control character entered in the character input register. The requested functions include: clear the character buffer, enter the control buffer output data in the character buffer or in the vertical format control section of the line printer, and print out characters stored in the character buffer and advance the paper to the line specified by the control character input in the vertical format control section.

Additionally, the character transfer circuit controls the loading of space characters in the line printer character buffer when a horizontal tab control character is in the character input register or when a previous printout of stored characters did not fill the 120-character line and the next printout is to appear below and to the right of the last printed line. When the control character decoding circuit decodes a horizontal tab control character, space characters are entered in the line printer character buffer until the next preselected tab stop is reached. This operation is analogous to typewriter tabbing in that if a tab stop is passed, a printable character appears at the next tab stop after a tab control loads the intervening line increment with spaces. If a vertical format control character, rather than a carriage return control character, initiates the printout of characters stored in the character buffer, space characters equal to the number of columns printed out are entered in the character buffer before the next character is processed. For example: assume that a line feed control character has initiated the printout of 60 columns and advanced the paper one line. Before the next character is processed by the line printer control, 60 space characters are entered in the character buffer. The next character, if printable, enters the sixty-first column point of the buffer.

Printer Commands

One of the PAS (print and space), ESC (enable space command) and EDS (enable data strobe) signals (Diagram 646-0-8, Appendix 1) is negative to enable generation of its respective printer command signal during the processing of each legal character in the character input register. The type of character (printable or control) and the printout format (previous line and present line) determine which signal is negative.

The DATA STROBE negative pulse samples the CB0-CB5 output of the control buffer (diagram 646-0-7, Appendix 1) into the line printer character buffer. The EDS signal is negative to allow generation of the DATA STROBE negative pulse if:

1. the character is printable, or
2. the character is a horizontal tab control (011), in which case the CB0-CB5 output contains a space character, or
3. a carriage return control (015) was not sent to the line printer control to return the printout point to the extreme left-hand column.

The PRINT and SPACE negative pulse initiates the printout of characters stored in the line printer character buffer and, if the character in process is not a carriage return control (015), samples the CB0-CB5 output representing a vertical format code into the line printer vertical control section. After the printout is completed, the paper advances to the line specified by the vertical format code. If the character was a carriage return control (015), the paper does not advance. The PAS signal is negative to allow generation of the PRINT and SPACE negative pulse if:

1. the character is a carriage return control (015), or
2. the character is a vertical format control (form feed, line feed, vertical tab, or DC0-DC4) and at least one character has been entered in the line printer character buffer; in other words, a form (paper) advance and not column 1.

The SPACE COMMAND negative pulse samples the CB0-CB5 control buffer output into the line printer vertical format control section. The paper advances to the line specified by the control character vertical format code. The ESC signal is negative to allow generation of SPACE COMMAND when the character is a control character specifying vertical paper motion (form advance) and characters have not been entered in the line printer character buffer.

Clear Control

A CONO instruction with I/O bit 7 a 1 produces a - 3 volt-to-ground transition for LPT CLEAR in the interrupt and status logic. This positive transition applied to Type 4604 Pulse Amplifier (C-2) generates a negative CPB (clear printer buffer) pulse to initiate clearing of the line printer character buffer. The LPT CLEAR transition also develops the CLEAR negative pulse

which, in turn, develops the positive CLEAR SPACE COUNTER and CLEAR COLUMN COUNTER pulses. Both counters (diagram 646-0-5, Appendix 1) are cleared of previous space and column counts. After the line printer character buffer is cleared, the BUFFER AVAILABLE positive pulse through two Type 4606 Pulse Amplifiers generates a negative pulse for WORD REQUEST (WORD TRANSFER COMPLETE is negative at this time) which goes to the interrupt and status logic to reset the busy flag and set the done flag. The latter initiates the done priority interrupt to request a five-character input from the program. The BUFFER AVAILABLE pulse occurs after every line printer function.

Character Transfer Cycle

A DATAO instruction enters five 7-bit characters in the character input register by generating a LPT DATA SET negative pulse. This same pulse also comes to a Type 4301 One-Shot (A-2, diagram 646-0-8, Appendix 1) to initiate the character transfer cycle. During the 2.5-microsecond delay interval, the first character is checked by the control character decoding circuit to determine if it is legal or illegal, and if legal whether it is a printable or control character. To perform the operations involved in a character transfer cycle for an illegal character, a printable character, and a control character, the following paragraphs describe the use of the character transfer circuit (diagram 646-0-8, Appendix 1).

The CHARTSYN (character transfer synchronizing) pulse output of Type 4606 Pulse Amplifier (A-14) is the primary control signal in all character transfer cycles. The initial CHARTSYN pulse occurs approximately 3 microseconds after the LPT DATA SET pulse. Thereafter, while WORD TRANSFER COMPLETE is negative, either BUFFER AVAILABLE or CHARACTER STROBE generates the CHARTSYN pulse. Each CHARTSYN pulse performs three functions.

1. It initiates generation of a printer command signal if character decoding enables one of the three gates.

2. It initiates generation of a CLEAR COLUMN COUNTER pulse if the character is decoded as a carriage return control (015 at negative).

3. It, after the 2.5-microsecond delay of Type 4301 One-Shot (B-4), generates a COLUMN COUNT pulse.

Detection of an illegal character by the control character decoding circuit places ILLEGAL CHARACTER (B-7) at the 0-volt level. Because this illegal character does not represent a function required of the line printer, the CHARTSYN pulse cannot generate any of the printer command signals. It does generate a COLUMN COUNT pulse which is gated through Type 4217 Inverter Gate (output pin T) to generate the SHIFT CHARACTER pulse. The inverter gate is enabled since the character is neither a horizontal tab control character (011) nor a vertical format control character thereby placing the $\overline{011}$ and $\overline{\text{FORM ADV}}$ signals at 0-volt levels to produce the required negative level at pin S of the gate. The SHIFT CHARACTER pulse goes to the character input register and the character counter (diagram 646-0-9, Appendix 1) to shift the illegal character out of the register and advance the character counter one count through generation of a CHARACTER COUNT pulse. This CHARACTER COUNT pulse also comes to the Type 4303 Integrating One-Shot (diagram 646-0-8, Appendix 1) which develops the CHARACTER STROBE signal (B-8). If the character shifted into position 1 of the character input register is also decoded as being illegal, the ILLEGAL CHARACTER level remains at 0 volts allowing the CHARACTER COUNT pulse to tirgger the one-shot. The CHARACTER STROBE level increases from - 3 volts to 0 volts and remains 0 volts for 6 microseconds. At the end of this interval, CHARACTER STROBE decreases back to - 3 volts. This negative transition triggers Type 4606 Pulse Amplifier (A-2) to initiate generation of a CHARTSYN pulse if $\overline{\text{WORD TRANSFER COMPLETE}}$ is negative. The CHARTSYN pulse produces the COLUMN COUNT pulse to again generate SHIFT CHARACTER. The operation continues until either a legal character is decoded or the word (five characters) transfer is completed.

A printable character transfer cycle follows. The LPT DATA SET pulse of a BUFFER AVAILABLE pulse initiates generation of a CHARTSYN pulse which in turn produces the DATA STROBE pulse (EDS - enable data strobe is negative) and a COLUMN COUNT pulse. The DATA STROBE pulse samples the control buffer CB0-CB5 output into the line printer character buffer. The COLUMN COUNT pulse through inverter gate (output pin T) generates SHIFT CHARACTER. The pulse also steps the column counter one count. The SHIFT CHARACTER pulse shifts the character out of the character input register and generates a CHARACTER COUNT pulse to step the character count (diagram 646-0-9, Appendix 1). The cycles continue in the same manner until a printable character is not decoded or the word transfer is completed.

The operations involved in character transfer cycles for control characters depend on the type of control character being processed.

A horizontal tab control character (011) results in the following circuit operations for a character transfer cycle. Decoding of the horizontal control character places CB0 of the control buffer at 0 volts and CB1 through CB5 at - 3 volts; these six bits represent a space character. A BUFFER AVAILABLE pulse or the LPT DATA SET pulse generates the CHARTSYN pulse. Because of the EDS negative level, this pulse initiates generation of the DATA STROBE which samples the CB0-CB5 content into the line printer character buffer. The COLUMN COUNT pulse resulting from the CHARTSYN pulse is inhibited from passage through the inverter gates to initiate a SHIFT CHARACTER pulse. The BUFFER AVAILABLE pulse indicating receipt of the CB0-CB5 data initiates generation of a CHARTSYN pulse if the horizontal tab character was not the last character of the five-character entry in the character input register ($\overline{\text{WORD TRANSFER COMPLETE}}$ negative). The entry of space characters in the line printer character buffer continues until the TS-1 and TS-2 inputs to Type 4114 Diode Gate (D-1) are simultaneously at 0 volts. The resulting negative level for TAB STOP enables the Type 4217 Inverter Gate (output pin L) to pass the COLUMN COUNT pulse to initiate a SHIFT CHARACTER pulse that shifts the horizontal tab control character out of the character input register. Thus, when the TAB STOP column point is reached, space characters have been entered between the last printable character and the column point for the next printable character entry. The column counter records the number of columns used since each COLUMN COUNT pulse steps the counter.

A vertical format control character which occurs at column 1 (characters not entered in character buffer) has a character transfer cycle similar to that of a printable character. The CHARTSYN pulse, produced by either LPT DATA SET or BUFFER AVAILABLE, initiates generation of the SPACE COMMAND pulse. The resulting COLUMN COUNT pulse through output pin T inverter gate initiates generation of a SHIFT CHARACTER pulse which shifts the vertical format control character out of the character register. The column counter does not advance as the EDS signal remains negative and inhibits the COLUMN COUNT pulse from stepping the counter (diagram 646-0-5, Appendix 1).

A vertical format control character transfer cycle after characters (printable or space) have been entered in the line printer character buffer (column count more than one) follows. The

CHARTSYN pulse initiates generation of PRINT and SPACE. The vertical format code is sampled into the vertical control section of the line printer to advance the paper to the specified line. The CHARTSYN pulse also produces the COLUMN COUNT pulse. This pulse cannot generate SHIFT CHARACTER as all three inverter gates are inhibited. The output pin T gate is inhibited since a form advance occurred and the column count exceeded one. The output pin L gate is inhibited since a horizontal tab control character is not being processed. The output pin H gate is inhibited since LOAD SPACE and EQUAL COUNT are not both at 0 volts. The FORM ADV · $\overline{1 \text{ COUNT}}$ level is negative to allow COLUMN COUNT to set the LOAD SPACE flip-flop (B-7). After the characters stored in the line printer character buffer are printed out and the paper advances to the specified line, BUFFER AVAILABLE initiates generation of a CHARTSYN pulse. The CHARTSYN pulse initiates generation of DATA STROBE which samples the CB0-CB5 content representing a space character into the character buffer. The CHARTSYN pulse also produces a COLUMN COUNT pulse which steps the space counter rather than the column counter (diagram 646-0-5, Appendix 1). The EQUAL COUNT circuit compares the count in the column counter to the count in the space counter representing space characters entered in the character buffer. The CHARTSYN pulse continues to generate the DATA STROBE pulse and the COLUMN COUNT pulse until the count of the space counter equals the count in the column counter. The CHARTSYN pulse at this equal count condition produces a COLUMN COUNT pulse, which, since both LOAD SPACE and EQUAL COUNT are 0 volts, passes through the output pin H gate to generate SHIFT CHARACTER. The vertical format character is shifted out of the character input register and a CHARACTER COUNT pulse steps the character counter. This CHARACTER COUNT pulse also generates a CLEAR SPACE COUNTER pulse which resets the LOAD SPACE flip-flop and returns the space counter to a zero-count status.

A decoded carriage return control character results in the following transfer cycle. The CHARTSYN pulse generates the PRINT and SPACE command signal and, since 015 (carriage return) is negative, the CLEAR COLUMN COUNTER pulse. The characters stored in the line printer character buffer are printed out but the INHIBIT SPACE signal prevents advance of the paper (diagram 646-0-7, Appendix 1). The clearing of the column counter in effect returns the printing point for subsequent characters to the extreme left-hand column of the page.

Control Character Decoding

The control character decoding circuit (diagram 646-0-7, Appendix 1) determines if a character in the character input register is legal or illegal; and if it is legal, what action is required. The circuit consists of two Type 4151 Binary-to-Octal Decoders which decode the seven bits of each character. Additional decoding is performed by diode gates, Type 4112 and Type 4113.

If the first two bits of the character complement each other or the character is decoded as a horizontal tab control (011) or a load space operation is to be performed as a result of a PRINT and SPACE command, the ENABLE DATA STROBE levels are at the voltage potentials to enable a CHARTSYN pulse to generate the DATA STROBE command.

Enable PRINT and SPACE is negative to allow generation of the PRINT and SPACE command signal if: a load space operation is not required; a carriage return control character (015) is decoded; or a paper advance occurred after characters were entered in the line printer character buffer (FORM ADV · $\overline{1\ \text{COUNT}}$).

FORM ADVANCE levels are at the enabling potentials when the character is decoded as being a vertical format control (see listing on diagram). FORM ADVANCE coupled with a column 1 count (diagram 646-0-5, Appendix 1) produces a negative enabling level for ENABLE SPACE COMMAND. FORM ADVANCE and $\overline{\text{FORM ADVANCE}}$ combines with the count in the column counter to provide enabling levels for the $\overline{\text{FORM ADV}}$ + 1 COUNT and FORM ADV + $\overline{1\ \text{COUNT}}$ signals used with the vertical format control characters.

An ILLEGAL CHARACTER enabling level occurs when the character cannot be decoded as a printable or as a control character.

If a control character is not decoded, the CB0-CB5 content reflects a printable character. Decoding of a control character alters the CB0-CB5 code to reflect the action required of the line printer vertical format control section in advancing the paper the specified number of lines.

SPECIAL CIRCUITS

With one exception, all modules used in the Line Printer Control Type 646 are fully discussed in the System Modules Catalog (C-100). The exception is the Type 4657 Diode Gates which

form the status register and drive the I/O bus lines (see diagram 646-0-6, Appendix 1). A detailed description ot this circuit is presented in the PDP-6 Arithmetic Processor Circuits Manual, F-67 (166 cir.).

CHAPTER 3

INSTALLATION AND INTERFACE

INSTALLATION

The Line Printer Control Type 646 is housed in a standard DEC cabinet separate from the line printer (refer to the PDP-6 Installation Manual, F-68). Its exact location will depend upon the configuration of the purchased system. In a new PDP-6 system, the control is mounted in the cabinet prior to shipment, and installation procedures after delivery consist of the following:

1. Remove tape securing line printer control modules in the module panels.

2. Check module types and installed positions with engineering documentation accompanying the PDP-6 system. If discrepancies are noted, change module types and/or positions to conform to the engineering documentation.

3. Compare the installation of the eight I/O bus cables to the engineering documentation; change if necessary.

4. Place all module panel power distribution control switches in the NORMAL position.

5. Install interface cable between the Line Printer Control Type 646 and the 646 Line Printer.

6. Refer to the PDP-6 Maintenance Manual for instructions pertaining to the application of primary power to the computer cabinet.

7. Connect 646 Line Printer to primary power source and check its operation per test procedures described in the Series 5 ANelex Printer Instruction Manual.

8. Perform diagnostic program routine described in PDP-6 Maintenance Manual to verify operation of line printer control and line printer.

If the Line Printer Control Type 646 and 646 Line Printer are to be added to an existing PDP-6 system, install the units according to the instructions in the accompanying engineering documentation. After installation is complete, perform steps 7 and 8 of the above installation procedure to verify operation of the line printer control and line printer.

INTERFACE

The interface signals between the line printer control and the I/O bus cables (Figure 3-1) are listed under Signal Interface (I/O Bus). The interface signals between the line printer control and the line printer (Figure 3-2) are listed under Signal Interface (Line Printer).

Signal Interface (I/O Bus)

Signal Levels

Logical 1: 0 volts nominal
Logical 0: -3 volts nominal

PI1-PI7 (priority interrupt) - seven lines, seven-level priority interrupt code input (done or error) to PDP-6 program.

IOS3-IOS9 (I/O select) - 14 lines (1 and 0), 7-bit device selection code. Line printer is selected when line printer control decodes 001 010 1 (124_8).

CONO CLEAR - one line, negative pulse clears control register of printer control.

CONO SET - one line, negative pulse enters done and error interrupt codes in line printer control register and clears line printer control and line printer character buffer if IOB7 bit is 1.

DATAO CLEAR - one line, negative pulse initiates clearing of the line printer control character input register.

DATAO SET - one line, negative pulse enters five 7-bit characters (IOB0 through IOB34) in line printer control character input register and initiates processing of first character.

PIN	I.O. CABLE #1	I.O. CABLE #2	I.O. CABLE #3	I.O. CABLE #4
A	GND	GND	GND	GND
B	IOB Ø (1) →	IOB 18 (1) →	POWER RESET →	DATAO CLEAR →
C	IOB 1 (1) →	IOB 19 (1) →	POWER ON -15V	DATAO SET →
D	IOB 2 (1) →	IOB 2Ø(1) →		CONO CLEAR →
E	IOB 3 (1) →	IOB 21 (1) →		CONO SET →
F	IOB 4 (1) →	IOB 22(1) →	IOS 3 (Ø) →	IOB ← DATAI →
H	IOB 5 (1) →	IOB 23(1) →	IOS 3 (1) →	IOB ← STATUS →
J	GND	GND	GND	GND
K	IOB 6 (1) →	IOB 24(1) →	IOS 4 (Ø) →	
L	IOB 7 (1) →	IOB 25(1) →	IOS 4 (1) →	
M	IOB 8 (1) →	IOB 26(1) →	IOS 5 (Ø) →	
N	IOB 9 (1) →	IOB 27(1) →	IOS 5 (1) →	
P	IOB 1Ø(1) →	IOB 28(1) →	IOS 6 (Ø) →	
R	IOB 11(1) →	IOB 29(1) →	IOS 6 (1) →	PI REQ 1 →
S	GND	GND	GND	GND
T	IOB 12(1) →	IOB 3Ø(1) →	IOS 7 (Ø) →	PI REQ 2 →
U	IOB 13(1) →	IOB 31(1) →	IOS 7 (1) →	PI REQ 3 →
V	IOB 14(1) →	IOB 32(1) →	IOS 8 (Ø) →	PI REQ 4 →
W	IOB 15(1) →	IOB 33(1) →	IOS 8 (1) →	PI REQ 5 →
X	IOB 16(1) →	IOB 34(1) →	IOS 9 (Ø) →	PI REQ 6 →
Y	IOB 17(1) →	IOB 35(1) →	IOS 9 (1) →	PI REQ 7 →
Z	GND	GND	GND	GND
	(SOURCE 3L6)	(SOURCE 3L7)	(SOURCE 2L13)	(SOURCE 2N25)

ARRANGEMENT OF
I/O BUS CONNECTORS

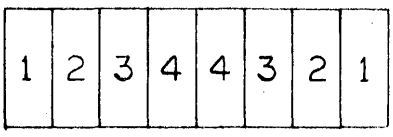


Figure 3-1 Signal Interface (I/O Bus)

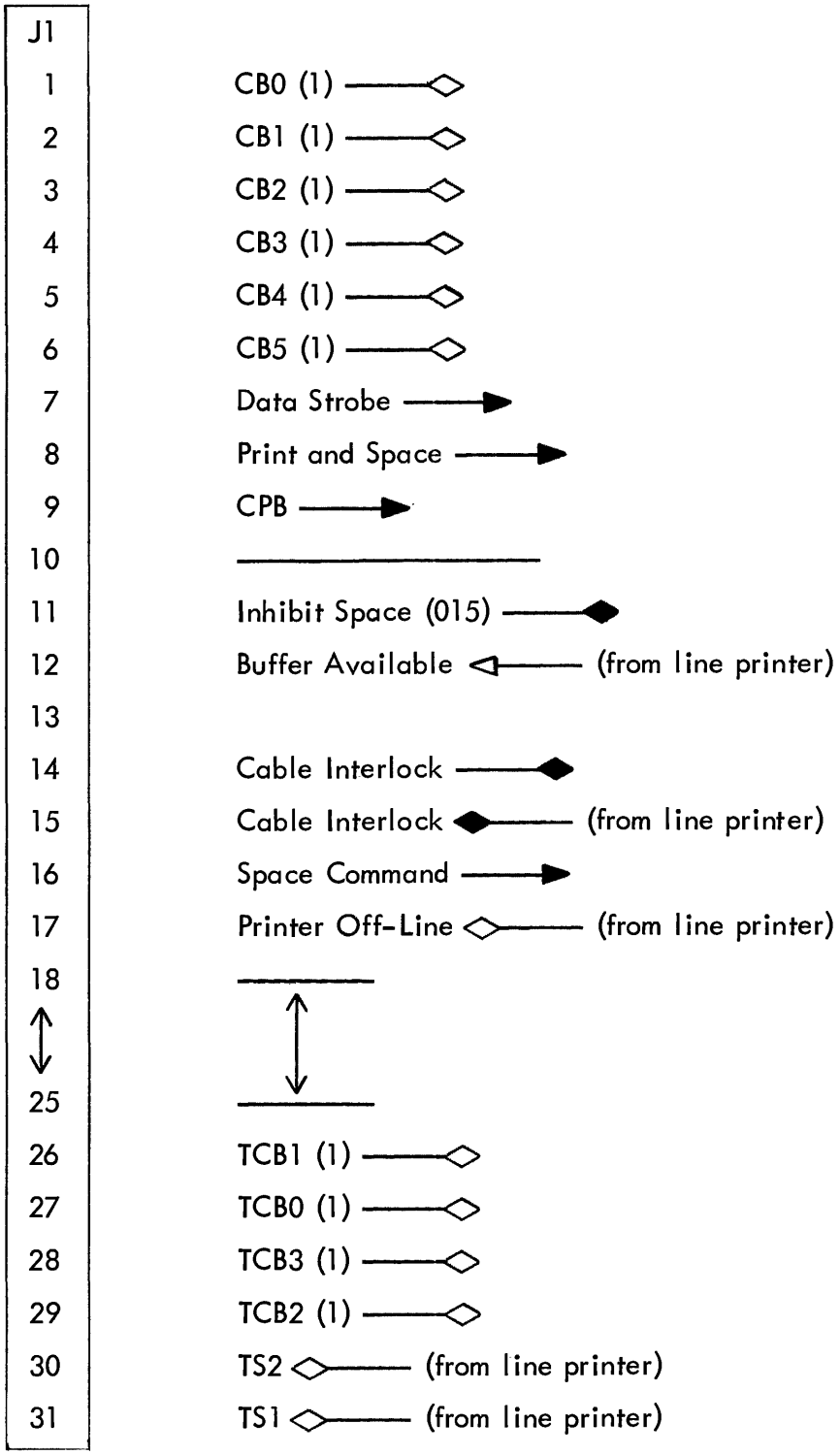


Figure 3-2 Signal Interface (Line Printer)

IOB0-IOB35 - 36 lines (two-way), logical levels (1 or 0) represent interrupt code for entry in control register by CONO, characters for entry in character input register by DATAO, or line printer control status signals for entry in PDP-6.

IOB STATUS - one line, -3 volt level coupled with line printer control selection samples contents of status register into PDP-6.

IOB POWER RESET - one line, negative pulse clears line printer control register.

Signal Interface (Line Printer)

Signal Levels

Logical 1: 0 volts nominal
Logical 0: -3 volts nominal

Output Signals (to line printer)

CB0-CB5 - six lines, printable character or control character code sent in two octal bits; printable character sampled into line printer character buffer by DATA STROBE pulse. Control character code entered in line printer vertical format control section by PRINT and SPACE or SPACE COMMAND signal unless INHIBIT SPACE signal occurs.

DATA STROBE - one line, negative pulse samples CB0-CB5 printable character into line printer character buffer.

PRINT and SPACE - one line, negative pulse initiates printout of characters stored in the line printer character buffer. Also samples CB0-CB5 control character code into line printer vertical control section. Paper advances to specified line after printout is completed unless INHIBIT SPACE signal occurs.

SPACE COMMAND - one line, negative pulse samples CB0-CB5 control character code into line printer vertical control section. Paper is advanced to line specified by code entry.

CPB (clear printer buffer) - one line, negative pulse initiates clearing of the line printer character buffer.

TC0-TC3 (tab count) - four lines, levels (1 or 0) indicate binary count of tabbed columns.

CABLE INTERLOCK - two lines, indicate by voltage level when line printer interface cable is connected.

INHIBIT SPACE - one line, negative level inhibits line printer from advancing paper when PRINT and SPACE signal occurs; stored characters in the line printer character buffer are printed out. A carriage return control character initiates the INHIBIT SPACE signal.

Input Signals (from line printer)

BUFFER AVAILABLE - one line, positive pulse indicates that line printer character buffer is ready to accept a CB0-CB5 printable character input.

TS1, TS2 (tab select) - two lines, both simultaneously at 0 volts only when preselected tab increment of 8, 10, 12, 16, or 20 columns is reached. Tab increment is selected by line printer tab selection switch.

PRINTER OFF-LINE - one line, 0-volt level indicates line printer is in off-line status; -3 volt level indicates on-line status.

CHAPTER 4

OPERATION

With two exceptions, the PDP-6 system through CONO and DATAO program instructions completely controls operation of the Line Printer Control Type 646. The first exception is the manual positioning of the 646 Line Printer tab selection switch to preselect the tab increment (8, 10, 12, 16, or 20 columns) for use with the horizontal tab control character input from the program. The tab selection switch is located within the right-hand access door of the 646 Line Printer on the test switch panel. The second exception is the MANUAL PRINT control located on the line printer console. Depression of this control initiates an immediate printout of characters stored in the line printer character buffer. Refer to the separate manual for the 646 Line Printer (series 5 ANelex line printer) listed under pertinent documents, page 1-7, for line printer operation.

Programming for the Line Printer Control Type 646 is discussed in detail in the PDP-6 Handbook (F-65) under the input/output chapter. A summary of programming for this device follows.

CONO (conditions out) - enters done and error priority interrupt codes in control register. Also clears line printer control and line printer character buffer if I/O bit 7 is 1.

DATAO (data out) - enters five 7-bit characters in character input register and initiates processing of first character.

Priority Interrupt Codes - I/O bits 12, 13, and 14 enter error priority interrupt code; I/O bits 15, 16, and 17 enter done priority interrupt code.

Device Selection Code - IOS3 through IOS9 lines must reflect 001 010 1 (124_8) to select the line printer.

CONO instruction information comes to the line printer control through the right half (I/O bits 0-17) of the 36 data lines. The status register is read out through the left half (I/O bits 18-36) of the data lines.

The five 7-bit character entries come to the line printer control through I/O bit lines 0-34.

If the line printer character buffer capacity of 120 printable characters and spaces is exceeded before the program sends a printout command, the line printer control locks in the busy state. Enter a carriage return control character by a DATAO instruction or send a CONO instruction to clear the line printer control and the line printer.

When a carriage return function is to be exercised, program the carriage return control character prior to the vertical format (paper advance) control character rather than after it. This relationship results in faster operation.

CHAPTER 5

MAINTENANCE

Maintenance of the Line Printer Control Type 646 consists of procedures repeated periodically as preventive maintenance and tasks performed in the event of equipment malfunction as corrective maintenance. The procedures presented here assume that the reader understands the function of the controls and indicators described in Table 4-1 and is familiar with PDP-6 input/output programming described in the PDP-6 Handbook and in Chapter 4 of this manual. Refer to the series 5 ANelex Line Printer Manual for maintenance of the line printer. Maintenance activities require use of the equipment listed in Table 5-1, or equivalent, as well as the use of standard hand tools, cleansers, and test cables and probes.

TABLE 5-1 MAINTENANCE EQUIPMENT

Equipment	Manufacturer	Model
Multimeter	Triplett or Simpson	630-NA or 260
Oscilloscope	Tektronix	540 Series
Parallel Drum Diagnostic Program Tape	DEC	DEC-1-137-M
System Module Extender*	DEC	1954
System Module Puller*	DEC	1960

*One supplied with the equipment

If it is necessary to remove modules during preventive or corrective maintenance, the Type 1960 System Module Puller should be used. Turn off all power before extracting or inserting modules. Carefully hook the small flange of the module puller over the center of the module rim, and gently pull the module from the mounting panel. Use a straight, even pull to avoid damage to plug connections or twisting of the printed-wiring board. Since the puller does not fasten to the module, grasp the rim of the module to prevent it from falling. Access to controls on the module for use in adjustment or access to points used in signal tracing can be gained by removing the module, connecting a Type 1954 System Module Extender into the mounting panel, and then inserting the module into the extender.

PREVENTIVE MAINTENANCE

Preventive maintenance consists of tasks performed prior to the initial operation of the Line Printer Control Type 646 and periodically during its operating life to ensure that it is in satisfactory operating condition. Faithful performance of these tasks forestalls possible future failure by correcting minor damage and discovering progressive deterioration at an early stage. A log book used to record data found during the performance of each preventive maintenance task will indicate the rate of circuit operation deterioration and provide information to determine when components should be replaced to prevent failure of the equipment. These tasks consist of mechanical checks, which include cleaning and visual inspections; checks of specific elements such as the power supplies, clock and delay module timing; and marginal checks which aggravate border-line conditions or intermittent failure so that they can be detected and corrected. All preventive maintenance tasks should be performed as a function of conditions at the installation site and the downtime limitations of equipment use. Perform the mechanical checks at least once each month or as often as required to allow efficient functioning of the air filters. All other tasks should be performed on a regular schedule, at an interval determined by the reliability requirements of the system. For a typical application, a schedule of every four months or 700 equipment operating hours, whichever occurs first, is suggested.

Mechanical Checks

Assure good mechanical operation of the equipment by performing the following steps and the indicated corrective action for any substandard conditions found:

1. Clean the exterior and the interior of the equipment cabinet housing the Line Printer Control Type 646 by using a vacuum cleaner or clean cloths moistened in nonflammable solvent.

2. Clean the air filter at the bottom of the cabinet. Remove the filter by removing the fan and housing, which are held in place by two knurled and slotted captive screws. Wash the filters in soapy water, dry in an oven or by spraying with compressed gas, and spray with Filter-Kote (Research Products Corporation, Madison, Wisconsin) before replacing them in the cabinets.

3. Lubricate door hinges and casters with a light machine oil. Wipe off excess oil.

4. Visually inspect the equipment for completeness and general condition. Repaint any scratched or corroded areas with DEC blue enamel, No. 5150-S65.

5. Inspect all wiring and cables for cuts, breaks, fraying, deterioration, kinks, strain, and mechanical security. Tape, solder, or replace any defective wiring.

6. Inspect all mounting panels of logic to assure that each module is securely seated in its connector.

7. Verify that eight I/O bus cables are firmly seated in their respective connectors.

8. Inspect power supply capacitors for leaks, bulges, or discolorations. Replace any capacitors giving these signs of malfunction.

Power Supply Checks

Check the output voltage and ripple content of the Type 728 Power Supplies and assure that they are within tolerance. Use the multimeter to make the output voltage measurements without disconnecting the load. Use the oscilloscope to measure the peak-to-peak ripple content on dc outputs of the supplies. These supplies are not adjustable; so if the output voltage or ripple content is not within the tolerance specified, the supply is considered defective and troubleshooting procedures should be undertaken.

Check the +10-volt output between the black (-) and the red (+) terminals to assure that it is between 9.5 and 11.0 volts with less than 800 millivolts ripple. Check the -15 volt output between the black (+) and blue (-) terminals to assure that it is between 14.5 and 16.0 volts with less than 400 millivolts ripple. Note that the black terminals are common with the power supply chassis.

CORRECTIVE MAINTENANCE

The Line Printer Control Type 646 is constructed of highly reliable transistorized modules and standard circuits. Use of these circuits and faithful performance of the preventive maintenance tasks ensure relatively little equipment downtime due to failure. Should a malfunction occur, the condition should be analyzed and corrected as indicated in the following procedures. No special tools or test equipment are required for corrective maintenance other than a broad bandwidth oscilloscope and a standard multimeter. The best corrective maintenance tool is a thorough understanding of the physical and electrical characteristics of the system. Persons responsible for maintenance should become thoroughly familiar with the theory of operation as described in Chapter 2, specific circuit modules as described in the DEC System Modules Catalog, the engineering drawings presented in Appendix 1, and the location of mechanical and electrical components as described in Chapter 1.

Diagnosis and remedial action for a fault condition are performed in the following phases:

1. Preliminary investigation to gather all information and to determine the physical and electrical security of the drum system.

2. System troubleshooting to locate the fault to within a module through the use of diagnostic programming, signal tracing, or aggravation techniques.

3. Circuit troubleshooting to locate defective parts within a module.

4. Repairs to replace or correct the cause of a malfunction.

5. Validation test to assure that the fault has been corrected.

6. Log entry to record pertinent data.

Preliminary Investigation

It is virtually impossible to outline any specific procedures for locating faults within a complexed digital system such as the line printer control. Before commencing troubleshooting procedures, explore every possible source of information. Ascertain all possible information concerning any unusual function of the system prior to the fault and all possible program information such

as routine in progress, condition of indicators, etc. Search the maintenance log to determine if this type of fault has occurred before or if there is any cyclic history of this kind of fault, and determine how this condition was previously corrected. When the entire line printer control fails, perform a visual inspection to determine the physical and electrical security of all power sources, cables, connectors, etc. Assure that the power supplies are working properly and that there are no power short circuits by performing the power supply checks as described under Preventive Maintenance.

System Troubleshooting

Do not attempt to troubleshoot the line printer control without first gathering all information possible concerning the fault, as outlined under Preliminary Investigation.

Commence troubleshooting by performing that operation in which the malfunction was initially observed, using the same program. Thoroughly check the program for proper control settings, and note all indicator light operations before and at the time of the error. Careful checks should be made to assure that the system is actually at fault before continuing with corrective maintenance procedures. Loose or faulty cable connections can often give indications very similar to those caused by internal malfunctions. Faulty ground connections between pieces of equipment are a common source of trouble.

If the fault has been determined to lie within the Line Printer Control Type 646, but cannot be localized to a specific logic function, perform the diagnostic program procedure. When the location of the fault has been narrowed to a logic element, continue troubleshooting to locate the defective module or component by means of signal tracing. If the fault is intermittent, a form of aggravation tests should be employed to locate the source of the fault.

Diagnostic Program

The most efficient means of troubleshooting the line printer control makes use of the diagnostic program described in Maindec 646. This routine provides a complete test of operations of the line printer control.

Signal Tracing

If the fault has been located within a functional logic element, program the PDP-6 to repeat some instruction in which all functions of that logic element are utilized. If this test is to be performed without the use of the computer, control flip-flops or register flip-flops can be cleared or set manually by momentarily supplying a ground potential to the appropriate flip-flop output terminals. Counting operations of registers can be checked by supplying count pulses to the register from the output of a variable clock. Under these conditions, use the oscilloscope to trace signal flow through the suspected logic element. Oscilloscope sweep may be synchronized with any line printer control signal by connecting the trigger input to the appropriate module terminal on the wiring side (front) of the equipment. Trace output signals from the connector back to the origin, and trace input signals from the connector to its final destination. The signal-tracing method can be used to determine with absolute certainty the quality of pulse amplitude, duration, rise time, and the correct timing sequence of this signal. If an intermittent malfunction occurs, signal tracing must be combined with an appropriate form of aggravation test.

Intermittent Failures

Intermittent failures caused by poor wiring connections can often be revealed by vibrating the modules while running a repetitive routine, such as the diagnostic program. Often, wiping the handle of a screwdriver across the back of a suspect row of modules is a useful technique. By repeatedly starting the program and vibrating fewer and fewer modules, the malfunction can be localized to within one or two modules. After isolating the malfunction in this manner, check the seating of the modules in the connector, the module connector for wear and misalignment, and the module wiring for cold solder joints or wiring kinks.

Circuit Troubleshooting

The procedure followed for troubleshooting and correcting the cause of faults within modules and power supplies depends upon thedowntime limitations of line printer control use. Where downtime must be kept at a minimum, it is suggested that a provisioning parts program be adopted to maintain one spare module or power supply which can be inserted into the cabinet when system troubleshooting procedures have traced the fault to a particular component. A list of

Often the response time of the multimeter is too slow to detect the rapid transients produced by the intermittent connections. Current interruptions of very short durations, caused by an intermittent connection, can be detected by connecting a 1.5-volt flashlight battery in series with a 1500-ohm resistor across the suspected connection. Observe the voltage across the 1500-ohm resistor with an oscilloscope while probing the connection.

Dynamic Bench Tests

Dynamic bench testing of modules can be performed through the use of special equipment. A Type 922 Test Power Cable and either a Type 722 or Type 765 Power Supply can be used to energize a system module. These supplies provide both the +10 vdc and -15 vdc operating power for the module as well as ground and - 3 volt sources which may be used to simulate signal inputs. The signal input potentials can be connected to any terminal normally wired to receive logic level signals by means of eyelets provided on the power cable. Type 911 Patch Cords may be used to make these connections between eyelets on the plug. In this manner logic operations and voltage measurements can be made throughout the circuit. When using the Type 765 Bench Power Supply, marginal checks of an individual module can also be obtained.

Repair

In all soldering and unsoldering operations in the repair and replacement of parts, avoid placing excessive solder or flux on adjacent parts or service lines. When soldering semiconductor devices (transistors, crystal diodes, and metallic rectifiers) which may be damaged by heat, the following special precautions should be taken:

1. Use a heat sink, such as a pair of pliers, to grip the lead between the device and the joint being soldered.

2. Use a 6-volt soldering iron with an isolation transformer. Use the smallest soldering iron adequate for the work.

3. Perform the soldering operation in the shortest possible time to prevent damage to the component and delamination of the module etched wiring.

When any part of the equipment is removed for repair and replacement, make sure that all leads or wires which are unsoldered, or otherwise disconnected, are legibly tagged or marked for identification with their respective terminals. Replace defective comments only with parts of equal or greater quality or narrower tolerance.

Validation Test

Following the replacement of any electrical component in the equipment, a test should be performed to assure the correction of the fault condition and to make any adjustment of the timing or signal levels affected by the replacement. This test should be taken from the preventive maintenance procedure most applicable to the portion of the system in which the fault was found. For example, if a filter capacitor was replaced in one of the power supplies, the ripple check for that power supply should be repeated as specified under Power Supply Checks. If repairs or replacements are made in an area which is not checked on preventive maintenance, an appropriate operational test should be devised. Normally, the diagnostic program serves this purpose if the error was found in a logic element pertaining to data transfer functions. If the fault occurred in the control elements of the machine, such as a flip-flop replacement, the register or control function performed by the flip-flop should be completely checked by manually setting and clearing or by programmed exercise of that function.

When time permits, it is suggested that the entire preventive maintenance tasks be performed as a validation test. The reasons for this are:

1. If one fault occurred and was corrected, other components may be marginal.

2. While the equipment is down and available, preventive maintenance can be performed and need not be scheduled again for four months (or the normal period).

Log Entry

Corrective maintenance activities are not completed until they are recorded in the maintenance log. Record all data indicating the symptoms given by the fault, the method of fault detection, the component at fault, the results of the validation tests, and any other information which would be helpful in maintaining the equipment in the future.

APPENDIX 1

ENGINEERING DRAWINGS

Reduced engineering drawings are produced in this appendix as an aid to understanding and maintaining the system. A complete set of formal engineering drawings is supplied separately with each system. Should any discrepancy exist between the drawings in this manual and those supplied with the equipment, assume the formal drawings to be correct.

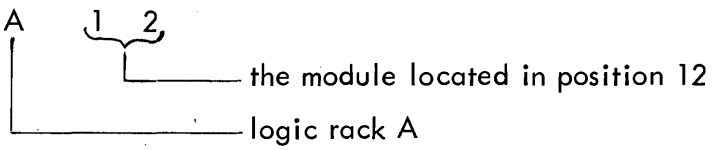
Refer to the Table of Contents for a complete list of engineering drawings for this system and the page on which the drawings appear.

For all modules illustrated, circuit type is always shown as a four-digit number. This number is the same type number used to identify the circuit in the DEC System Modules Catalog (C-100).

Example: 4112 --- Six 2-Input Negative Diode Gates (500-kc series)

On all circuit modules, the circuit location code is lettered directly below the circuit type number. Circuit location code is shown as a single letter followed by one or two digits.

Example:



As indicated on certain block schematics, the normal four-digit module identification number is followed by the letter R. This letter indicates that these modules contain internal jumpers which connect output terminals to clamped load resistors located on the module. Such modules are shipped from Digital with all jumpers connected. Before these modules can be installed in a particular Digital component, the module user must consult the block schematic for that component to determine which jumpers are to be removed and which jumpers are to remain connected. A number following a module identification number indicates which circuits of the module have clamped loads and which have unclamped loads. Decode this number (octal to binary) with 0 indicating an unclamped load and 1 indicating a clamped load. Each engineering logic drawing is divided into 32 zones (four horizontal and eight vertical) by marginal map coordinates. Figure references in the text are usually followed by a letter and a digit specifying the zone in which the referenced circuit is located.

SYMBOLS AND TERMINOLOGY

Engineering drawing numbers for this equipment contain five pieces of information, separated by hyphens. Reading from left to right, these bits of information are a two-letter code specifying the type of drawing, a one-letter code specifying the size of the drawing, the type number of the equipment, the manufacturing series of the equipment, and a two-digit number specifying the number of a drawing within a particular series. The drawing type codes are:

1. BS, block schematic or logic diagram
2. CD, cable diagram
3. CS, circuit schematic
4. FD, flow diagram
5. ID, interconnection drawing
6. PW, power wiring
7. RS, replacement schematic
8. SD, system diagram
9. TD, timing diagram
10. TFD, timing and flow diagram
11. UML, utilization module list
12. WD, wiring diagram

Logic Signals

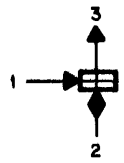
All logic signals are either standard Digital logic levels or standard Digital pulses. A standard Digital logic level is either a ground (0 to -0.3 volts) or -3 volts (-2.5 to -3.5 volts). Logic signals are generally given mnemonic names which indicate the condition represented by assertion of the signal. An open diamond (——◇) indicates that the signal is a level and that ground represents assertion; a solid diamond (——◆) indicates that the signal is a level and that -3 volts represents assertion.

All logic levels applied to the conditioning-level inputs of capacitor-diode gates must be present either 1 or 3 microseconds (depending on the module used) before an input triggering pulse is applied to the gate.

The standard Digital negative pulse is indicated by a solid triangle (——▶) and goes from ground to -2.5 or -3 volts (-2.3 to -3.5 volt tolerances). The standard Digital positive pulse, indicated by an open triangle (——▷), goes either from -3 volts to ground or from ground to +2.5 volts (+2.3 to +3.0 volts). The width of the standard pulses used in this equipment is either 1.0, 0.4, or 0.07 microseconds, depending on the module and application.

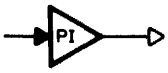
Occasionally, the transition of a level is used at an input where a standard pulse is otherwise expected and a composite symbol (——◇▷) is drawn to indicate this fact. The triangle is drawn open or solid depending, respectively, on whether the positive (-3 volt to ground) or the negative (ground to -3 volt) transition triggers circuit action. The shading of the diamond is the same as that of the triangle to indicate triggering on the leading edge of a level, or is opposite that of the triangle to indicate triggering on the trailing edge.

Any other signal is nonstandard and is indicated by an arrowhead (——→) pointing in the direction of signal flow. Figure A1-1 shows the standard symbols used in all Digital logic drawings.

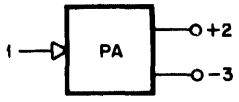


CAPACITOR-DIODE GATE, POSITIVE OR NEGATIVE INDICATED BY POLARITY OF THE INPUTS.

1. PULSE INPUT
2. CONDITIONING LEVEL INPUT
3. PULSE OUTPUT



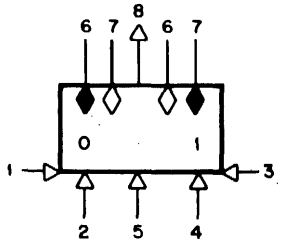
PULSE INVERTER



PULSE AMPLIFIER

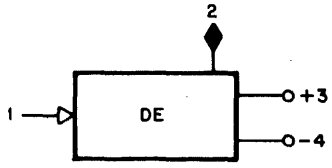
1. PULSE INPUT, POLARITY INDICATED BY INPUT SIGNAL

2,3. TRANSFORMER-COUPLED PULSE OUTPUT



FLIP-FLOP (MOST FLIP-FLOPS HAVE ONLY SOME OF THE FOLLOWING):

1. DIRECT-CLEAR INPUT
2. GATED-CLEAR INPUT
3. DIRECT-SET INPUT
4. GATED-SET INPUT
5. COMPLEMENT INPUT
6. OUTPUT LEVEL, -3 V IF 0, 0 V IF 1
7. OUTPUT LEVEL, 0 V IF 0, -3 V IF 1
8. CARRY PULSE OUTPUT



DELAY (ONE-SHOT MULTIVIBRATOR)

1. INPUT PULSE
2. OUTPUT LEVEL, -3V DURING DELAY

3,4. TRANSFORMER-COUPLED PULSE OUTPUT

Figure A1-1 Digital Logic Symbols

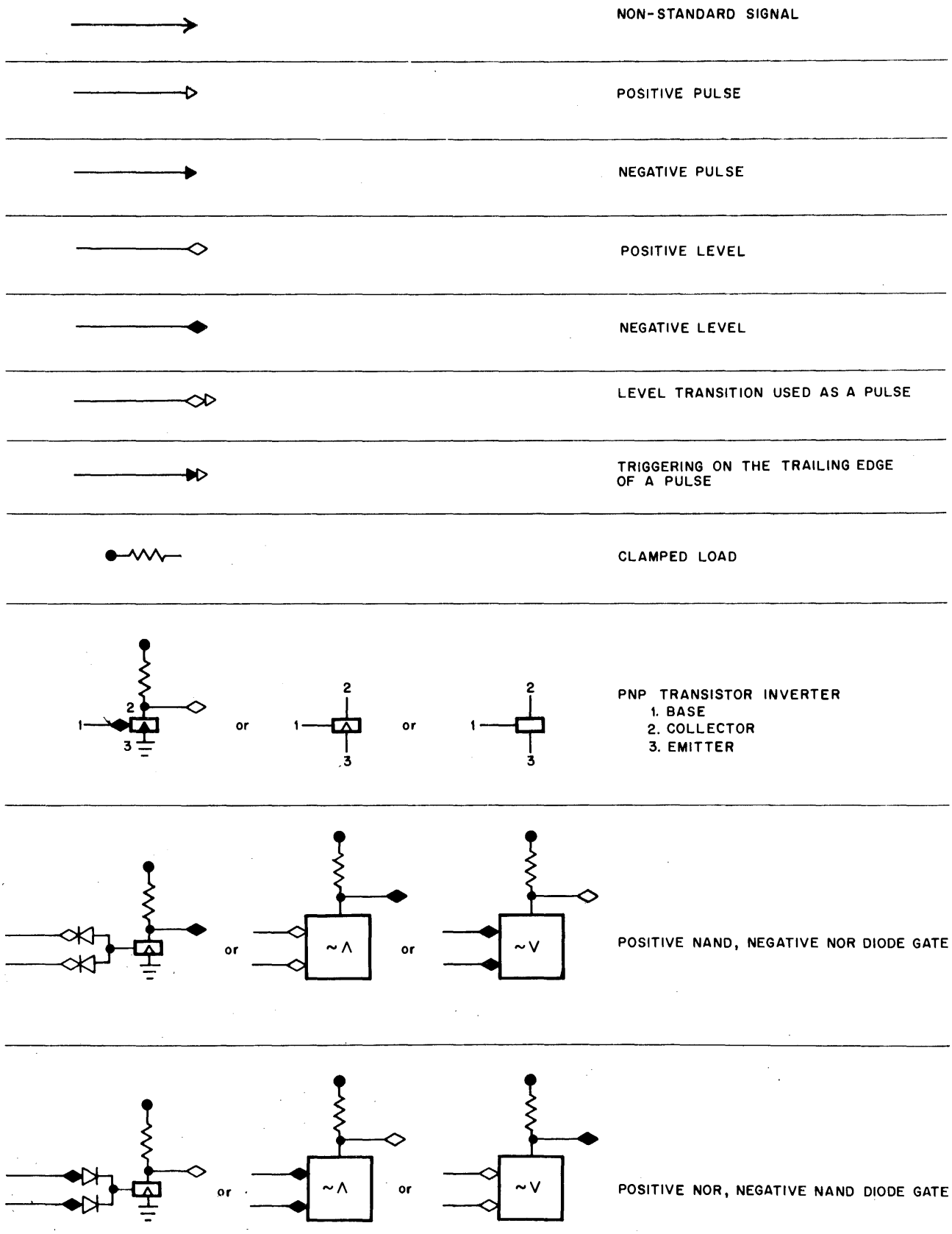
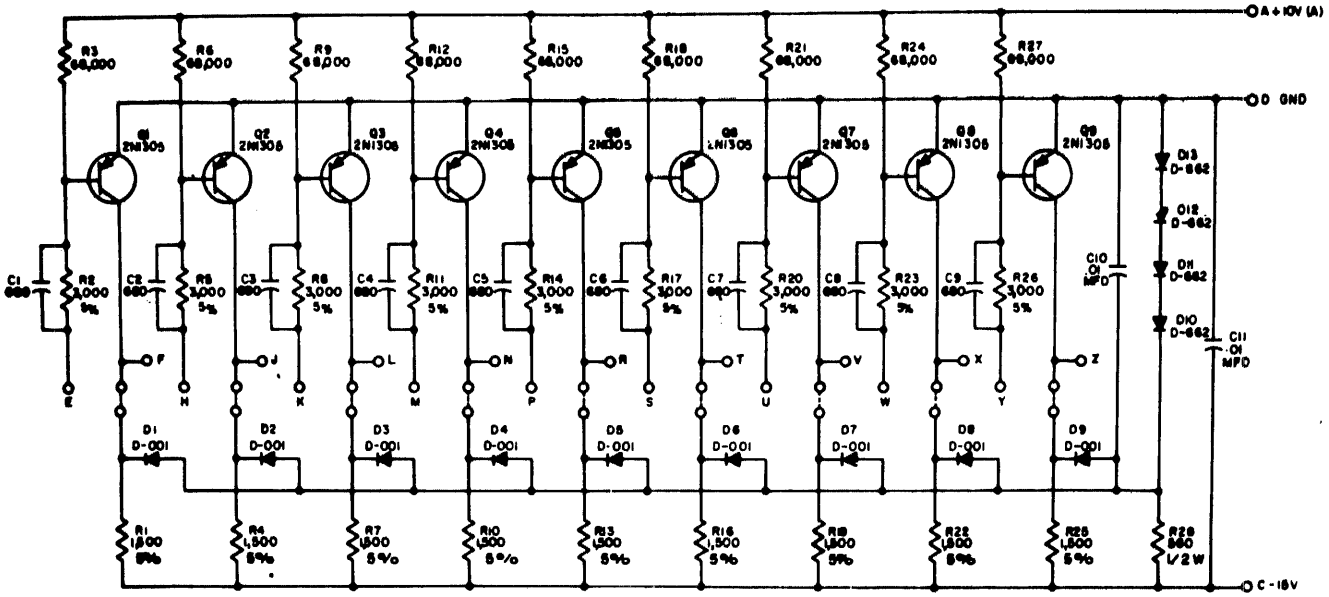


Figure A1-1 Digital Logic Symbols (continued)

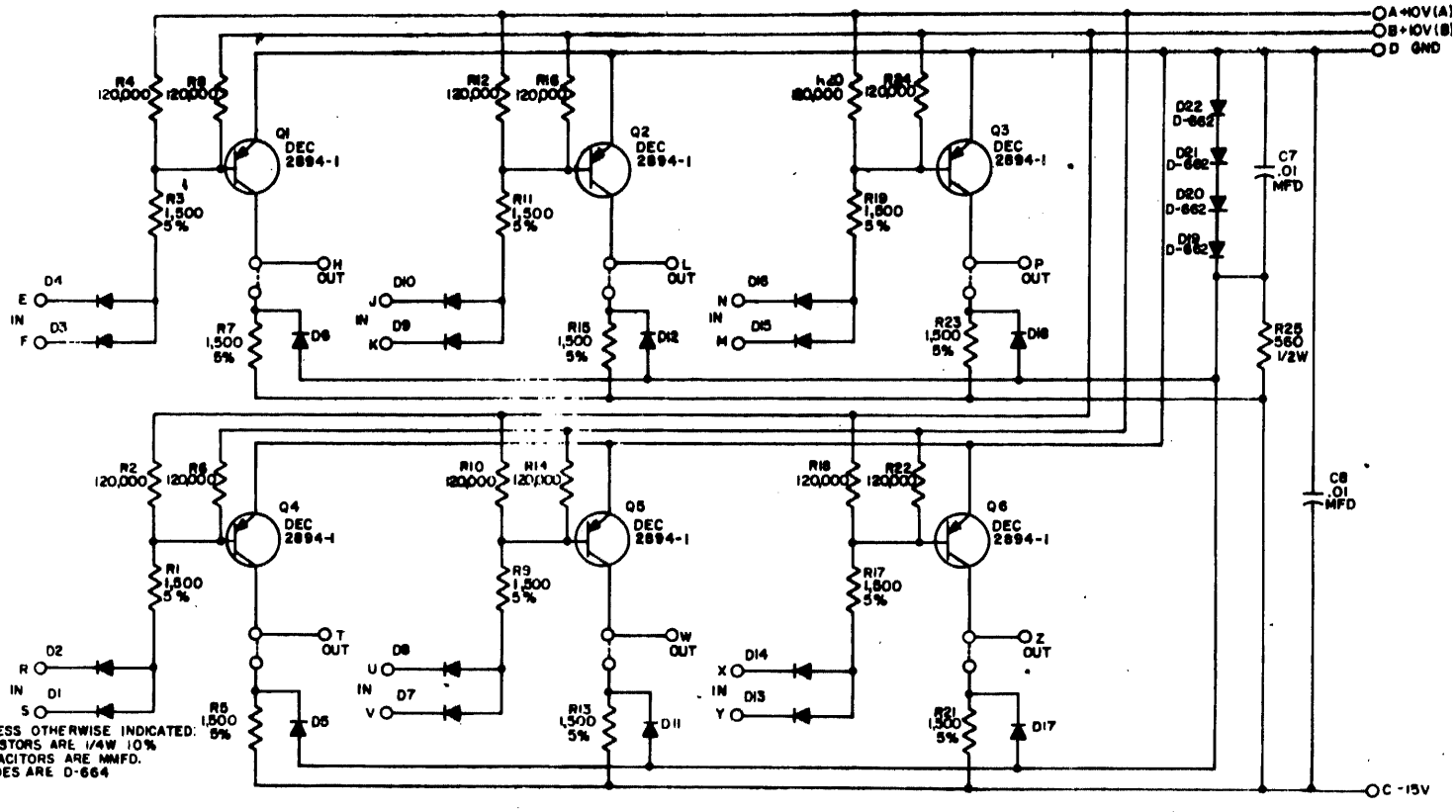


UNLESS OTHERWISE INDICATED
RESISTORS ARE 1/4W 10%
CAPACITORS ARE MMFD

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
[illegible]	[illegible]		

Inverter

RS-4102

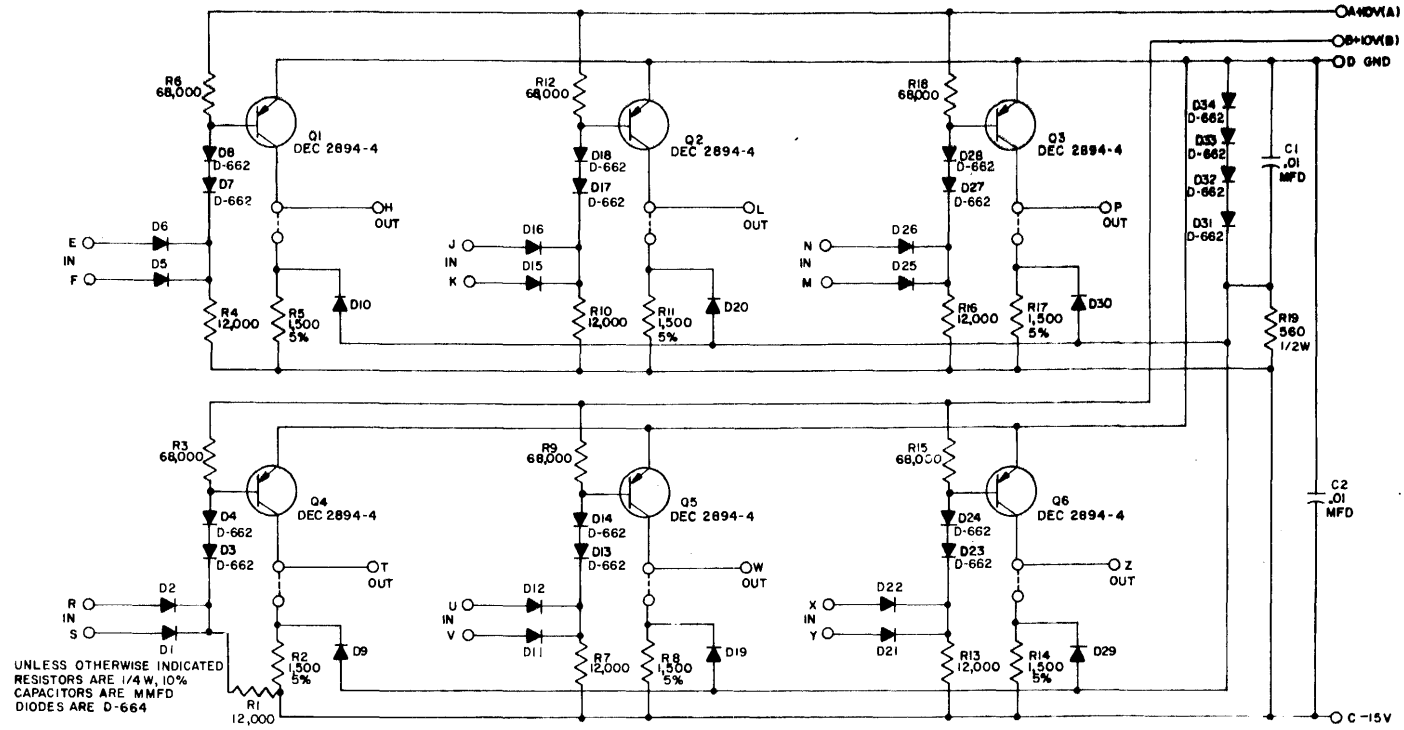


UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W 10%
CAPACITORS ARE MMFD.
DIODES ARE D-664

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
[illegible]	[illegible]		

Diode

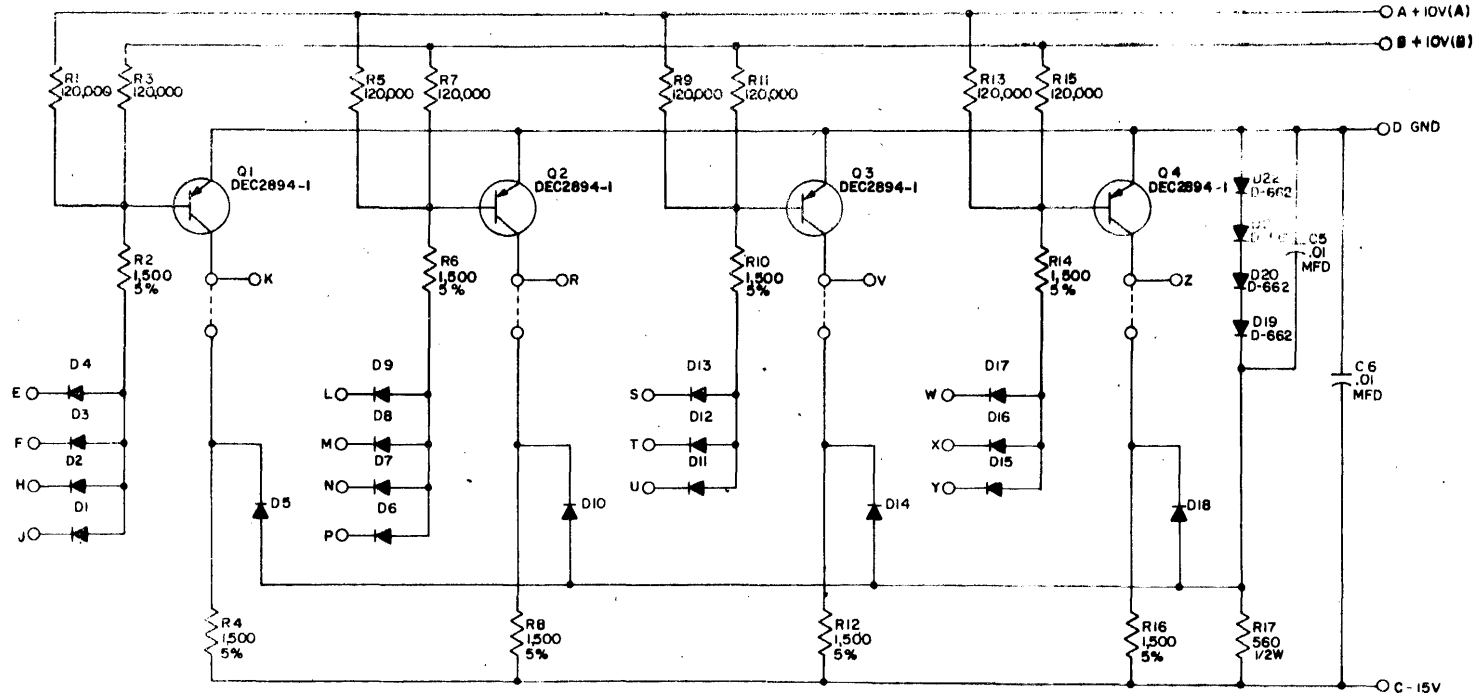
RS-4112



TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-4	DEC 2894		
D-662	IN645		
D-664	IN3606		

Diode

RS-4113

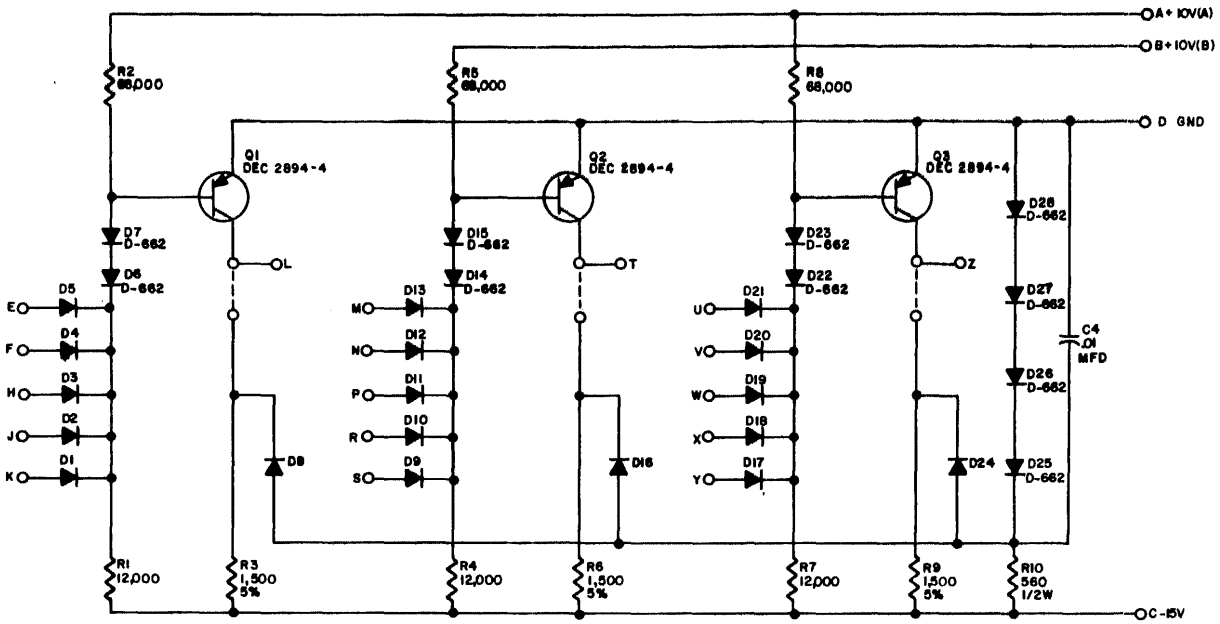


UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%
CAPACITORS ARE MMFD.
DIODES ARE D-664

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC2894-1	DEC2894-1		
D-662	IN645		
D-664	IN3606		

Diode

RS-4114

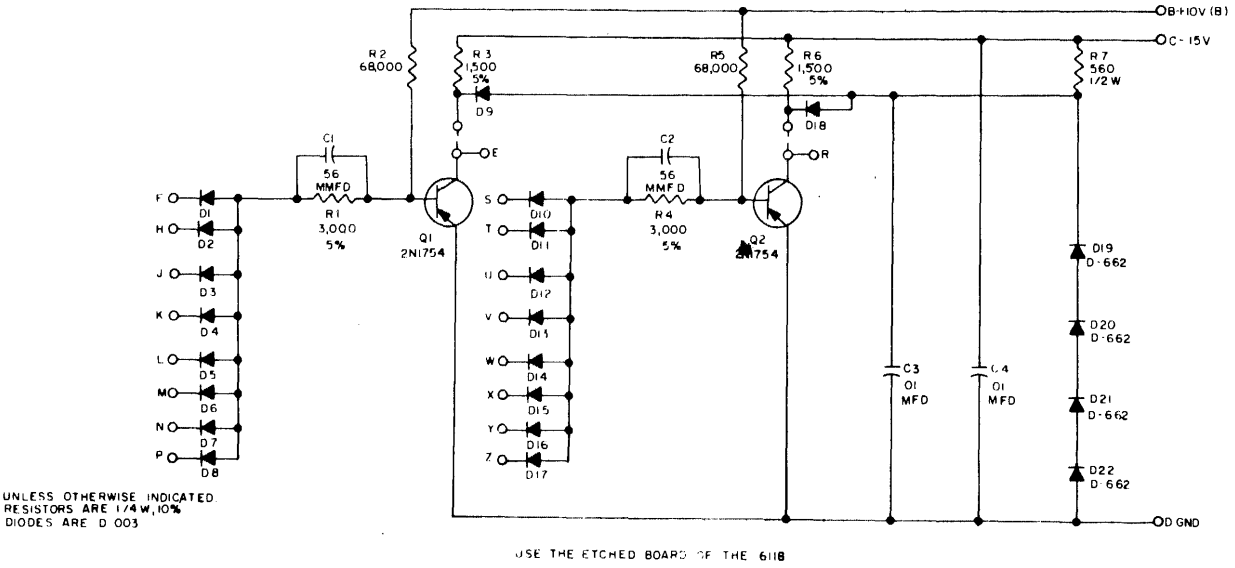


UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%
DIODES ARE D-664

Diode

RS-4117

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-4	DEC 2894		
D-664	IN3606		
D-662	IN645		



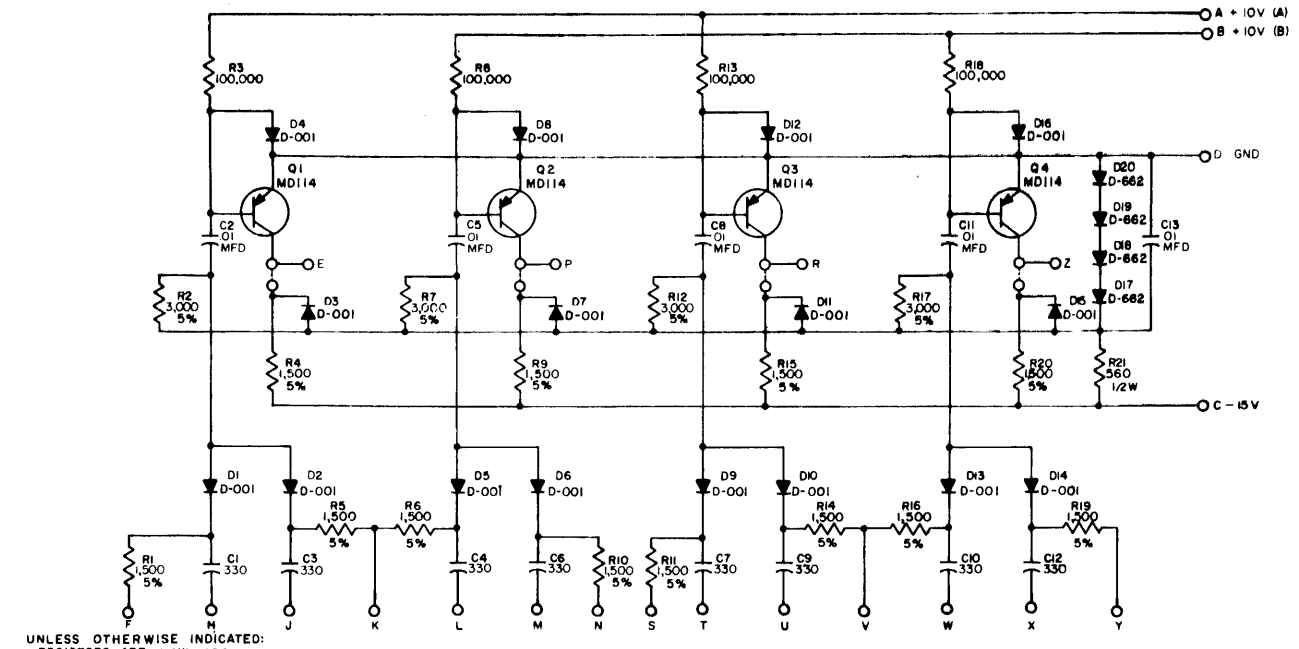
UNLESS OTHERWISE INDICATED
RESISTORS ARE 1/4 W, 10%
DIODES ARE D 003

USE THE ETCHED BOARD OF THE 6118

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
D-003	IN994		
D-662	IN645		
2N1754	2N1754		

Negative Diode NOR

RS-4118

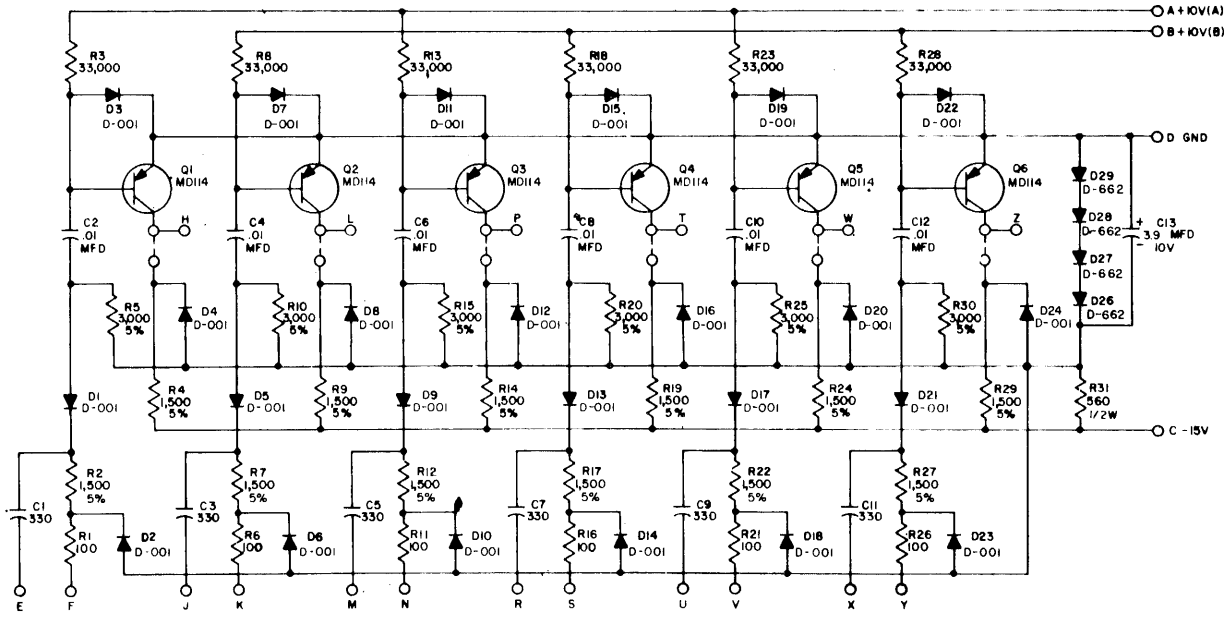


UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%
CAPACITORS ARE MMFD

TRANSISTOR & DIODE CONVERSION CHART			
DEC	ETA	DEC	ETA
MD114	2N1499A		
D-001	IN276		
D-662	IN646		

Capacitor-Diode-Inverter

RS-4125

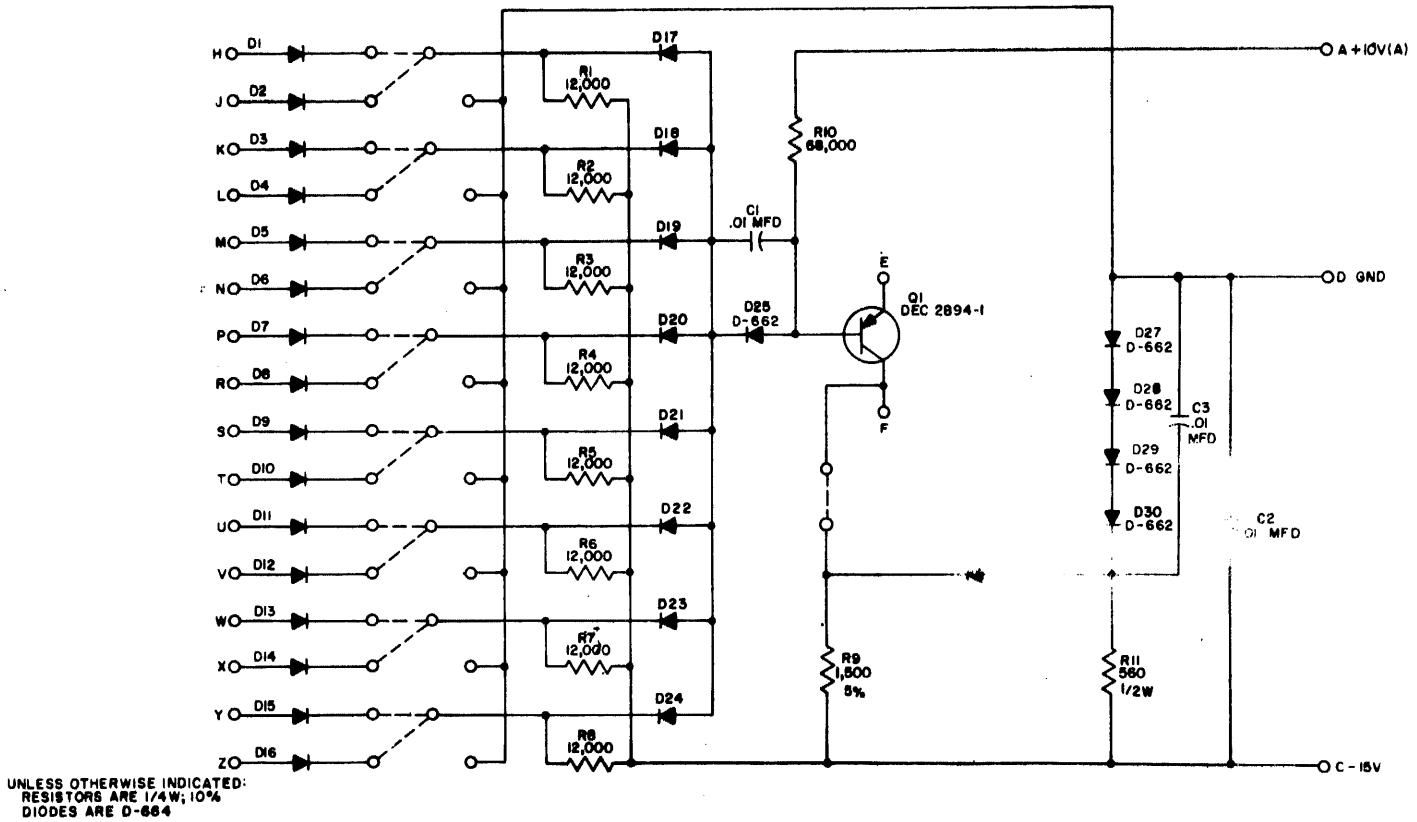


UNLESS OTHERWISE INDICATED
RESISTORS ARE 1/4W, 10%
CAPACITORS ARE MMFD

TRANSISTOR & DIODE CONVERSION CHART			
DEC	ETA	DEC	ETA
MD114	2N1499A		
D-001	IN276		
D-662	IN646		

Capacitor-Diode-Inverter

RS-4127

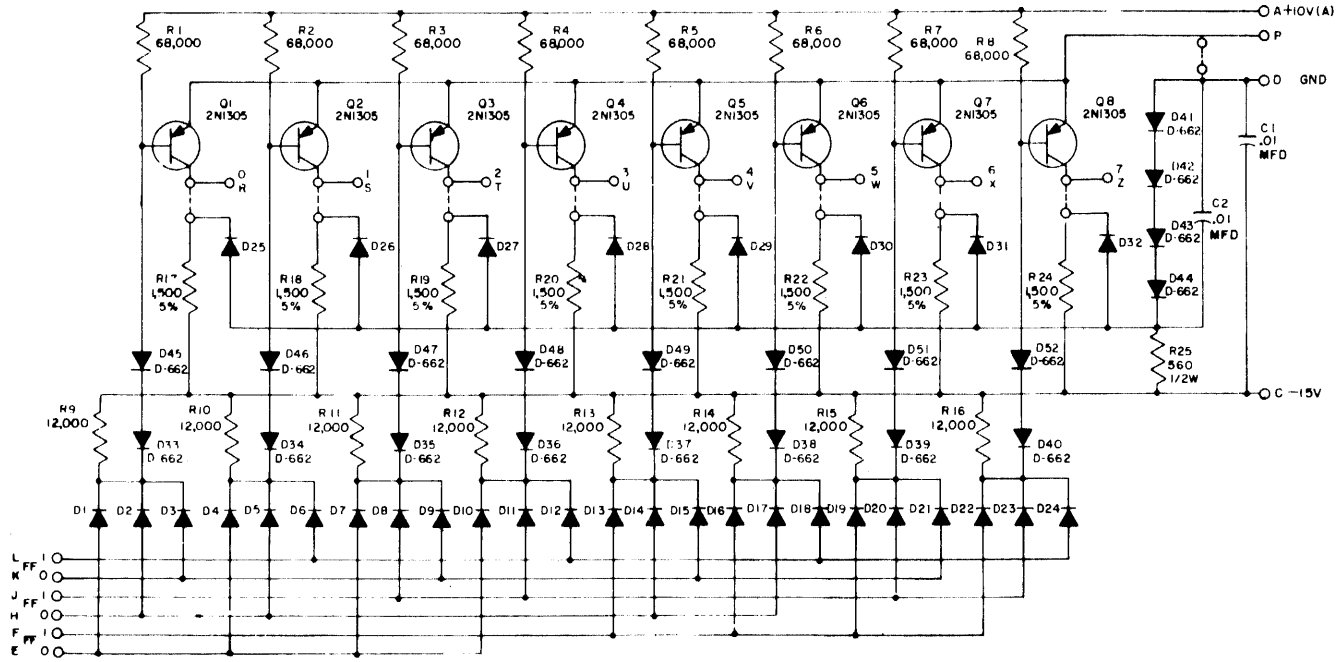


UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4W; 10%
DIODES ARE D-664

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-1	DEC 2894-1		
D-662	IN645		
D-664	IN3609		

Diode Unit

RS-4141

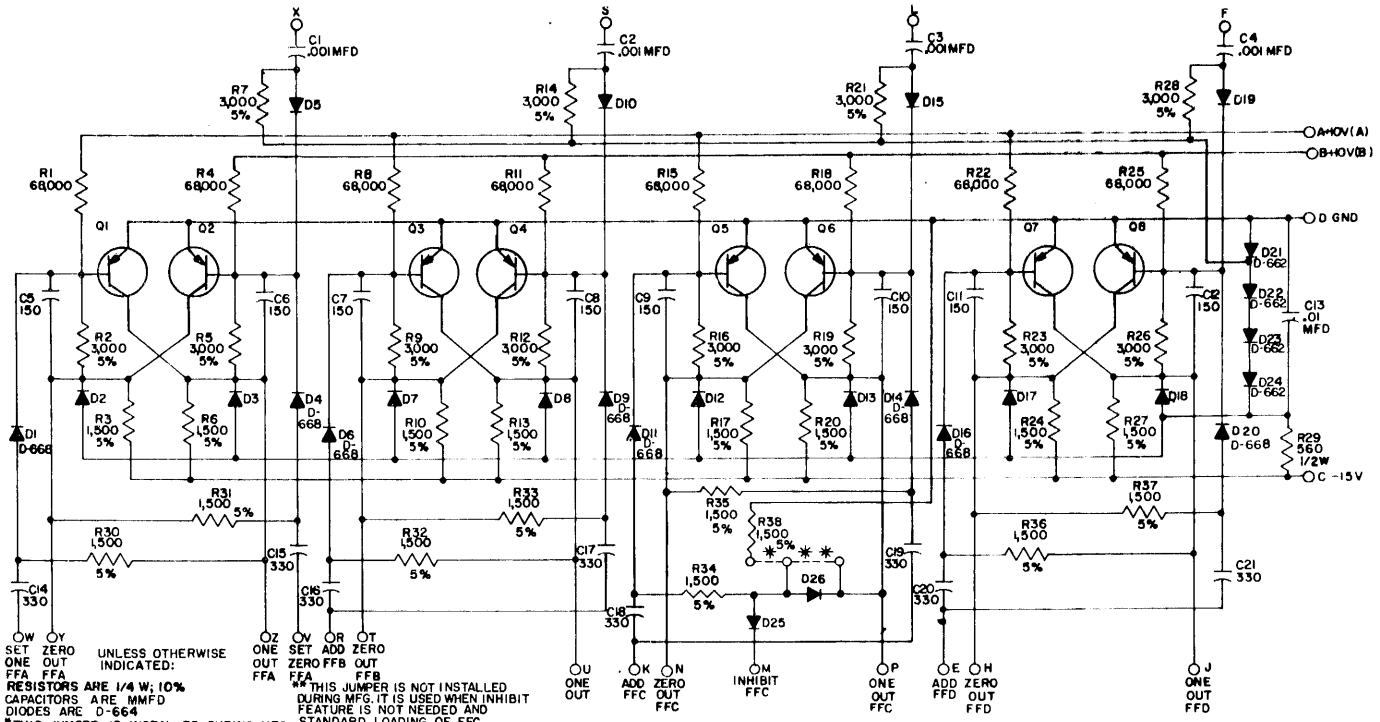


UNLESS OTHERWISE INDICATED
RESISTORS ARE 1/4W, 10%
DIODES ARE D-001

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
2N1305	2N1305		
D-001	IN276		
D-662	IN645		

Binary-to-Octal Decoder

RS-4151

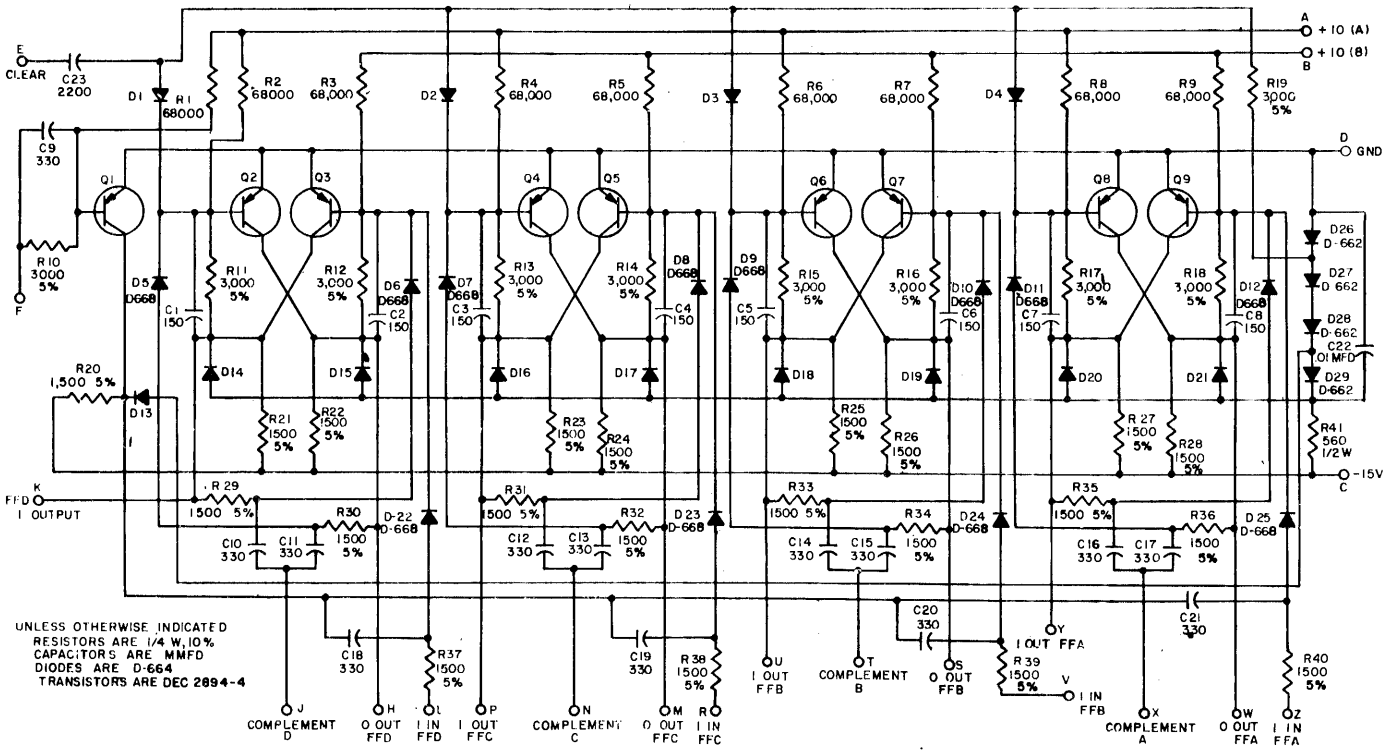


TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-4	DEC 2894		
D-668	D-668*		
D-662	IN645		
D-664	IN3606		

NOTES
* (TWO) IN3606 IN SERIES

4-Bit Counter

RS-4215

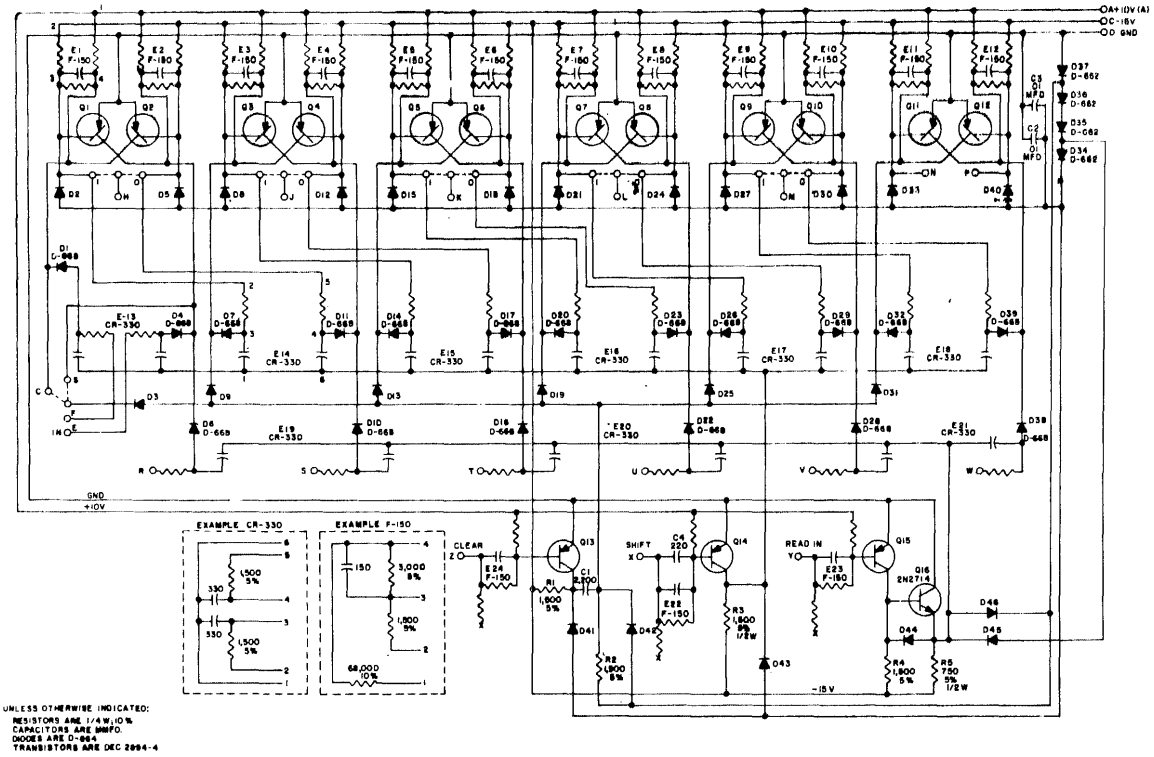


TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-4	DEC 2894		
D-668	D-668*		
D-662	IN645		
D-664	IN3606		

NOTES
* (TWO) IN3606 IN SERIES

4-Bit Counter

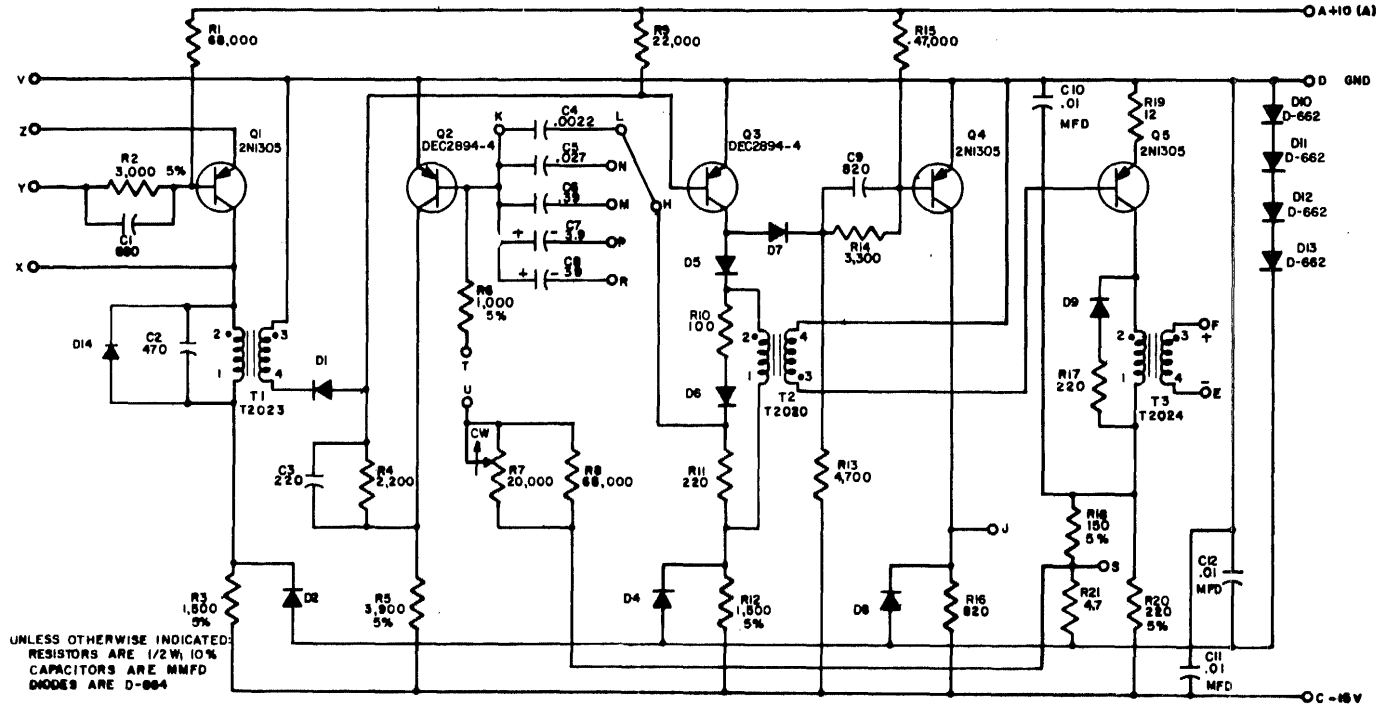
RS-4217



TRANSISTOR & DIODE CONVERSION CHART				NOTES
DEC	EIA	DEC	EIA	* (TWO) INSIDE IN SERIES
DEC 2894-4	DEC 2894			
2N1305	2N1305			
D-662	D-662			
D-663	D-663			
D-664	1N3604			

6-Bit Shift Register

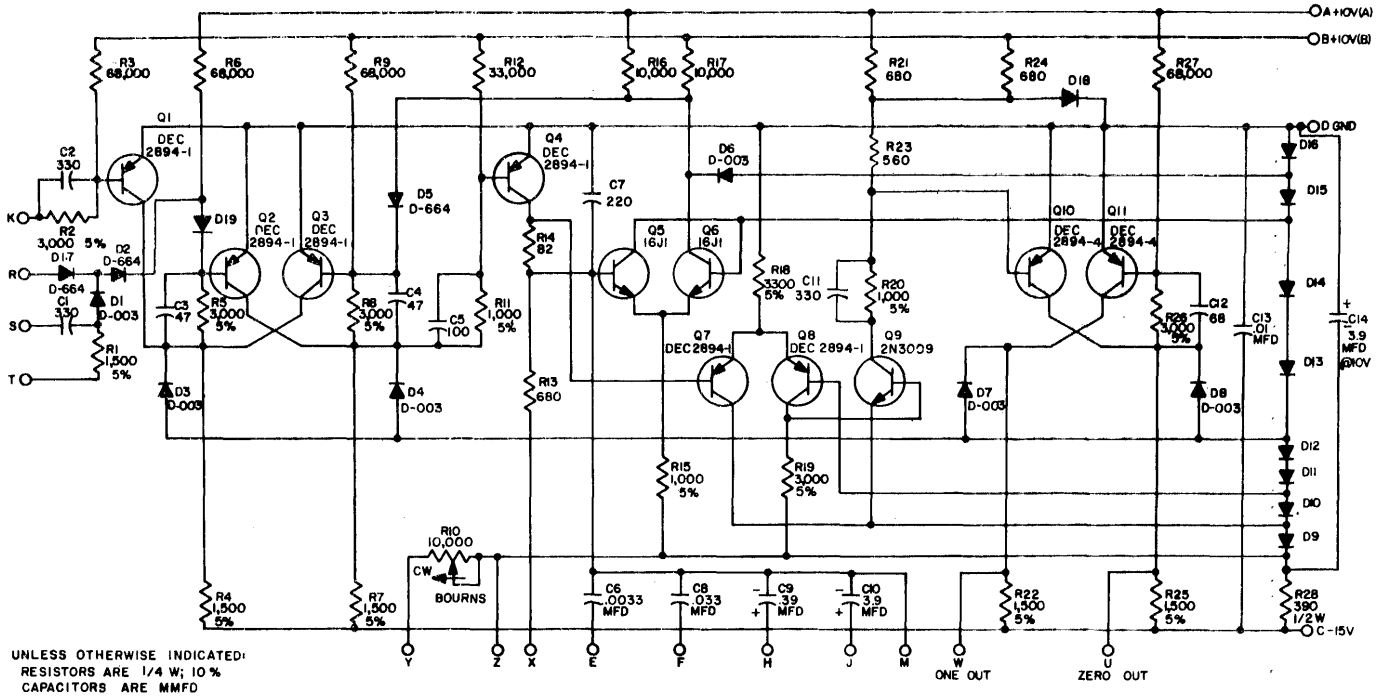
RS-4221



TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-4	DEC 2894		
2N1305	2N1305		
D-664	1N3604		
D-663	IN648		

Delay

RS-4301

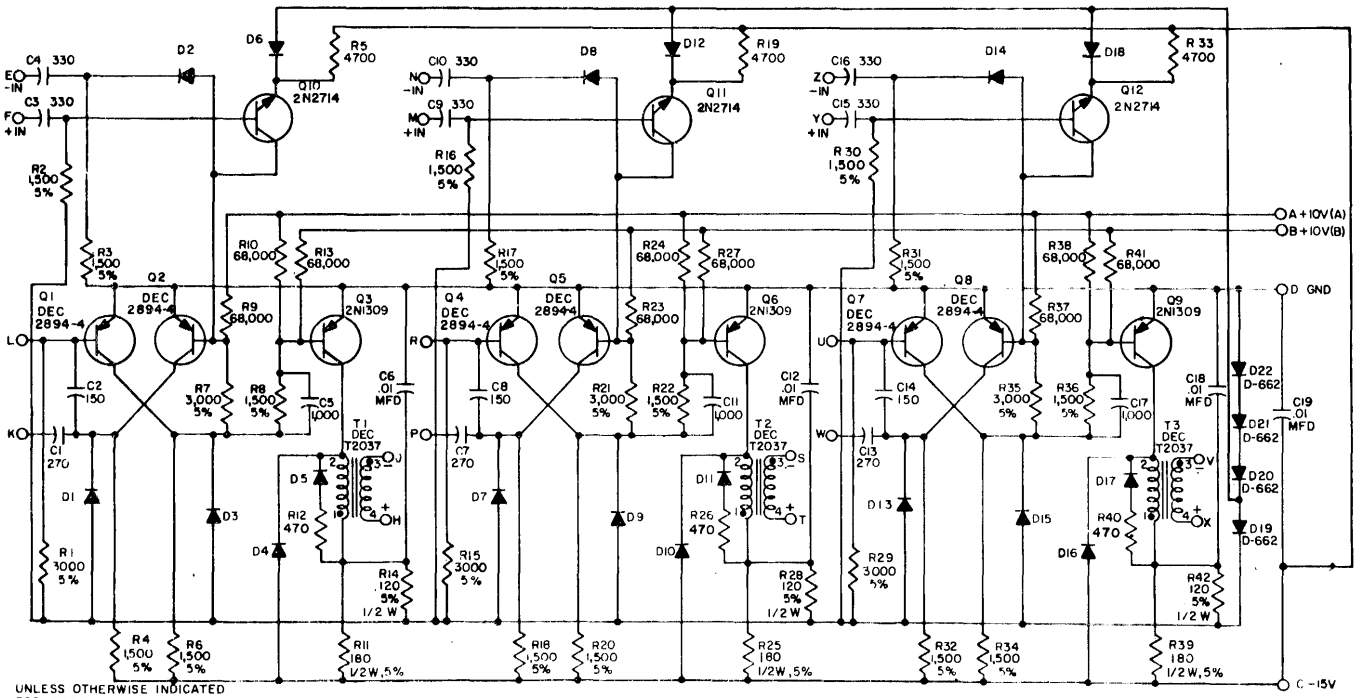


UNLESS OTHERWISE INDICATED:
RESISTORS ARE 1/4 W; 10%
CAPACITORS ARE MMFD
DIODES ARE D-662

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-1	DEC 2894	D-662	IN648
2N3009	2N3009	DEC 2894-4	DEC 2894
16J1	16J1		
D-003	IN994		
D-664	IN3605		

Integrating One-Shot

RS-4303

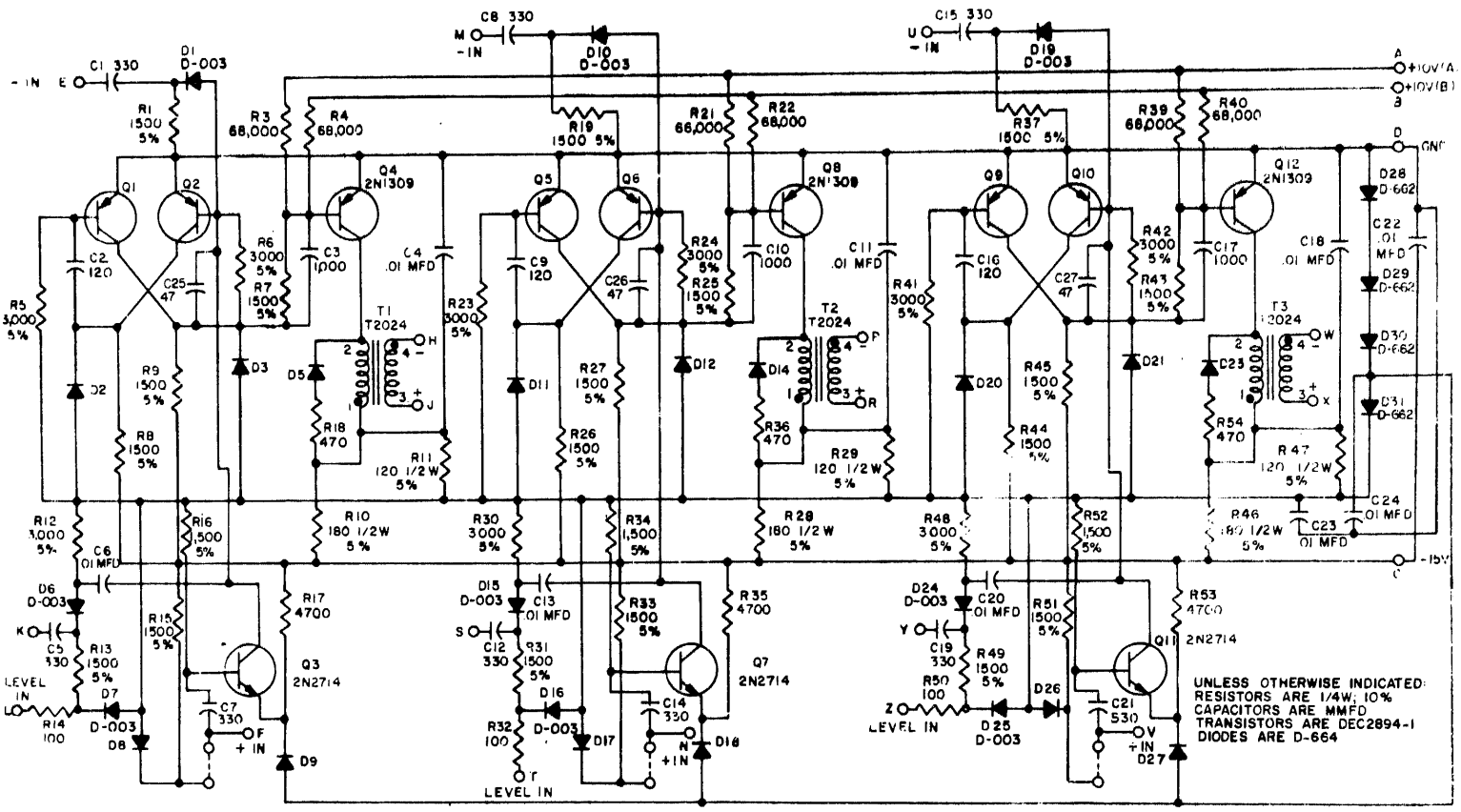


UNLESS OTHERWISE INDICATED
RESISTORS ARE 1/4 W, 10%
CAPACITORS ARE MMFD
DIODES ARE D-664

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
2N2714	2N2714		
DEC 2894 4	DEC 2894		
2N1309	2N1309		
D-664	IN3606		
D-662	IN648		

Pulse Amplifier

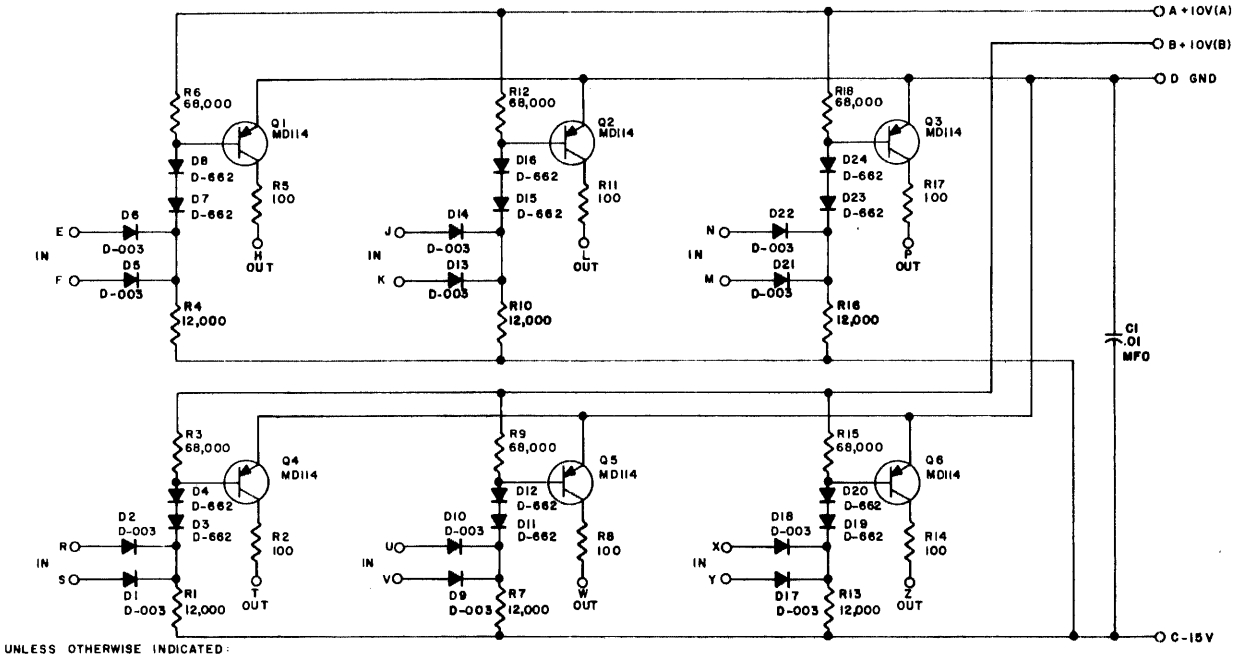
RS-4604



TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-1	DEC 2894	D-003	IN994
2N2714	2N2714		
2N1309	2N1309		
D-664	IN3606		
D-662	IN646		

Pulse Amplifier

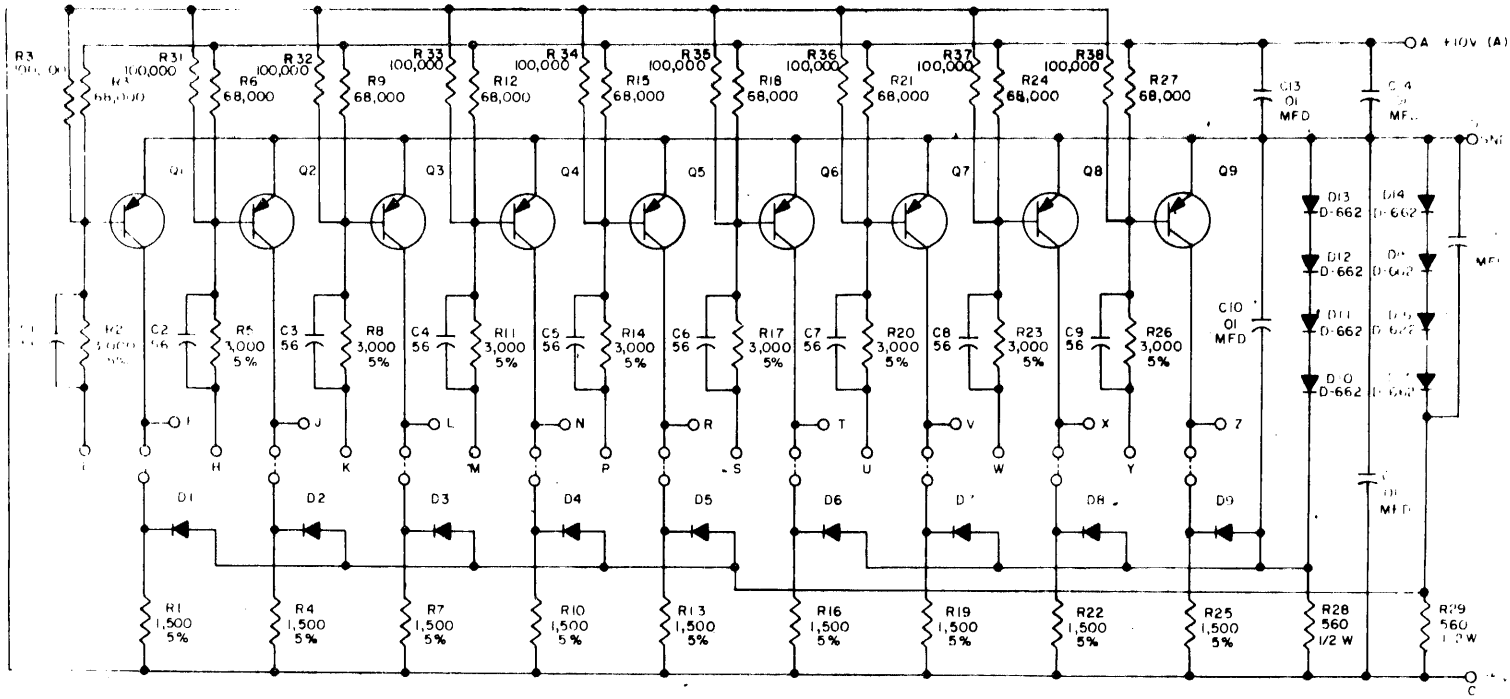
RS-4606



TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
MD114	2N1409A		
D-003	IN994		
D-662	IN646		

I/O Bus Driver

RS-4657

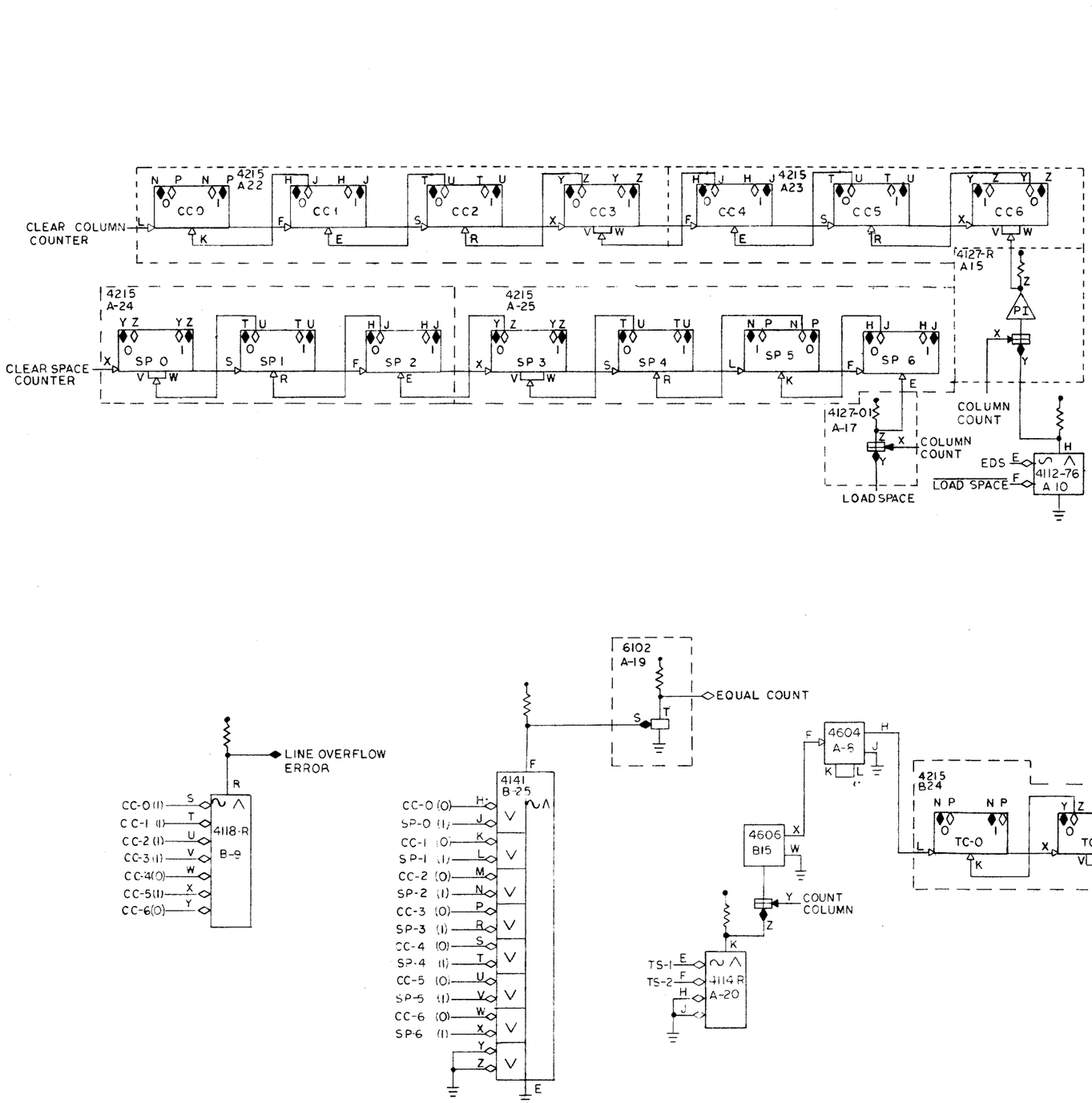


UNLESS OTHERWISE INDICATED
RESISTORS ARE 1/4 W, 10%
CAPACITORS ARE MMFD
TRANSISTORS ARE DEC 2894-1
DIODES ARE D 664

TRANSISTOR & DIODE CONVERSION CHART			
DEC	EIA	DEC	EIA
DEC 2894-1	2N[illegible]4		
D-664	IN[illegible]4		
D-662	IN[illegible]		

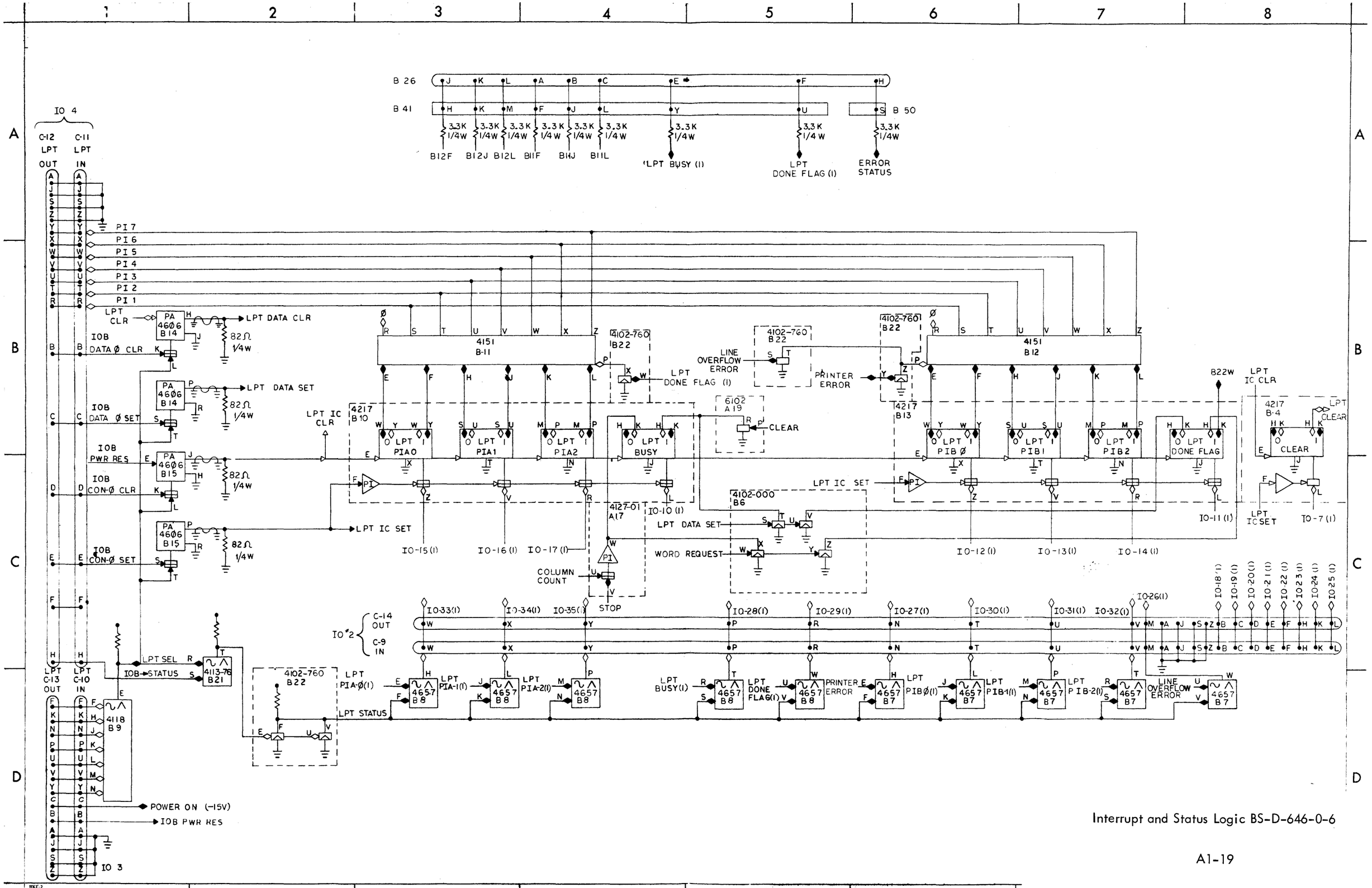
Inverter

RS-6102

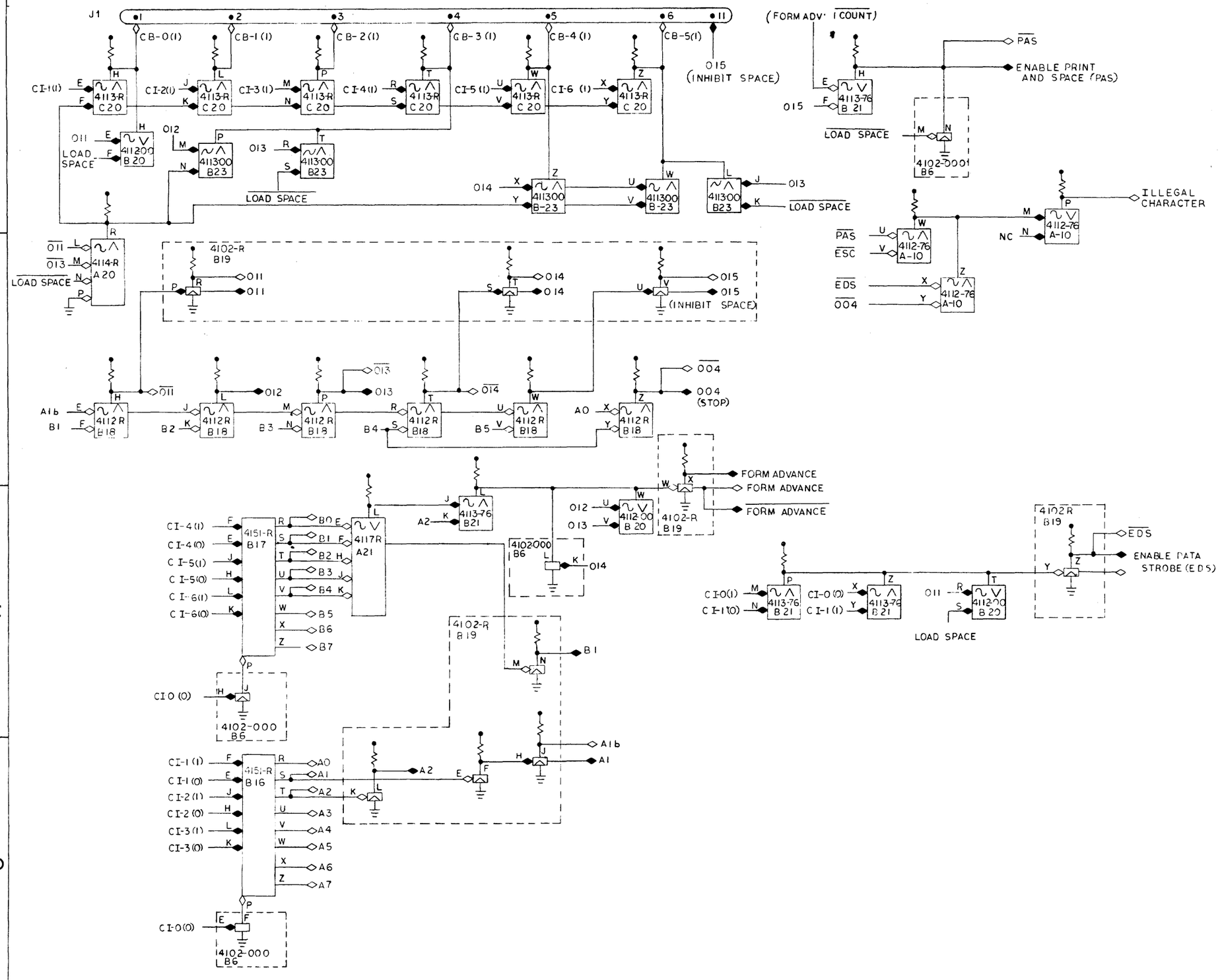


Counters and Tab Selection BS-D-646-0-5

A1-17



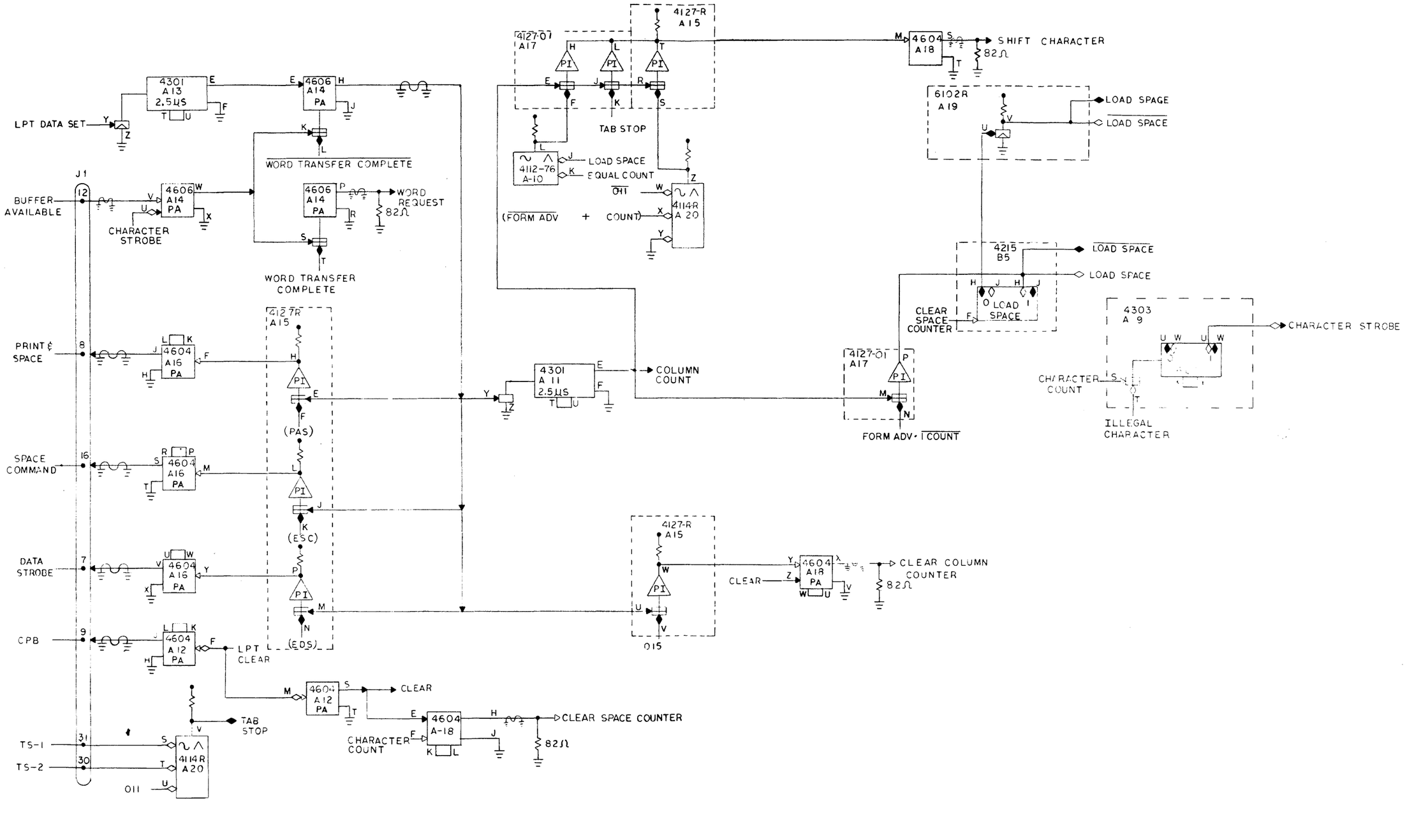
Interrupt and Status Logic BS-D-646-0-6



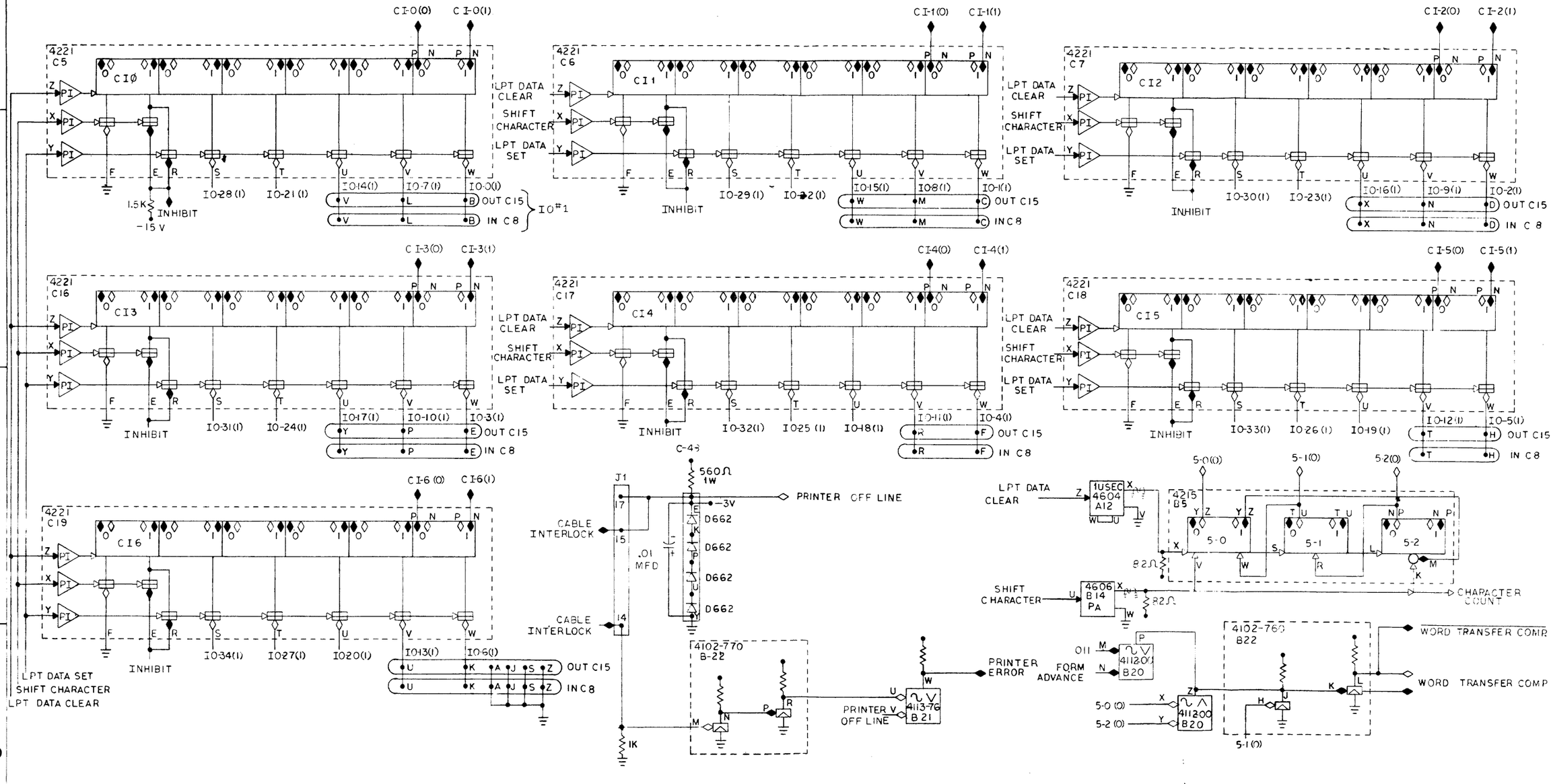
CONTROL CHARACTERS

HT-011	
LF-012	(VF 6) (TR-8)
VT-013	(VF 5) (TR-7)
FF-014	(VF 7) (TR-1)
CR-015	
DC_0-020	(VF 0) (TR-2)
DC_1-021	(VF 1) (TR-3)
DC_2-022	(VF 2) (TR-4)
DC_3-023	(VF 3) (TR-5)
DC_4-024	(VF 4) (TR-6)
EOT-004	

Control Character Decoding BS-D-646-0-7



Character Transfer BS-D-646-0-8



Character Input Register BS-D-646-0-9

JACK [X] PLUG [ ] FEMALE [X] MALE [ ]			LOCATION, LENGTH, ROUTE **LOCATED RIGHT SIDE OF PANEL C, LENGTH AS NEEDED**				
COLOR	PIN	PIN	NAME	COLOR	PIN	PIN	NAME
WHT	C-20H	1	CB0	WHT	A-19F	26	TCB-1(1)
	C-20L	2	CB1		A-19J	27	TCB-0(1)
	C-20P	3	CB2		A-19L	28	TCB-3(1)
	C-20T	4	CB3		A-19N	29	TCB-2(1)
	C-20W	5	CB4		A-20T	30	TS-2
	C-20Z	6	CB5		A-20S	31	TS-1
GRY TWP	A-16V	7	DATA STROBE			32	
	A-16J	8	PRINT & SPACE			33	
	A-12J	9	CPB			34	
BLK	GND	10	CBF			35	
WHT	B-18W	11	INHIBIT SPACE			36	
GRY TWP	A-14V	12	BUFF.AV(PULSE)			37	
		13	BUFF AV(LEVEL)			38	
WHT	B-22M	14	CABLE INTERLOCK			39	
WHT	C-43E	15	CABLE INTERLOCK			40	
GRY/TWP	A-18S	16	SPACE COMM.			41	
WHT	C-43E	17	PRINTER ERROR (LEVEL)			42	
		18				43	
		19				44	
		20				45	
		21				46	
		22				47	
		23			-15 Volt	48	-15V TURN ON
		24			SIGNAL GND	49	SIGNAL GND
		25			CHASSIS GND	50	CHASSIS GND

JACK ☐ PLUG ☒ FEMALE ☒ MALE ☐	LOCATION, LENGTH, ROUTE B-26			
COLOR	PIN	PIN	NAME	REMARKS
W/BLK (X)	B41F	A	LPT PIAØ	
W/BRN (Z)	B41J	B	LPT PIA1	
W/RED (R)	B41L	C	LPT PIA2	
W/ORN (O)		D		
W/YEL (Y)	B41Y	E	LPT BUSY	
W/GRN (N)	B41U	F	LPT DONE FLAG	
W/BLU (B)	B39S	H	ERROR STATUS	
W/VIO (V)	B41H	J	LPT PIBØ	
W/GRY (G)	B41K	K	LPT PIB1	
WHT (W)	B41M	L	LPT PIB2	
W/BLK (X)		M		
W/BRN (Z)		N		
W/RED (R)		P		
W/ORN (O)		R		
W/YEL (Y)		S		
W/GRN (N)		T		
W/BLU (B)		U		
W/VIO (V)		V		
W/GRY (G)		W		
WHT (W)		X		
		Y		
		Z		

Indicator Cable for Interrupt and Status Logic

CL-A-646-0-13

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
								4604	4303	4112–76	4301	4604	4301	4604	4127-R	4604	4127-01	4604	6102-R	4114-R	4117-R	4215	4125-R	4215	4125
A								TC CLEAR D-5-C4	CHARACTER STROBE	CC6 D-5-A5 SHIFT CHAR D-8-A5 ILLEGAL CHARACTER ESC D-5-A7 ILLEGAL CHARACTER	COLUMN COUNT	CPB D-8-C2 CLEAR D-8-D3 CLEAR 5 0-2 D-9-C6	CONTROL STROBE D-8-A2	CONTROL STROBE D-8-A2 WORD REQUEST D-8-A2 BUFFER AVAILABLE D-8-A2	PRINT & SPACE D-8-B2 SPACE COMMAND D-8-C2 DATA STROBE D-8-C2 SHIFT CHARACTER D-8-A4 CLEAR COLUMN COUNTER D-8-C4 CC6 D-5-B5	PRINT & SPACE D-8-B2 SPACE COMMAND D-8-C2 DATA STROBE D-8-C2 DATA STROBE D-8-C2	SHIFT CHARACTER D-8-A4 SHIFT CHARACTER D-8-A4 FORM FEED D-8-B5 TAB(1) D-8-B5 LPT BUSY D-6-C4 SP-6 D-5-B2	CLEAR SPACE COUNTER D-8-A5 SHIFT CHARACTER D-8-A6 CLEAR COLUMN COUNTER D-8-C5	TCB-1(1) D-5-B6 TCB-0(1) D-5-B6 TCB-3(1) D-5-B6 TCB-2(1) D-5-C6 LPT BUSY D-6-C5 EQUAL COUNT D-5-B4 LOAD SPACE D-8-A6 OT4 + COUNT D-5-A7 FORM ADV 1 COUNT D-5-A8	TC CLEAR D-5-D4 CB-0 (1) D-7-A1 TAB STOP D-8-D2 SHIFT CHARACTER D-8-B4	ENABLE SPACE COMMAND D-7-C3 CC-1-4 D-5-A5 FORM ADV COUNT D-5-A8	CC 1 D-5-A3 CC 0 D-5-A2 CC 2 D-5-A1 CC 3 D-5-A2	CC 4 D-5-A4 CC 5 D-5-A4 CC 6 D-5-A5	SP 2 D-5-A4 SP 1 D-5-A4 SP 0 D-5-A5	SP 6 D-5-B2 SP 5 D-5-B3 SP 4 D-5-B4 SP 3 D-5-B4
				4217	4215	4102-000	4657	4657	4118-R	4217	4151	4151	4217	4606	4606	4151-R	4151-R	4112-R	4102-R	4112-00	4113-76	4102-760	4113-74	4215	4141-R
B				CLEAR D-6-B8 D-9-C7 D-9-C7 D-9-C6	LOAD SPACE D-8-B6 5-2 5-1 5-0	A 0-7 D-7-D2 B 0-7 D-7-C2 FORM ADV FORM ADV D-7-C3 ENABLE PRINT & SPACE D-7-A5 OT4 + COUNT D-5-A8 LPT BUSY D-6-C5 LPT DONE FLAG D-6-C5 LPT BUSY D-6-C5 LPT DONE FLAG D-6-C5	10-27 (1) D-6-D6 10-30 (1) D-6-D6 10-31 (1) D-6-D7 10 32(1) D-6-D7 10 26(1) D-6-D8	10-33 (1) D-6-D3 10-34 (1) D-6-D3 10-35 (1) D-6-D4 10-28 (1) D-6-D5 10-29 (1) D-6-D5	LPT SEL D-6-D1 LINE OVERFLOW ERROR D-5-C1	LPT BUSY D-6-B4 LPT PIA$_2$ D-6-B4 LPT PIA$_1$ D-6-B3 LPT PIA$_0$ D-6-B3	PI 1-7 D-6-B3	PI 1-7 D-6-B7	LPT DONE FLAG D-6-B8 LPT PIB$_2$ D-6-B7 LPT PIB$_1$ D-6-B7 LPT PIB$_0$ D-6-B6	LPT DATA CLEAR D-6-B1 LPT DATA SET D-6-B1 CHARACTER COUNT D-9-C6	LPT IC CLEAR D-6-C1 LPT IC SET D-6-C1	A 0-7 D-7-D2	BO 0-7 D-7-C2	OT1 D-7-B1 012 D-7-B1 OT3 D-7-B2 OT4 D-7-B3 015 INHIBIT SPACE D-7-B3 004 STOP D-7-B3	A1 D-7-D3 A1 D-7-D3 A2 D-7-D2 B1 D-7-C3 011 D-7-B2 014 D-7-B3 015 D-7-B4 FORM ADV D-7-C4 ENABLE DATA STROBE D-7-C6	CB-0 (1) D-7-A1 ESC D-5-A7 WORD TRANS. COMP. D-9-D6 ENABLE DATA STROBE D-7-C6 ENABLE SPACE CONTROL D-7-C3 WORD TRANS. COMP. D-9-D7	ENABLE PRINT & SPACE D-7-A5 ENABLE SPACE COMMAND D-7-C3 ENABLE DATA STROBE D-7-C4 LPT STATUS D-6-D2 ERROR PRINT D-9-D5 ENABLE DATA STROBE D-7-C5	LPT STATUS D-6-D2 WORD TRANS. COMP. D-9-D7 WORD TRANS. COMP. D-9-D7 ERROR STATUS D-9-D4 ERROR STATUS D-9-D4 LINE OVERFLOW ERROR D-6-B5 LPT STATUS D-6-D2 LPT DONE FLAG D-6-B4 ERROR STATUS D-6-B4	CB-0 (1) D-7-A1 CB5-0 D-7-A4 CB-3(1) D-7-A2 CB-5(1) D-7-A4 CB-4(1) D-7-A3	TC-0 TC-1 TC-2 TC-3	EQUAL COUNT
					4221	4221	4221	IN	IN	IN	IN	OUT	OUT	OUT	OUT	4221	4221	4221	4221	4113-R					
C					CI 0 D-9-A3	CI 1 D-9-A5	CI 2 D-9-A8									CI 3 D-9-B3	CI 4 D-9-B5	CI 5 D-9-B8	CI 6 D-9-C3	CB-0 (1) D-7-A1 CB-1 (1) D-7-A2 CB-2 (1) D-7-A3 CB-3 (1) D-7-A3 CB-4 (1) D-7-A3 CB-5 (1) D-7-A4					

APPENDIX 2

ASCII LINE PRINTER CODE

@	100	Y	131	2	162
A	101	Z	132	3	163
B	102	[	133	4	164
C	103	\	134	5	165
D	104	]	135	6	166
E	105	↑	136	7	167
F	106	←	137	8	170
G	107	space	140	9	171
H	110	!	141	:	172
I	111	"	142	;	173
J	112	#	143	<	174
K	113	$	144	=	175
L	114	%	145	>	176
M	115	&	146	?	177
N	116	(apos) '	147	End of Transmission (EOT)	004
O	117	(	150		
P	120	)	151	Horizontal Tab	011
Q	121	*	152	Line Feed	012
R	122	+	153	Vert Tab	013
S	123	(comma) ,	154	Form Feed	014
T	124	-	155	Carriage Return	015
U	125	.	156	DCØ (SKIP 1 line)	020
V	126	/	157	DC1 (SKIP 2 lines)	021
W	127	Ø	160	DC2 (SKIP 3 lines)	022
X	130	1	161	DC3 (SKIP 6 lines)	023
				DC4 (SKIP 11 lines)	024

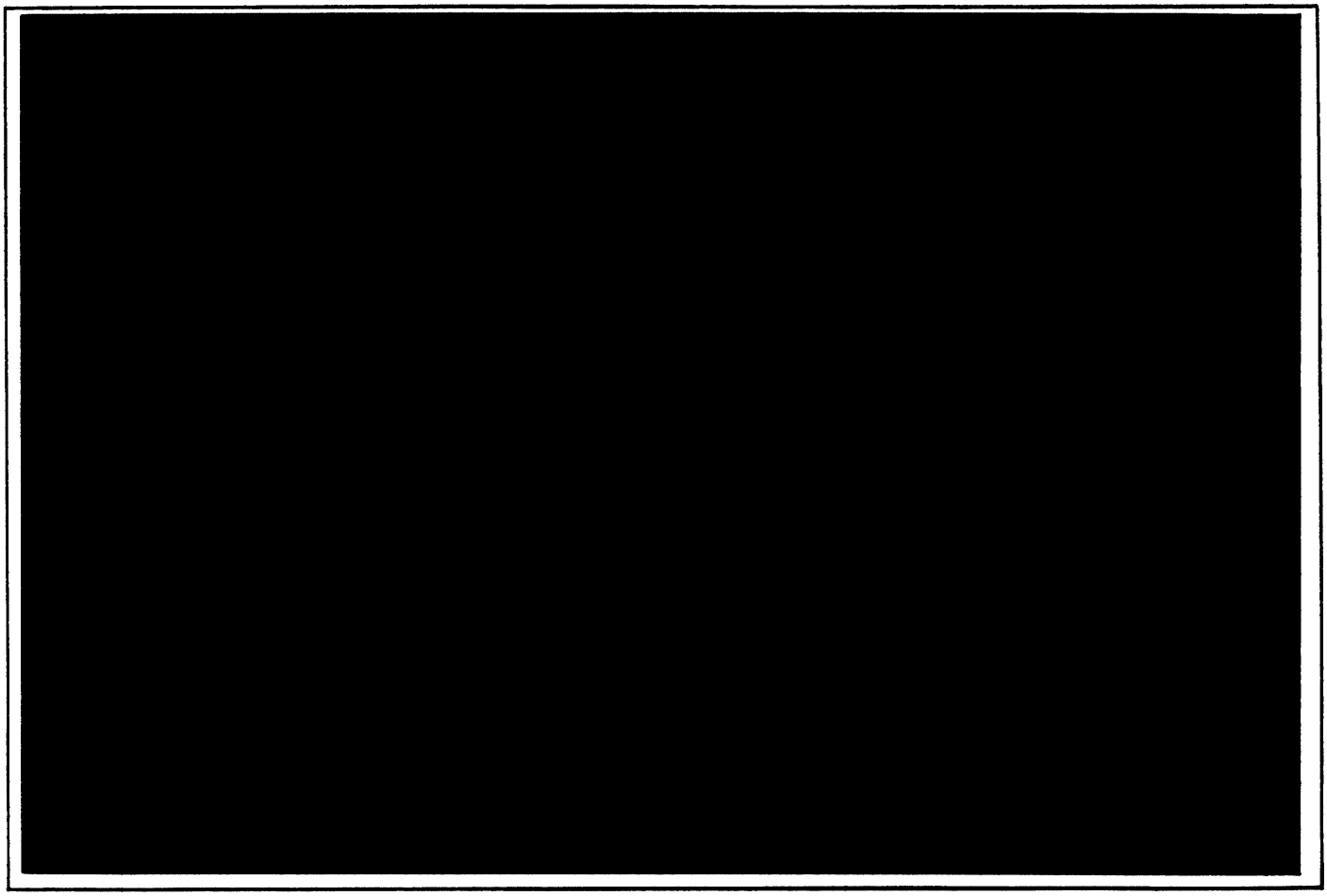
All others are ignored

5132 PRINTED IN U. S. A. 2-2/65

F-67 (166)

ARITHMETIC PROCESSOR 166
INSTRUCTION MANUAL

VOLUME 1



DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

J. Holloway

F-67(166)

PDP-6 ARITHMETIC PROCESSOR 166
INSTRUCTION MANUAL

VOLUME 1

DIGITAL EQUIPMENT CORPORATION • MAYNARD, MASSACHUSETTS

PREFACE

This instruction manual is published in two volumes to aid personnel in the operation and maintenance of the Arithmetic Processor Type 166 and four of the more common PDP-6 input-output devices: Paper Tape Reader Type 760, Paper Tape Punch Type 761, Teletype Keyboard-Printer Type 626, and Card Reader Type 461. Maintenance information for the in-out devices is confined primarily to those portions of DEC manufacture; separate manuals for the devices themselves are furnished with the system.

The first three chapters present a general description of the system and its operation. Chapter 1 lists the operating specifications and describes the physical and electrical characteristics of the system. Chapter 2 provides a general description of system organization at the block diagram level, explaining what the system does rather than describing the circuit hardware involved in the various functions. This chapter also describes the number system and instruction formats used in the Type 166 Processor. Chapter 3 explains the use of all controls and indicators on the operator control panels and in-out devices, and outlines basic operating procedures.

The next five chapters present a complete, detailed description of the system logic. Chapter 4, Drawing Conventions and Flow Charts, discusses PDP-6 documentation and describes the symbols and terminology used in the logic drawings and flow charts. This chapter also escorts the reader through the flow charts in sequence, so that he may better understand the flow of operations in the processor, and discusses in detail some sequences that neither appear as coherent hardware units in the logic drawings nor are obvious from the flow charts. The next two chapters in this group describe the hardware for the main control sequence and logical and arithmetic processing; the final two describe the processor interfaces with the memory bus and the in-out bus. Also included in the last chapter are the control units for four common in-out devices. The reader is strongly advised not to embark upon any logic chapter in this or any other PDP-6 system manual without first gaining a thorough understanding of the material presented in Chapter 4.

Chapter 9 contains information useful in maintaining the system, including a discussion of maintenance operation, maintenance programs, and preventive and corrective maintenance.

Following Chapter 9 are appendixes on engineering drawings and spares, a glossary, and several convenient tables. All logic drawings and flow charts referred to in the text are in Volume 2; all other figures are interleaved with the text.

FOREWORD

PDP-6 is a general-purpose, digital computing system consisting of processors, memories, and input-output devices, each of which has independent internal timing. Processors in a system may share memories and input-output equipment; the memories themselves may have different speeds.

A central processor, usually the Type 166 Arithmetic Processor, performs arithmetic and logic operations and governs the movement of information between memory and peripheral devices. The Type 166 includes an executive system that allows a number of programs, each restricted to a definite area in core, to share processor time. The central processor uses two busses for system intercommunication, one to the memory system, the other to its input-output devices. A system may contain any number of central processors, each with a memory bus and an in-out bus. The memory buses permit the memory complex to accommodate several processors—as many as four may address a single memory module. In order to deposit or retrieve information, the processor supplies an address and requests a memory cycle. Upon accepting the request, the addressed memory times its own cycle and furnishes the appropriate response to the processor.

Through the in-out bus the processor controls all information transfers to and from the peripheral equipment. A priority interrupt system in the processor allows a device to signal when it needs service so that the processor is free during the actual input-output time. One of the units that may be connected to the bus is the Type 167, an autonomous drum processor which supplies direct memory access for high-speed devices such as drums, discs, magnetic tape, and displays. In addition to its in-out bus connection to the central processor, the drum processor is itself connected to the memory system via its own memory bus, and has its own smaller scale in-out bus through which it may govern up to three input-output control units. Although the central processor must provide initial conditions and commands, the drum processor then operates independently, so large blocks of information may be transferred between an in-out device and a memory without reducing central processor efficiency.

All PDP-6 memories store words of 36 bits but may be of different sizes and speeds. Core memories usually have core banks of 8192 or 16,384 words. Cycle times for reading from and writing back into memory are typically 2 and 5 μsec, although in each case access time is much shorter: when reading, the processor need wait only until data is available; when writing, only until data is accepted. A fast flip-flop memory, with access time less than 1/2 μsec, is normally used instead of the bottom 16 locations in core.

The instruction format allows the basic instructions to address one of 262,144 locations in memory for an operand, one of 15 index registers for modifying the memory address, and one of 16 accumulators for a second operand. Instruction results may be stored in an accumulator, in memory, or in both. In-out instructions govern the transfer of data in both directions over the in-out bus, the transfer of control information, including priority interrupt channel assignments, to the peripheral equipment, and the gathering of status information from that equipment. In addition to addressing a memory location and an index register, an in-out instruction may address one of 128 devices, two of which are the priority interrupt system and the processor itself.

For further information on the overall system, refer to PDP-6 Programming (DEC publication K-06), which also describes system software and discusses programming for the processor and most in-out devices. Maintenance documentation for the system is provided by a series of manuals. This one discusses system maintenance for the Type 166 Arithmetic Processor and several common in-out devices, others cover the several types of memories that may be used in a PDP-6 memory system, and still others treat the drum equipment, magnetic tape equipment, DECtape, and other in-out devices. A separate circuit manual discusses circuit maintenance and describes most standard circuits including all those used in the equipment described in the present manual and all logic circuits used in the memories. Descriptions of specialized circuits, such as those associated with the core stack, reading and writing on magnetic tape, and the like, are included in the appropriate system manuals.

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

CONTENTS (continued)

CONTENTS (continued)

ILLUSTRATIONS

CHAPTER 1

INTRODUCTION

The Type 166 is a general-purpose central processor that performs all of the arithmetic, logical, executive, and internal data transmission operations in a PDP-6 system. It also controls all transfers of data between memory and peripheral equipment, although in many cases it may provide control merely by supplying system commands and initial conditions to an in-out processor. It contains two bus interfaces, one for connection to memory, the other to the input-output system.

Except for certain control information held permanently in the processor, the state of the processor resides entirely in memory. The only information carried over by the processor from one instruction to the next is the program count, flags, and information for a user mode which allows a number of programs, each restricted to a definite area in core, to share computer time. Besides operating on a stored program, the processor must retrieve all operands for every instruction, and all data and results of computations are stored at the completion of an instruction. Thus the arithmetic registers in the processor contain information only during actual processing and the registers used for address modification are the same as those used for computations within a single instruction. The accumulators, 15 of which double as index registers, actually occupy the bottom 16 memory locations and are usually contained in a fast memory. Most basic instructions have three addresses which select an accumulator, a memory location (which may be another accumulator), and an index register for memory address modification. All instructions may use multiple-level indirect addressing and some may use a single address to call two adjacent accumulators for processing double-length operands. With a single instruction, the processor is capable of performing a full-word or half-word transfer, a block transfer, or the manipulation of a character (byte) of variable size. The processor includes hardware for performing Boolean functions, shift operations, both fixed- and floating-point arithmetic, jumps, logical and arithmetic comparisons, and a variety of modification and testing instructions.

In addition to standard instruction operations, the hardware also includes a program-assignable priority interrupt system through which an external device or an internal condition can interrupt

the normal program sequence; a number of flags that allow checking of various conditions and facilitate double-precision arithmetic; and memory protection and relocation registers that allow an executive routine to assign a specific area in core to each user program.

1.1 OPERATING SPECIFICATIONS

All timing in the Type 166 is completely asynchronous, and all processing is done in parallel except for a few extremely fast serial functions, such as carry propagation in the main arithmetic register and in dc adders that relocate memory addresses. Information handled by the processor has the following characteristics:

Word Length	36 bits
Instruction Format	
Basic	Instruction code, 9 bits Accumulator address, 4 bits Indirect, 1 bit Index register address, 4 bits Memory address, 18 bits
Input-Output	Instruction code, 6 bits Device code, 7 bits Indirect, 1 bit Index register address, 4 bits Memory address, 18 bits
Internal Number System	Binary
Negative Representation	2's complement
Number Format	
Fixed Point	Sign, 1 bit; magnitude, 35 bits
Floating Point	Sign, 1 bit; exponent, 8 bits; fraction, 27 bits

The time required for execution of any particular instruction varies tremendously because of the completely asynchronous operation. The basic operations, such as addition in the arithmetic register or a sequence of shifts controlled by the shift counter, are performed by built-in

hardware subroutines. These are called whenever necessary either from the main instruction sequence or by special sequences such as byte manipulation, block transfer, floating add-subtract, divide, etc., which are in turn entered from the main instruction sequence. Even at the level of individual events, the execution time may vary; for example in the basic addition or subtraction subroutine, the complement function and the parital addition each require 100 nanoseconds but carry propagation, which is serial, depends upon the number of carries needed. Similarly on a larger scale, multiplication and division are performed by a series of additions and subtractions and the time required for such a major sequence depends upon the number of times it must call various subsequences. Most processor control functions involved in the retrieval and setup of instructions, and retrieval and storage of operands take a negligible amount of time when compared to memory access time. Exact instruction execution times may be determined from the flow charts included in Chapter 4. For each memory access, the processor must first check for memory protection and relocation and then wait until the addressed memory is free; the time required for access once the memory is free depends upon the type of memory.

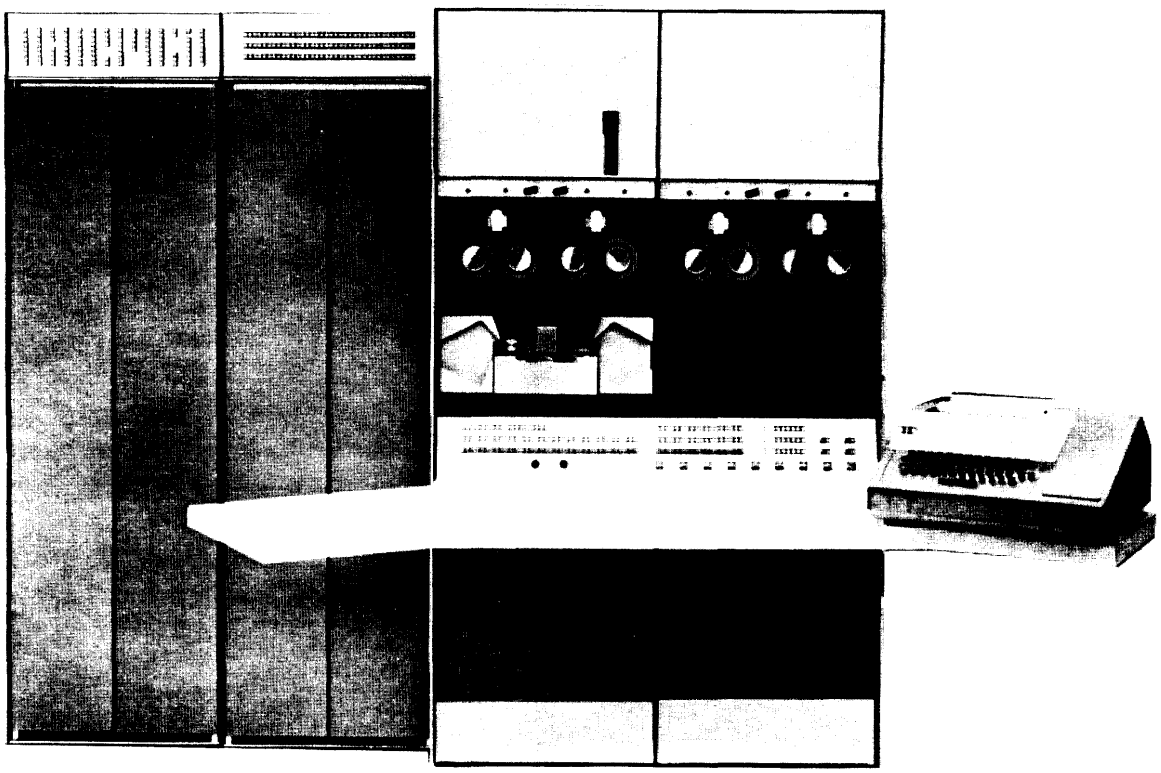
The processor must set up all transfers of control information and data to and from the peripheral equipment; but since a device can signal the processor by means of the priority interrupt system when it requires service, no processor time need be lost in waiting, and processor and peripheral equipment can operate in parallel. Every transfer over the I/O bus does, however, require 2.5 microseconds. The four I/O devices included in this manual have the following operating specifications.

Paper Tape Reader	400 8-bit characters per second
Paper Tape Punch	63.3 8-bit characters per second
Keyboard-Printer	10 8-bit characters per second
Card Reader	200 80-column cards per minute

1.2 PHYSICAL CHARACTERISTICS

Most DEC equipment is housed in steel bays with aluminum control panels. The arithmetic processor with its console uses four such bays bolted together. The front of each bay can accommodate up to twelve 19 inch by 5-1/4 inch panels lettered A to N from top to bottom

(skipping G and I). Bays 1 and 2, which house the bulk of the processor logic, each have an indicator panel at the top with the remainder of the bay occupied by eleven standard logic panels mounted behind double doors. Each mounting panel can hold up to 25 DEC system plug-in modules numbered from left to right when viewed from the front. At the center of bays 3 and 4, which hold the console, is the main operator control panel. Usually, a tape reader is mounted in the left console bay just above the control panel and a paper tape punch at the top (the front panel of the drawer containing the punch has an opening for removing fan-folded tape). At the top of bay 4 behind a metal cover are the marginal check controls and an indicator panel for the standard in-out equipment. The remaining space above the control panel may be used for DECtapes, displays, or other equipment. The space below the console table can hold up to eight logic mounting panels, two of which are used for the arithmetic processor; the remainder may be used to hold the control logic for some of the in-out equipment. Inside the double doors at the back of each bay is an inner plenum door, on which are mounted the required power supplies and power control panels.



Arithmetic Processor Type 166

Physical dimensions are as follows:

Arithmetic Processor

Height	69-1/2 inches
Width	100 inches
Depth	60 inches (75 inches with rear plenum doors open)
Weight	1300 pounds including tape reader and punch

Keyboard-Printer, Teletype Model 35 KSR

Height	38-1/2 inches
Width	20 inches
Depth	24 inches
Weight	151 pounds

Card Reader, Burroughs B122

Height	50 inches
Width	48 inches
Depth	29 inches
Weight	200 pounds

Intake fans at the bottom of every bay cool the logic modules by blowing air out between them. All equipment described in this manual can operate in an ambient temperature range from 50° to 100°F. However, if the installation includes temperature sensitive equipment such as magnetic tape, air conditioning is required. The floor should be capable of supporting approximately 150 pounds per square foot.

1.3 ELECTRICAL CHARACTERISTICS

All PDP-6 equipment uses standard line power at 105 to 125 vac, 60 cycles, single phase. All power cables use Hubbell Twist-Lok connectors; both cable and connector are rated at 30 amperes. The arithmetic processor, console and console-mounted standard in-out equipment together use two lines and two power controls. The main power control is usually a Type 829

or Type 835; it provides ac to all of the power supplies for the processor logic and any in-out control logic mounted below the console operator panel. The dc voltages required by the logic are +10 and -15 volts. Some power supplies provide both, others provide only the negative voltage. In some cases, two -15 volt supplies may be connected in series to provide -30 volts to solenoid drivers for in-out equipment. One -15 volt line turns on a secondary power control (Type 811 or 834) that provides ac to the motors for the reader, punch, and keyboard-printer. For the punch, ac is fed through a Type 823 Power Control (mounted directly on the punch) that allows the processor logic to control application of punch motor power. Another -15 volt signal is applied to external power controls (usually Type 811 or 834) via the in-out bus to turn on the peripheral equipment. Still another -15 volt signal turns on the power controls (usually Type 836) in the memories. This last dc turnon signal is not sent via the memory bus; instead it is included in a small bus that also carries marginal check voltages from a variable power supply located in the console.

Current consumption of the equipment described in this manual is as follows:

Arithmetic Processor, including console and console-mounted in-out logic	25 amperes, 1900 watts Turnon surge, 40 amperes
Tape Reader	1.8 amperes, 150 watts Turnon surge, 2.8 amperes
Tape Punch	1.85 amperes, 65 watts Turnon surge, 9 amperes
Keyboard-Printer	2.6 amperes, 140 watts Turnon surge, 7 amperes
Card Reader	1.5 amperes, 145 watts Turnon surge, 7 amperes

All PDP-6 logic is solid state; transistors and diodes operate on static logic levels of 0 and -3 vdc (tolerances are 0 to -.3 volts and -2.5 to -3.5 volts). Most logic modules include an internal supply to derive the negative logic level from the -15 volt input. PDP-6 logic uses pulse timing almost exclusively. Pulses are of either polarity depending upon gate input requirements. Pulse amplitude is 2.5 volts from ground with tolerances of +2.3 to +3.0 volts and -2.3 to -3.5 volts. Pulses at inverter outputs may be from ground to -3 volts or vice versa. Pulse widths may be 1 microsecond or 400, 70, 40, or 25 nanoseconds depending upon module type and application. Occasionally, an input may be triggered by a level transition instead of a pulse.

CHAPTER 2

SYSTEM FUNCTION

The logical configuration of the Arithmetic Processor Type 166 is shown in Figure 2-1. Large blocks at the top and bottom represent the buses that connect the processor to the input-output equipment and the memory; the figure shows all connections to these buses, both data and control (each connection is labeled with the number of physical lines required). Between the buses is a block diagram of the processor showing all registers, with transfers among them represented by lines connecting the register blocks. Each block is labeled with both the name of the register and the number of bits. The registers vary considerably in size. Data registers have 36 bits, those that handle only addresses have 18. Registers that handle floating-point exponents have 9 bits; those that control memory protection and relocation have 8; 7-bit registers govern the requesting and granting of program sequence breaks through the priority interrupt system. The figure does not show the control lines within the processor, but all control pulses for each register are written beside the corresponding block. For an explanation of signal names, refer to the discussion of signal notation in Chapter 4.

The heart of the arithmetic processor is a set of three full-size registers which handle all data transfers and in which are performed all logical and arithmetic operations. These are arithmetic register AR, multiplier-quotient register MQ, and memory buffer MB. All transfers between processor and memory are made through MB, transfers between processor and peripheral equipment are made via AR. At the console, the operator may communicate with the system through a register of data switches for sending information in via AR and a register of memory indicators for displaying memory words via MB. MB takes part in all data transfers, but in logical and arithmetic operations it usually plays a passive role by holding an operand which is combined with the operand in AR, the result appearing in AR. MQ serves primarily as an extension of AR for handling double-length operands.

The processor performs a program by executing instructions retrieved from consecutive memory locations as counted by the program counter PC, although the program may change its own sequence by changing the address in PC. To gain access to memory for retrieval or storage,

the processor requests a memory cycle and supplies an address over the bus from memory address register MA. This register also serves as a control link to the operator in that the system receives addresses via MA from the address switch register on the console. When a word is retrieved at MB as an instruction, its left half passes to the instruction register IR which controls further retrieval of index registers and accumulators, and which is decoded to govern the actual execution of the instruction. The right half in MB is the memory operand address, which may be modified in AR by the contents of an index register.

The operands are brought from memory to AR, MB, and sometimes MQ, for whatever operations are necessary for the execution of the instruction. In some of the more complicated operations, these full-size registers are aided by the shift counter SC and the floating-exponent register FE. SC is used for subsidiary computations such as the calculation of the exponent in floating-point arithmetic, and it also controls the count of any operation performed by the repetition of basic steps in the three main registers. FE is used only for temporary storage of preliminary results while SC is controlling the remainder of the operation.

Besides the registers that enter into the regular execution of the program and its instructions, the processor contains an executive system and a priority interrupt system. The first contains two registers for memory protection and relocation. When the processor is in executive mode, all instructions and all memory are available to the program. In user mode, a number of programs share processor time with each program restricted to a specific area in core, and certain instructions are illegal. All programs are written using the lower addresses but these are not supplied directly to memory from MA. In requests for memory access, the address in MA is compared with the contents of the memory protection register PR. The number in this register defines the size of the block available to the program and prevents it from addressing any location outside its assigned area. The address is then changed to one within the assigned area by adding a constant contained in the relocation register RLR to the address in MA.

The priority interrupt system allows peripheral devices and certain conditions internal to the processor to interrupt the normal program sequence. There are seven interrupt channels through which sequence breaks are allowed on a priority basis as governed by three control registers. The first register allows the program to turn individual channels on and off; the second synchronizes break requests to internal processor timing and assigns the break to the highest

priority channel that has been recognized; the third holds the break and prevents further interruption by lower priority channels. A break is executed by performing the instruction in a particular memory location associated with each channel. The assignment of channels to devices is entirely under program control; the program may assign several devices to a single channel or give a device no assignment. One of the devices to which the program may assign a channel is the processor itself. For this purpose the processor has an I/O interface containing a number of flags that allow internal conditions to interrupt the sequence; the flags may be sensed and controlled by the program. Through this interface, the processor may also bring information in from the console DATA switches or supply memory protection and relocation information to the executive system.

Timing for all operations in the processor is supplied by asynchronous pulse chains. Processor operation is initiated by means of a special key cycle that supplies timing for events associated with operator intervention at the console and provides entry into the main sequence. When the processor is running, timing is supplied by the main sequence which is repeated for each instruction. The main sequence uses a hierarchy of other sequences—built-in hardware subroutines—which can be called directly by the main sequence or by any sequence of higher rank. Thus the processor operates using many levels of nested sequences; each sequence stops upon calling a lower ranked sequence and restarts upon return from it (although the restart need not be at the point of departure). For example, in a block transfer, the main sequence calls the block transfer subroutine which in turn calls others to perform the necessary arithmetic and obtain memory access.

2.1 PROGRAMMING

The first 16 locations in memory function as accumulators, index registers, or ordinary memory locations. Their particular functions are determined entirely by the processor under program control, but they differ from the remainder of the memory system only in that they are usually contained in a fast memory. All 16 locations may be used as accumulators or ordinary memory, but only locations 1 to 17 may be addressed as index registers because a zero index register address specifies no indexing. Since ordinary memory addresses are 18 bits, only the information contained in the right half of an index register is actually used for address modification.

In systems that include a fast memory, it replaces the first 16 core locations (which normally hold a readin loader) for normal processor operations—operations in the reading area can be initiated only from the console; and once an instruction has been taken from outside this area, it again becomes inaccessible to the program. In all systems locations 40 and 41 are used for programmed operators, 42 to 57 are used by the priority interrupt system—a programmer should be wary of using these locations for other purposes.

The logic descriptions contained in this manual assume that the reader is completely familiar with the processor instructions, all of which are described in detail in PDP-6 Programming (K-06). That manual describes the instructions in terms of elements available to the program, i.e., by their effect on accumulators, memory locations, flags, and control registers. For convenience Table 4-1 lists the mnemonic and octal codes for all instructions. The remainder of this section describes the Type 166 number system, instruction format, and flags.

a Number System

For arithmetic computations the hardware is capable of handling numbers in two formats, fixed point and floating point. Both formats use the full 36-bit word; bits are numbered 0 to 35 from left to right. In all numbers, bit 0 represents the sign, 0 for positive, 1 for negative. In floating point, bits 1-8 represent an exponent, bits 9-35 represent a fraction. In fixed point, bits 1-35 represent magnitude, which is usually interpreted as a full fraction with the binary point between sign and magnitude or as an integer with the binary point to the right of bit 35. Of course, the assumed position of the point has no effect on the processor and a program may adopt any consistent point convention. However, the fixed-point hardware does include special provisions to facilitate handling of integers in multiplication and division because these operations make use of double-length numbers.

In ordinary arithmetic, the negative of a number is usually formed merely by changing the sign. This notation is inconvenient for a machine so hardware arithmetic represents negatives by 1's and 2's complements. If x is an n-digit binary number, its 2's complement is $2^n - x$ and its 1's complement is $(2^n - 1) - x$ or equivalently $(2^n - x) - 1$. Subtracting a number from $2^n - 1$ (i.e., from all ones) is precisely equivalent to performing the logical complement, i.e., changing all zeros to ones and all ones to zeros. Therefore, to form the 1's complement,

the processor uses the logical complement—usually referred to merely as the complement—and to form the 2's complement it complements and adds one to the result.

In 1's complement notation, one can read a negative number by attaching significance to the zeros instead of the ones. For 2's complement notation, this simple interpretation is not possible because adding one to the logical complement changes at least the final bit and in fact changes bits as far as the carry propagates. Thus in 2's complement notation, one can read a negative number by attaching significance to the rightmost 1 and attaching significance to the zeros to the left of it. A 2's complement system has the following characteristics. A number all ones represents -1. All even numbers both positive and negative end in 0. In a negative integer, ones may be discarded at the left. In a negative fraction, zeros may be discarded at the right; as long as only zeros are discarded, the number remains in 2's complement form because it still has a 1 which possesses significance. However if a portion including the rightmost 1 is discarded, the remaining part of the fraction is now a 1's complement.

In the Type 166, the 2's complement is used to represent negatives for both fixed and floating numbers. In a positive fixed-point number, the sign bit is 0 and bits 1-35 represent magnitude in normal binary fashion. In a negative, the sign is 1 and the remainder of the word contains the 2's complement of the magnitude of the corresponding positive number. Since 0 is considered one of the positive numbers, the magnitude of the largest positive number is one less than the magnitude of the largest negative number. Fixed-point integers thus have a range from -2^{35} to $2^{35} - 1$; for fractions, the range is -1 to $1 - 2^{-35}$.

The floating-point hardware interprets a computer word as containing an 8-bit exponent and a 27-bit fraction. For a positive number, the sign is 0, as before; but the contents of bits 9-35 are now interpreted only as a binary fraction and the contents of bits 1-8 are interpreted as an integral exponent in excess 128 (200_8) code, i.e., exponents from -128 to +127 are represented by the binary equivalents of 0 to 255. Floating-point zero and negatives are represented in exactly the same way as fixed point: zero by a word containing all zeros, a negative by the 2's complement. The negative thus has 1 for its sign and the 2's complement of the fraction, but since every fraction must contain a 1 unless the entire number is 0 (see below), it has the 1's complement of the exponent code in bits 1-8. Since the exponent is in excess 128 code, an actual exponent x is represented in a positive number by x + 128, in

a negative number by 127 - x. The program, however, need not concern itself with these representations because the hardware compensates automatically. For example, for the instruction that scales the exponent without affecting the fraction, the hardware interprets the scale factor in standard 2's complement form but produces the correct 1's complement result for the exponent.

In all floating-point operations, the hardware assumes that all nonzero operands are normalized and always normalizes a nonzero result. Floating-point numbers are considered to be normalized if the magnitude of the fraction is greater than or equal to 1/2 and less than 1. The test for normalization is thus that either the sign bit differs from bit 9 or bits 9-35 contain 400000000_8, the latter being required for the special case of the fraction -1/2, in which bits 9 and 0 are equal. Floating-point numbers thus have a fractional range in magnitude from 1/2 to $1-2^{-27}$ and an exponent range of -128 to +127. Note that the signed fractional part -1 (i.e., a 1 in bit 0 and all zeros in bits 9-35) satisfies the test for normalization but the hardware always changes it to -1/2 and adjusts the exponent appropriately. The hardware may not give the correct result if the program supplies an operand that is not normalized or that has a zero fraction with a nonzero exponent.

The characteristics of 2's complement notation require additional precautions in floating-point operations and fixed-point fractional multiplication because these have double-length results. The programmer must remember that discarding the low-order part of a double-length negative leaves the high-order part in correct 2's complement form only if the low-order part is null. In floating point, the programmer may request rounding, which automatically restores the high-order part to 2's complement form if it is negative.

b Instruction Format

All but the input-output instructions and programmed operators use a basic format with bit assignment as follows:

0-8		Instruction code
9-12		Accumulator address
13	I	Indirect bit
14-17	X	Index register address
18-35	Y	Memory address

Bits 0-8 determine which operations are executed for the instruction. Bits 9-12 and 14-17 each address the first 16 memory locations which serve as accumulators and index registers. On some occasions, bits 9-12 are used for control purposes instead of addressing an accumulator, for example to address flags. The effective address E of an instruction depends on the values of I, X, and Y. The contents of index register X (zero X specifies no indexing) are added to Y to produce an address. If I is 0, this address is used as the effective address; if I is 1, this address is indirect and is used to retrieve another word. The new word is processed in exactly the same manner as above, i.e., X and Y are used to determine the effective address if I is 0; otherwise, they are used to retrieve another word. The process continues until a word is found in which I is 0. This calculation using I, X, and Y is carried out for all instructions even when E is to be used as an operand or control information instead of a memory address.

IOT instructions (designated by three ones in bits 0-2) have the following bit assignment:

0-2		111
3-9		Device code
10-12		Instruction code
13	I	Indirect bit
14-17	X	Index register address
18-35	Y	Memory address

Bits 3-9 address an I/O device out of a possible 128, bits 10-12 specify one of eight IOT instructions; the processor and the priority interrupt system are considered devices. As in the basic format, I, X, and Y are used to calculate E, which is used as an address in some cases, as control information in others.

A programmed operator is designated by three zeros in bits 0-2. Whenever such an operator appears in the program, the processor calculates an effective address from bits 13-35 of the instruction word in the usual manner but it does no further decoding; instead it stores the contents of the instruction register in the left half of location 40 and the calculated effective address in the right half, and then executes the instruction contained in location 41 (which is usually a JSR to an appropriate subroutine).

c Program Flags

The processor contains a number of flags that may be sensed by the program. Flags that are set automatically, e.g., by error conditions, usually cannot be set by the program; whereas, flags that allow the program to enable specific operations can always be both set and cleared. Some flags are governed primarily by jump instructions but most are contained in the processor I/O interface and are governed by IOT instructions. Any flag listed as being able to cause a priority interrupt does so on the channel assigned to the processor provided the priority interrupt system is active.

AR CRY0, AR CRY1	These flags are set by carries from the corresponding bits in AR. They are useful primarily for double-precision arithmetic and in correcting a result that has overflowed.
AROV AROV ENABLE	The overflow flag may be set by arithmetic operations in a variety of instructions. It indicates a loss of information, an incorrect result of a computation, or failure of the processor to perform a computation. Setting OV causes a priority interrupt if the enable flag has been set by the program.
PC CHG PC CHG ENABLE	The PC change flag is set when the program sequence is changed by a skip or jump instruction. Setting PC CHG causes a priority interrupt if the enable flag has been set by the program.
PDL OV	The flag is set and triggers a priority interrupt when a pushdown or pullout instruction has gone outside of the core area assigned to the pushdown list.
NON EXIST MEM	This flag is set and triggers a priority interrupt when the memory system fails to respond to a memory request.
CLOCK CLOCK ENABLE	The clock flag is set every sixtieth of a second by a signal from the main power control. It causes a priority interrupt if the enable flag has been set by the program.

USER

As a flag, the sole function of USER is to indicate to the executive routine whether a user program was interrupted, either by a priority interrupt or UUO (the executive routine must service all priority interrupts) or by the trapping of an illegal instruction (instructions that are illegal during a user program are a JRST that attempts to dismiss an interrupt or halt the computer, and any IOT). The execution of a JSR during a PI cycle or following a UUO or the trapping of an illegal instruction clears USER.

As a control flip-flop, USER implements the restrictions on user programs. Thus in order to restrict the operation of a user program, the executive routine must set USER when it transfers control to the program. If the sole purpose of an interrupt is to service a block IOT and there is no overflow, USER stays set and control automatically reverts to the user program after the IOT.

ILLEG OP

An attempt by a user program to address a location outside of its restricted area in core sets this flag causing a priority interrupt. At the time of the interrupt, PC may point either to the location of the instruction which tried to use the address or to the location following this instruction.

Some subroutine-calling jump instructions store what is referred to in the logic as "miscellaneous bits" in the left half of the location that receives the program count. In returning from the subroutine, the program may use a jump that restores the bits to their original states. Included in the miscellaneous bits are the two carry flags, overflow, PC change, user, and a control bit that is used in a special case for returning from a priority interrupt. The four byte manipulation instructions that load or deposit a character require two main sequences for their execution, and a priority interrupt can occur between them. The first part fetches and, if necessary increments the pointer; the second operates on the byte. If the program jumps to a subroutine for an interrupt that occurs between the two parts, bit 4 is set in the PC store location. Then in the subroutine, the program may determine whether a character operation was interrupted; and upon the return, the stored bit ensures that the interrupted instruction, which must be restarted, will not reincrement the pointer.

2.2 MAIN SEQUENCE

This section is devoted primarily to the manner in which the processor main sequence executes an instruction and sequences the program, but it also treats the control elements that allow entry into the main sequence from the console and the executive system which controls the sharing of processor time by user programs.

a Console Control

Operator control over the processor is exercised through two types of logical inputs associated with the keys and switches. Inputs from the switches are control levels that may provide data or addresses for use by the processor or gates to govern specific processor events. The keys are momentary contact switches that trigger specific events or initiate sequences although the level output of a key may also be used as a gate for events associated with the key action. The complete effect upon the computer of all keys and switches is described in detail in Chapter 3, Operation; we are concerned here only with the way in which the keys affect processor operation, in particular the main sequence.

The logic associated with the console keys consists primarily of a key cycle time chain and a control flip-flop RUN. Normal processor operation is initiated by triggering the main sequence and setting RUN—the 1 state of this flip-flop allows the completion of each main sequence to trigger the next so that the processor executes one instruction after another. Whenever RUN is cleared either from the console or by the program, operation ceases at the end of the current main sequence. The stop keys can, of course, affect the computer while it is running, but most keys that initiate events cannot; only the initiating keys trigger the key cycle. For those key functions that make use of the main sequence, the key cycle performs the necessary preliminary operations, such as transferring information in from the console data and address switches; but for those functions that do not use the main sequence, the key cycle controls the entire operation.

The operator may place the processor in normal operation by means of the START, READ IN, and INSTRUCTION CONTINUE keys. For these functions, the key cycle sets RUN and triggers the main sequence. The INSTRUCTION STOP key halts the processor at the end of the current main sequence by clearing RUN. The processor may also be stopped at the end of any memory access by means

of the MEMORY STOP key, which disables the return from the memory subroutine to the waiting sequence. In this case, the processor is still "running" and normal operation may be resumed through the MEMORY CONTINUE key which simulate a memory subroutine return. The operator may also deposit information in the memory location addressed by the ADDRESS switches or examine the contents of that location while the processor is running. For these two console functions, a single key cycle is merely inserted between two main sequences. For the remaining functions, the processor cannot be running, i.e., RUN must be 0. The EXECUTE key causes the processor to execute as an instruction the word contained in the DATA switches. For this instruction, the key cycle triggers the main sequence but does not set RUN, so the processor stops when the instruction is complete. There are also two keys that allow the operator to examine or deposit information into a sequence of consecutive memory locations without addressing them individually. Each such examine or deposit requires a key cycle and these functions cannot be performed unless the processor is stopped.

For maintenance purposes, the console has a REPEAT switch. When this switch is on, any key function can be repeated at a rate determined by a pair of speed controls. The logic enables this by having the key cycle retrigger itself through a delay whose interval is determined by the speed setting.

b Instruction Execution

Most instructions are executed by the five cycles that comprise the main sequence: instruction, address, fetch, execute, and store. Each main sequence begins when the instruction cycle requests memory access to retrieve an instruction from the location specified by the program counter. Upon receiving the instruction, the processor enters the address cycle and performs the effective address calculation as outlined in 2.1b. If an address is indirect, a new address word is retrieved from memory and the cycle begins again. After repeating the cycle as many times as is necessary to produce the effective address, the processor goes on to the fetch cycle to retrieve the necessary operands. If an accumulator is specified, it is retrieved first and sent to AR. If the instruction uses a double-length operand, a second word is fetched from the next consecutive accumulator (with location 0 being taken as following location 17) and sent to MQ. In some instructions, an extra word must be retrieved from the memory location addressed by either the right or left half of the addressed accumulator. This type of operand is also sent

to MQ. Finally the processor fetches the memory operand as specified by the effective address and leaves it in MB. This last fetch is skipped if E is to be used as control information, an operand, or a jump address.

After fetching the operands, the processor enters the execute cycle in which it performs whatever logical, arithmetic, or control functions are necessary to carry out the instruction. This cycle also increments the program counter by one so that it points to the next instruction in normal sequence. If a jump or skip is being performed, PC is changed following the count. Finally the processor enters the store cycle to deposit the result, which is usually contained in AR. For most instructions, the result may be deposited in an accumulator, in memory or in both as specified by the instruction; for a double-length result, AR and MQ may be stored in consecutive accumulators. The processor then returns to a new instruction cycle.

Although most instructions are performed by the sequence outlined above, there are many that are performed by variations of it. The more complicated instructions are performed by special sequences that are entered from the execute cycle and usually return to the store cycle. Sometimes a special sequence handles the storage itself and returns directly to the instruction cycle. Other instructions must first fetch and operate on a pointer that provides information necessary for the retrieval of the true operand; such instructions require in effect two main sequences. A block transfer repeats the fetch and execute cycles once for every word in the block. Whenever the execute cycle occurs more than once for a single instruction, the incrementing of the program counter is inhibited in all but the final occurrence. In this way, PC points to the next instruction only when the current one is bound to be completed before any interruption can occur.

The actual form of the sequence and the operations carried out in it are determined entirely by the instruction code as decoded from the instruction register. The codes are divided into eightclasses according to the configuration of bits 0-2. If these bits contain 111, the instruction is in the special IOT format and IOT control decodes bits 10-12 to determine which of eight instructions is specified. If bits 0-2 are 000, the instruction is taken to be a programmed operator—there is no further decoding and the processor enters a special sequence from which a subroutine must properly interpret the remainder of the code (and of the instruction word for that matter). In the other six classes the remaining six bits are decoded by the hardware,

primarily by the logic associated with IR. They may be decoded in a variety of ways depending upon the instruction class. Occasionally, single bits are used to represent specific operations, such as specifying the left or right half in a half-word transfer or whether fixed-point multiply is to interpret the operand as an integer or a fraction. In other cases, groups of bits are decoded; for example in the Boolean class four bits determine which of 16 Boolean functions is specified, the other two determine the mode of execution. In some cases, all six bits are decoded to a single control level for an individual instruction that has no modes.

There are some instruction codes that are not used and are executed as no-ops; the unused octal codes are those for which no mnemonic is listed in Table A4-1. Since most instruction codes are divided into sets of bits that are decoded in different ways, it is possible for some combinations of mode and instruction to have no effect on the state of the computer and these may be considered as no-ops. An obvious example is a full-word transfer that does not change the operand and is performed in the self mode.

The way in which instructions are executed is also influenced by the requirements of the priority interrupt system and the executive system. The interrupt channels are strobed at the beginning of every instruction and address cycle; and if a request is discovered, the processor honors it by entering a special PI cycle in which it executes the instruction in the location corresponding to the channel being serviced. For a PI cycle, the processor starts a new main sequence and executes it in the normal fashion except that the address supplied to MA for instruction retrieval comes from a channel address encoder in the PI system rather than from PC, and the strobe is disabled so that the PI cycle cannot itself be interrupted. The instruction executed in a PI cycle must either do an I/O data transfer or transfer control to a subroutine for further service. If the data transfer requires no further service, the processor automatically returns to the interrupted instruction; if further action is required, a second PI cycle is executed so that control can be transferred to a subroutine before honoring any other interrupt. If control is transferred to a subroutine, the interrupt is "held" so that the processor may again be interrupted but only on a channel of priority higher than the one being held; the subroutine is responsible for releasing the interrupt upon completion.

The executive system restricts processor operation in order to permit time sharing by several programs. When running restricted (user mode), each program must operate within the area of

core assigned to it; an attempt to use an address outside of the assigned area causes a flag to be set and immediately initiates an interrupt on the processor channel (the location to which PC points depends upon the time within the main sequence that the illegal memory request was made). The execution of a programmed operator (UUO) is unrestricted, but the locations used by UUOs (40 and 41) are inaccessible to user programs; UUOs executed by user programs always transfer control to the (unrestricted) routine responsible for overall system operation. Besides restricting addresses, the user mode traps (as if they were UUOs) attempt to halt the processor, dismiss an interrupt channel, or operate an I/O device. Instructions executed in PI cycles are unrestricted even if the interrupted program was running in user mode.

c Executive System

The executive system includes the 8-bit memory protection and relocation registers PR and RLR, nets that monitor user instructions, and the user flag. These logic elements allow the processor to be run in a restricted mode to permit time sharing of several user programs. A program that runs unrestricted (the executive routine) must be responsible for overall system operation. The executive routine is responsible for scheduling user programs (assigning core areas, entering user mode, and transferring control to the current program, interrupting when its time is up), for servicing all interrupts and UUOs, for servicing all I/O needs of user programs, and for taking action when it receives control because a user program attempted to use an illegal address or instruction or gave up control through a UUO.

Following power turnon, the processor is automatically in executive (unrestricted) mode, and when it is not running, the operator may place it in executive mode by pressing the I/O RESET key (this action also clears the I/O equipment). During a priority interrupt cycle, the processor runs unrestricted; but if a user program is interrupted, the user flag (which normally implements the user restrictions) remains set. Thus unless one instruction suffices to service an interrupt, the executive routine must within a PI cycle clear the user flag to return the processor to the executive mode and transfer control to one of its subroutines. Similarly, since all UUOs are under executive routine control, the instruction in location 41 must be a JSR, which stores and clears the user flag. The executive routine enters a user program by means of a jump which sets the user flag (JRST with a 1 in bit 12). The return to a user program after an interrupt or UUO may be made by means of a restoring JRST (a 1 in bit 11). This instruction restores all

other flags to their original states and can set the flag but can never clear it. This prevents a user program from leaving user mode as a result of an incorrect restoring JRST.

Each user program is assigned a block in core whose first location is an integral multiple of 2000 octal (since the executive routine must use locations 40 to 57 to service UUOs and interrupts, 2000 is the lowest first address available for a user block); the block size is also an integral multiple of 2000. A user program is restricted to addresses from zero to one less than its block size; if it attempts to use an address equal to or greater than its block size, the illegal operation flag is set and an interrupt occurs immediately on the processor channel. To assign a core area to a program, the executive routine uses a processor DATAO, which loads PR and RLR, respectively, from bits 0-7 and 18-25 of the data word. Each time the memory subroutine is called during a user program, the executive system tests for an illegal address by checking that the address does not exceed C(PR) x 2000 + 1777; the size of the block is equal to [C(PR) + 1] x 2000. At the same time the user address is relocated by adding the block starting address to it; i.e., the address sent out on the memory bus is equal to C(RLR) x 2000 + C(MA). Addresses 0 to 17 are never relocated, so all programs have access to fast memory (note that this means that no user program ever uses the first 16 core locations in its assigned block).

The user flag implements the restrictions on a user program by enabling the relocation and protection circuits and enabling the nets that monitor user instructions. A user program may not use a JRST with a 1 in bit 9 or 10 (an attempt to dismiss an interrupt channel or halt the processor) nor any IOT. These instructions are trapped by having their IR decoder outputs drive the UUO command line when the processor is in user mode. As mentioned above, UUOs are unrestricted, i.e., unrelocated location 41 is executed. Thus in user mode, an illegal instruction is executed as if it were a UUO and thereby returns control to the executive routine.

2.3 ARITHMETIC LOGIC

The arithmetic part of the processor includes the three full-size registers AR, MB, and MQ, the two 9-bit registers SC and FE, the time chains that execute the special sequence instructions and subroutines, and a subroutine interface through which connections are made from the special time chains to the gating for the three main registers. Included in the AR part of

the logic are four flags, AROV, AR CRY0, AR CRY1, and PC CHG. The states of these flags are stored as miscellaneous bits and may be restored by a JRST; they may also be sensed for a jump and cleared by a JFCL.

Transfers of full words or half words may be made between MB and AR, transfers of full words between MB and MQ. MB may also receive PC, IR, or the miscellaneous bits for storage in a UUO and in certain jumps. The two halves of a word can also be interchanged (swapped) in MB. Although the AND function of MB and AR can be formed in MB, it usually plays a passive roll in logical and arithmetic operations by holding an operand which is combined with an operand in AR. Associated with AR is a myriad of gates that implement the clearing or setting of individual bits in a word according to a mask, the formation of the complement, OR, AND, and exclusive OR logic functions, and the shifting of bits left or right. There is also a carry function which can be triggered at any point in the register and produces an arithmetic carry to the left; i.e., it complements the first bit, complements the second if the first changes from 1 to 0, and ripples to the left in this manner until it complements a 0 bit. If this carry chain is triggered only at the register LSB, it adds 1 to the number represented by the contents of the register. Some instructions use the left and right halves of a word to hold a word count and an address; in order to allow indexing of both half words simultaneously, the carry chain can be triggered at AR17 and AR35. Although this is used as two simultaneous index functions, there is no break in the carry chain and an overflow from the right half can carry into the left: hence the pair of index functions effectively adds 1000001 to AR. The above listed functions are the only ones that can be performed directly—all others are executed by combinations of them. If following an exclusive OR, the carry function is triggered at a number of places in the register (the particular places being determined according to the addition algorithm by the previous configuration of the words in MB and AR), it generates the algebraic sum in AR of the numbers originally in MB and AR. Negation (which always means arithmetic negation) is performed by complementing and adding 1. In subtraction, the number in MB must be subtracted from that in AR: for fixed point, the processor performs subtraction by complementing AR, then adding and complementing the result; for floating point, MB and AR are switched and the subtrahend in AR is then negated so the result can be produced merely by adding. Multiplication or division is a sequence of shifts with additions or subtractions interspersed.

The third register, MQ, is used occasionally for temporary storage and there is a special case in character operations where AR and MQ are shifted in parallel for control purposes, but MQ serves primarily as a right extension of AR for handling double-length operands. For actions on a pair of accumulators, the two registers are joined end to end and the double-length operand may be shifted in either direction. Moreover, the opposite ends of both registers may be joined to form a ring and the contents rotated in either direction. In multiplication, the multiplicand comes from AC and the multiplier is either C(E) or E, but when performing the actual arithmetic operation, MB holds the multiplicand and the multiplier in MQ controls the formation of partial products in AR. As bits of the multiplier are used and shifted out of MQ, the low-order bits of the double-length product are shifted in. In division, MQ holds the low-order half of the double-length dividend and as bits are shifted out to AR for use by the division steps, bits of the quotient are shifted in at the least significant end. At the completion of the computation, MQ contains the quotient and AR the remainder, but the divide subroutine then switches their positions so the quotient can be stored in AC.

In floating-point operations, the exponent is first calculated in SC, whose gating provides addition and indexing. In floating scale, the only operations performed are on the exponents. For other floating-point instructions, the exponent is calculated in SC and then stored in FE while SC is used to count the steps in the fixed-point part. Following computations, the exponent is transferred back to SC in case it must be changed while normalizing the result, and finally from SC it is inserted in the exponent part of AR. SC is also used to calculate the position portion of a pointer for a character operation that increments, and from SC the new position is inserted in the pointer in AR.

In addition to the registers, the arithmetic logic also includes the time chains and many control nets for executing the special sequences and subroutines. The basic subroutines, which can be called from any higher level, are the AR subroutine group (which includes fixed-point addition, subtraction, negation, and indexing in either direction), the SC addition subroutine, and the SC shift-count subroutine which simultaneously counts SC and shifts AR and/or MQ (for shift instructions both registers are shifted even though for a single operand only AR contains information). For fixed add and subtract, the execute cycle calls the AR subroutines directly. For other instructions, the processor switches from the execute cycle to a special sequence which calls the lower rank subroutines and which usually returns to the store cycle.

The sequences for character operations, block transfer, shift operations, and floating scale call only the basic subroutines (including the memory subroutine). Floating multiply and divide begin by calling the exponent calculate subroutine, then the multiply or divide subroutine whichever is appropriate, and both terminate by entering the normalize return subroutine which also follows the floating add-subtract sequence. The fixed multiply sequence calls only the multiply subroutine; fixed divide does not make use of an intermediate special sequence but instead enters directly into the divide subroutine.

2.4 MEMORY INTERFACE

The interface that connects the processor to the memory bus includes the memory address register MA, memory buffer MB, user mode registers PR and RLR, memory indicator register MI, and the control logic for the memory subroutine. A processor cycle or special sequence gains access to memory by triggering the memory subroutine, which has entries for read, write, and read-pause-write which must later be followed by a read-write restart. The calling sequence must also supply an address to MA, and if information is to be written, a word to MB. If the processor is in executive mode, the subroutine places the appropriate request levels on the bus immediately, but for user mode there is a delay while the address in MA is compared with PR. An illegal address causes the processor to go to the end of the current main sequence and sets the illegal operation flag requesting an interrupt on the processor channel.

While the comparison against PR is being made, the outputs of RLR and the more significant MA bits are applied to a set of dc adders whose outputs represent the sum of the two registers. If the address in MA is legal, memory control puts the relocated address on the bus (low-order bits are supplied directly from MA, high-order bits from the relocation adders). In the address as received by memory, MA34 supplies the least-significant bit of the address within a single memory and bit 35 is used as the LSB to select the bank. In this way, consecutive addresses are interleaved—all odd addresses in one bank, all even in another. A switch at the memory allows the operator to disable this feature when using a 16K bank (with 8K banks, addresses must be interleaved).

The processor memory subroutine requests a memory cycle by calling memory as a subroutine, and it must wait until the addressed memory accepts the request, which does not occur until

the memory is free and this processor has priority. The processor restarts upon receipt of an acknowledgement signal from memory. If the request is for a write cycle, the processor need wait only until the memory accepts the word in its own buffer; but for a read cycle, it must wait until it receives the information read from the memory location. If the request is made to fast memory, the write takes slightly longer than the read because there is no buffer. If the request is not acknowledged within a considerable time compared to a memory cycle, the nonexistent memory flag is set, requesting an interrupt on the processor channel. Following the acknowledgement signal, the memory subroutine sends a restart pulse to the waiting sequence unless the MEMORY STOP key is on. To restart the processor after a memory stop, the operator must simulate the return to the waiting sequence by pressing the MEMORY CONTINUE key.

If the address in MA is the same as that in the console ADDRESS switches or the operator is examining or depositing information from the console, the contents of the memory buffer are displayed in the memory indicators. On a read MI displays the information read, on a write it displays the information to be written.

2.5 INPUT-OUTPUT SYSTEM

At the processor end of the I/O bus is the in-out transfer control logic that times the transfer of data, initial conditions, and status over the bus by sending command signals (also over the bus) to the device control units. Two of the devices on the bus are the priority interrupt system located in the processor, and the processor itself whose I/O interface contains a number of flags through which internal processor conditions can request priority interrupts and which allow the processor to check its own internal status with IOT instructions.

When the code 111 appears in bits 0-2 of the instruction register, the processor IOT control decodes bits 10-12 to determine the specific IOT instruction. Upon reaching the execute cycle, the processor switches to a special IOT sequence that times the instruction operations and generates the necessary command signals. Only four types of command signals are sent out on the bus; these are for DATAI, DATAO, CONO, and STATUS, of which the first three correspond to individual IOT instructions. BLKI or BLKO requires signals on the bus only

after conversion to a DATAI or DATAO. CONI, CONSZ, and CONSO bring conditions in and the latter two then perform tests; all three generate the STATUS command and affect the peripheral equipment in exactly the same way.

While IOT control is generating the command signals, the device code from IR bits 3-9 is supplied over the bus to enable a gate in the device with that assigned code; signals are sent to all devices but only the selected device can respond. Data or initial conditions are supplied from AR over the bus to the selected device; data or status is supplied from the device over the bus to AR. Among the initial conditions that CONO may supply to a device is a priority interrupt assignment; CONO assigns a channel from 1 to 7 (zero is no assignment); and whenever the device requires service, it requests an interrupt by sending a request signal to the PI system on the bus line corresponding to its assigned channel. Every device except the PI system itself can receive at least one PI assignment.

The PI lines go to the priority interrupt system which contains three 7-bit registers, PIO, PIR, and PIH, to control the seven channels. A given channel is governed by one flip-flop from each register. The PIO flip-flop turns the channel on or off. The PIR flip-flop synchronizes the request to the processor main sequence and in conjunction with the remaining PIR flip-flops and a priority chain, generates an internal request signal for the channel that has priority. The PIH flip-flop holds a break on the selected channel. There are also three control flip-flops for the interrupt system, one that activates it, another that places the processor in a PI cycle, and a third that detects overflow from a block IOT performed in a PI cycle. By checking status, the program can determine whether the system is active and which channels are on. The processor strobes the PI lines at the beginning of every instruction and address cycle, and synchronizes a request signal from any PI line provided that the corresponding channel is on. If a PIR is on (a CONO can set the PIR for a channel even if the channel is off), the processor enters a PI cycle and starts a new main sequence which honors the request by performing the instruction in a particular memory location associated with the channel (if several PIR flip-flops are set, the lowest numbered channel has priority). To retrieve the instruction, MA receives the address from an encoder in the PI system. The encoder outputs are connected to MA in such a way that the channel number is doubled and added to 40. Thus for channel n, the processor executes the instruction in location 40 + 2n (the PI system uses locations 42 to 57, two for each channel). This instruction should be either a JSR to an appropriate subroutine or a block IOT to handle

a data transfer. If it is a BLKI or BLKO and there is no overflow, the processor returns immediately to the interrupted program (another priority interrupt can occur before any instruction in the interrupted program is actually executed). If there is overflow, the processor goes into a second PI cycle in which it performs the instruction in location 41 + 2n, which should be a JSR to an appropriate subroutine. An instruction in a PI cycle should be either a BLKI, BLKO, or JSR; other instructions can be executed but they usually would have unfortunate consequences for the program and could even hang up the processor. If there is a jump to a subroutine, the break is held by setting the PIH flip-flop for the channel. This disables part of the priority chain so that the break routine can be interrupted only by a channel of higher priority. At the completion of the subroutine, the program should dismiss the channel so as to reenable all lower priority channels as well as the channel on which the break occurred.

Since a BLT may require considerable execution time, the PI request lines are also strobed following each word processed in the block. Whenever a request is discovered, the current source and destination addresses are stored in the accumulator and the partial block is terminated. The processor then begins a new main sequence as if to restart the block transfer, but is interrupted instead.

The I/O interface for the processor contains the flags discussed in 2.1c except for the four flags associated with AR and the user flag in the executive system. However two of the AR flags, overflow and PC change, can be set by the CONO that controls the interface flags and supplies it with a PI assignment, and these AR flags plus the user flag can be sensed as processor status (although the user flag is meaningless as status since it is 0 by definition whenever an IOT can be executed). The flag set by an illegal user address, a pushdown list overflow, or a request made to a nonexistent memory automatically requests an interrupt on the processor channel. Setting the clock, overflow, or PC change flag can cause an interrupt only if it has been enabled by a CONO. The program may also use data instructions for the processor: DATAO loads PR and RLR for a user program; DATAI brings in information from the console data switches.

Control units for other devices each contain a data buffer for transfers between the I/O bus and the device, an interface for control connections to the device, and an interface for control connections to the bus. The size of the buffer depends upon the device. It is 36 bits if full

words can be transferred, but smaller if the transfers must be single characters. For an output device, the buffer can be loaded by DATAO; for an input device, DATAI gates the buffer onto the bus. The interface between the control unit and the bus includes a control register and a status register, which usually overlap and may be identical. For initial conditions, CONO can provide at least one PI assignment, place the device in operation, clear whatever error flags there may be, and often provide additional information such as determining the mode of operation or selecting an individual device from several that are connected to the same control unit. Usually all control bits can be examined as status and often additional status signals are supplied by the device. In most cases, the data instructions also perform certain control functions. For input, the loading of the buffer with information from the device usually sets a flag causing a priority interrupt. The processor responds with a DATAI that not only gates the buffer onto the bus but also clears the flag and initiates the retrieval of more information. For output, the transfer of information from the buffer to the device sets a flag causing an interrupt, and the processor responds with a DATAO that not only supplies new information but clears the flag and initiates the next transfer from buffer to device.

Included in this manual are the control units for four I/O devices. The paper tape reader has a 36-bit buffer but information may be retrieved in two modes. In alphanumeric mode, only one 8-bit line is read from the tape; in binary mode, the control unit accepts data from holes 1-6 only in lines in which hole 8 is punched, but it assembles six such characters into a 36-bit word. The punch handles only one character at a time but it still has two modes. In alphanumeric, it punches an 8-bit character; in binary, it punches a 6-bit character in holes 1-6, never punches hole 7, and always punches hole 8. The keyboard-printer is actually two independent devices with one PI assignment. For output, the processor prints single characters; for input, each character typed by the operator is placed separately on the bus. Characters typed at the keyboard are not printed unless the program sends them back out. With the card reader, only a CONO can initiate operations but only one CONO is required per card because once a card is started all 80 columns are read. The program can specify whether an interrupt shall be requested following each column or only when the buffer is full. In binary mode, all twelve holes of each column are read and three columns are assembled into a word. In alphanumeric mode, the Hollerith character in a column is translated into a 6-bit character and the control unit assembles six into a word. If an interrupt is requested but is not serviced before a new column is read, an error flag is set.

CHAPTER 3

OPERATION

This chapter discusses the normal operation of the arithmetic processor, reader, punch, Teletype, and card reader; some maintenance information is included, but the detailed discussion of operation for maintenance purposes is in Chapter 9. Although this chapter is relatively self-contained, it is recommended that the reader first familiarize himself with the functional organization of the equipment as presented in Chapter 2.

3.1 CONTROL PANELS

This section describes the function of the controls and indicators that are readily accessible to the operator; those mounted behind the doors of the bays are described in 9.1. All controls for normal operation of the processor, reader, and punch are on the main operator panel at the center of the console; this panel also contains most of the processor indicators. The panels at the top of bays 1 and 2 contain only indicators, most of which are for maintenance purposes. Indicators for the four in-out devices are on the upper part of the panel located behind the metal cover at the top of the right console bay (the lower part of this panel contains the marginal check controls, which are described in 9.1).

The name used in the logic drawings for a register or control level is listed in parentheses whenever it differs from the name engraved on the panel. When any indicator is lit, the associated flip-flop is in the 1 state or the associated function is asserted. Indicators for logic elements that retain their states over a considerable number of main sequences display useful information while the processor is running, but most indicators change too frequently and are therefore discussed in terms of the information they display when the processor has stopped. For maintenance purposes, the processor may be stopped from the console after every memory subroutine. Switches located inside the bay doors allow stopping after AR subroutines and single stepping through a shift-count. However, the discussion here is limited to stops at the end of a main sequence, i.e., at the completion of an instruction. This includes all programmed halts as well as the situation in which the operator latches down the INSTRUCTION STOP key to run a program at slow speed stopping after every instruction.

a Console Operator Panel

This panel contains indicators for most of the registers and control flip-flops that are of concern to the operator and contains all of the operating keys and switches. The switches supply continuous levels and all but the rotary speed controls and the console lock are 2-position toggles for which up is 1 or on. The keys are momentary contact levers that initiate or terminate operations, or produce an action only while held on.

Indicator Registers

INSTRUCTION (IR0-8) - Bits 0-8 of the instruction just completed. If the left three lights are all off, the instruction is a UUO and the remaining bits are defined by the program; if the left three are on, the instruction is an IOT and the remaining lights display the first six bits of the device code. Any other configuration of the first three bits indicates the basic format, for which the register contains the instruction code.

AC (IR9-12) - For instructions using the basic format, these four bits are usually an accumulator address, but for some instructions they are used for special purposes such as addressing flags. In an IOT instruction the left bit is the LSB of the device code; the remaining three bits specify one of the eight IOT instructions.

I (IR13) - This is the indirect bit, and it should always be off when the processor has stopped at the end of an instruction.

INDEX (IR14-17) - Contains the address of the last index register used in the instruction just completed. If the four lights are all off, there was no address modification in the final address cycle.

MEMORY (MI) - This 36-bit register displays the contents of the memory location associated with any console examine or deposit operation. The lights may also be used to display any desired location while the processor is running.

PROGRAM COUNTER (PC) - This 18-bit register contains the address of the next instruction in the program.

MEMORY ADDRESS (MA) - On a programmed halt this 18-bit register indicates an address one greater than that of the location containing the halt instruction. On an instruction stop in slow speed operation, the register usually contains the address used for the last memory access. However, if there was no storage, either in the store cycle or in a subroutine, it contains the effective address, which may or may not be the address of the last memory access.

In addition to the above there are three processor registers located at the top of the in-out panel (c below).

Switch Registers

DATA (DS) - This register allows the operator to supply a 36-bit word to the processor. The operator may either deposit the word in memory or cause the processor to execute it as an instruction. The program may also read DATA with a DATAI for the processor.

ADDRESS (MAS) - By means of this 18-bit register the operator may specify address for use with the operating keys and switches. Whenever the memory subroutine gains access to the location specified by ADDRESS, the contents of that location are displayed by MEMORY. For a read request MEMORY displays the word read; for a write request the word written is displayed.

Control Indicators

RUN - Lit while the processor is running in normal operating mode, with each main sequence triggering the next. When the light goes off, the processor stops upon completion of the current instruction.

MEM STOP (MC STOP) - If this light goes on at the beginning of a memory subroutine, the processor stops after memory access is completed because the subroutine fails to send a restart

pulse to the calling sequence. If RUN is also on, the processor can be restarted only by lifting the MEMORY CONTINUE key. If RUN is off, other keys may be used but only MEMORY CONTINUE restarts the interrupt key function.

PI ON (PI ACTIVE) - Indicates that interrupt requests can be granted by the priority interrupt system.

PI ACTIVE (PIO1-7) - These lights indicate which PI channels are on. The numerals below the lights specify the channels for these and the following two sets of indicators.

PI REQUEST (PIR1-7) - These lights indicate the channels on which requests have been synchronized. The program can force a request even if a channel is not on; for a request from any other source the REQUEST light can go on only if the corresponding ACTIVE light is on.

PI IN PROGRESS (PIH1-7) - These lights indicate the channels on which breaks are currently being held. Several lights may be on simultaneously, but while a given light is on, no higher-numbered light may go on; a lower-numbered channel can interrupt following the PI cycle(s), and the channel that is actually being serviced is the lowest-numbered one whose light is on. When a PROGRESS light goes on (following a jump to a routine for the break), the corresponding REQUEST goes off and cannot go on again until PROGRESS goes off.

If a break is serviced by a block IOT without overflow, the PIH flip-flop is set and cleared within a single PI cycle so REQUEST goes off without PROGRESS going on. If there is overflow, two PI cycles are required; at the end of the first, PROGRESS will not yet be on and REQUEST will still be on even though the break is being serviced. However, in this case the PI OV, PI CYC, and PI REQ lights at the top of bay 1 will all be on. The lights act in this way because PROGRESS can go on and remain on in a PI cycle only if the instruction performed is not an IOT. Thus a faulty program can hang up the processor in a PI cycle, and the only visual indication that the break is being held is that PI CYC and PI REQ will be on. For example if the channel location contains a CONO, the processor will repeat the instruction indefinitely with REQUEST on and PROGRESS off; PC will be static and will point to the next instruction in the program.

Operating Keys

In the right half of the operator panel is a row of eight 3-position switch levers, each of which is two logical keys. The momentary contact, up and down positions of a given lever are the on positions for the keys whose names are written above and below; the stable center position is off for both keys. The two levers at the right end of the row control the reader and punch, and these may be used at any time whether the processor is running or not. The other twelve keys affect the processor, and of these, two are stop keys, the others are initiating keys (i.e., they trigger the key cycle). Although special considerations for individual keys are given below, it is assumed throughout the discussion that the executive system is not in use, i.e., that both operator and program have access to all of memory and no operations are illegal. In order to use the keys properly when the executive system is in use, the operator must be fully aware of the special conditions imposed (a complete discussion of the relation of the keys to the programmed operation of the system is presented in 3.3b).

START - This key functions only if RUN is off. It places the processor in normal operation (lighting RUN) and causes the first instruction cycle to retrieve an instruction from the location specified by the ADDRESS switches.

READ IN - This key is exactly the same as START except that it also causes the processor to enter the readin mode, lighting the RIM SBR indicator at the top of bay 1. In this mode the fast memory is disabled, and any memory call with an address 17 or less is given access to the readin area, the normally inaccessible bottom 16 core locations. Whenever an instruction is retrieved from any location above 17, the processor leaves the readin mode.

INSTRUCTION STOP (INST STOP) - Turns off RUN, causing the processor to stop at the completion of the current instruction. This key has a catch that allows it to be left in the on position for single step operation. The turnon of the key triggers events that facilitate emergency stops (for details see 3.3c).

INSTRUCTION CONTINUE (INST CONT) - This key functions only if RUN is off. It causes the processor to resume normal operation (lighting RUN) beginning with the instruction in the

location specified by PROGRAM COUNTER. By leaving INSTRUCTION STOP on, the operator can single step instructions by pressing INSTRUCTION CONTINUE. The latter key also has a catch so that by leaving both keys on and using the REPEAT switch, a program can be run at slow speed.

MEMORY STOP (MEM STOP) - This key has a catch that allows it to be left in the on position for single step operation. While the key is on, the MEM STOP light goes on at the beginning of every memory subroutine, causing the processor to stop at the completion of each memory access. During single step operation a call for read-pause-write in the fetch cycle generates only a read request so the processor does not hold memory during the stop. The subsequent restart then triggers a separate write cycle. This key is used only for maintenance purposes and the meaning of the lights depends upon where the stop occurs within the main sequence. However, MEMORY ADDRESS always displays the location to which access was made.

MEMORY CONTINUE (MEM CONT) - This key functions only if the MEM STOP light is on, and it then restarts whatever sequence was interrupted by the MEMORY STOP key (it also turns off the light). By leaving MEMORY STOP on, the operator can use MEMORY CONTINUE to single step by memory calls. The latter key also has a catch so that by leaving both keys on and using the REPEAT switch, a program can be run at slow speed from one memory call to the next.

EXECUTE (EXEC) - This key functions only if RUN is off. It causes the processor to execute the instruction contained in the DATA switches and stop immediately upon completing it. While the key is on, the normal program counting in the execute cycle is inhibited; thus PC cannot be affected unless a skip or jump is executed. A programmed skip always increments PC once for the normal program count and may increment it a second time for the skip; from the console a skip increments PC at most once. A programmed jump always increments PC before saving it so that it points to the next instruction in the program; when executed from the console, a jump loads PC normally but saves the count that is already in it.

IO RESET - This key functions only if RUN is off. It clears all flags, control flip-flops and control registers in the processor (placing it in executive mode) and in most equipment connected to the in-out bus.

DEPOSIT THIS (DEP) - Deposits the contents of DATA in the location specified by ADDRESS. The word deposited is displayed by MEMORY. If RUN is off during the deposit, the processor stops with the MA lights displaying the address of the affected location. This key should be used while the processor is running only if there is no chance of a program halt occurring (3.3b).

DEPOSIT NEXT (DEP NXT) - This key functions only if RUN is off. It deposits the contents of DATA in the location whose address is one greater than that specified by MEMORY ADDRESS, and the word deposited is displayed by MEMORY. At the completion of the operation MA contains the address of the affected location.

EXAMINE THIS (EX) - Causes MEMORY to display the contents of the location specified by ADDRESS. If RUN is off during the operation, the processor stops with the MA lights displaying the address of the examined location. This key should be used while the processor is running only if there is no chance of a program halt occurring (3.3b).

EXAMINE NEXT (EX NXT) - This key functions only if RUN is off. It causes MEMORY to display the contents of the location whose address is one greater than that specified by MEMORY ADDRESS. At the completion of the operation MA contains the address of the examined location.

READER ON - Turns on the reader motor, energizes the brake, and triggers a PI request on the reader channel.

READER OFF - Turns off the reader motor, releases the brake, and triggers a PI request on the reader channel.

READER FEED - Feeds tape through the reader while held on (provided the reader is on).

PUNCH FEED - While this key is held on, the punch generates blank tape, i.e., tape with only feedholes punched.

Operating Switches

The first four switches are toggles located at the right end of the operator panel, and associated with each is an indicator that lights while the switch is on.

POWER - This switch applies power to the processor and the control units for reader, punch, and Teletype, and makes power available to all external units (memories, peripheral equipment) whose local power controls are in remote. Almost every unit has its own power switch, which if left on, allows the unit to come on with system power. Exceptions include the reader, which must be turned on and off at the processor console, and the punch, which is turned on by the logic and goes off automatically whenever it is not called for 5 sec. After turning POWER on, wait a few seconds to allow the power clear to terminate and memory power to come on.

ADDRESS STOP (ADDR STOP) - While this switch is on, a memory stop occurs whenever access is made to the location specified by the ADDRESS switches. At the stop the MEM STOP light is on, and MEMORY displays the word read or written. Throughout the time that the switch remains on, any fetch cycle call for read-pause-write generates only a read request, so the processor does not hold memory following the stop. The subsequent restart then triggers a separate write cycle.

DISABLE MEMORY (MEM DISABLE) - While this switch is off, the failure of a memory to respond within 100 μsec to a request for access turns on the NONEX MEM light on the bay 1 indicator panel, causing a PI request on the processor channel. If the switch is on, such failure causes the processor to hang up in the memory subroutine. The operator can free the processor by pressing INSTRUCTION STOP and then I/O RESET.

REPEAT - Causes the sequence initiated by an operating key to be repeated as long as the key is held on. The sequence is iterated at a rate determined by the SPEED switches.

SPEED - These switches allow the operator to vary the repeat interval from 3.4 μsec to 8 sec in six overlapping ranges. They include a 5-position rotary range switch and a potentiometer knob for fine control within each range.

Console Disable - In the lower right corner of the panel is a key-locked switch. Turning the key clockwise disables all operating keys and switches on the panel (except those for the reader and punch) so no one can interfere with the operation of the processor.

b Bay Indicator Panels

Figure 3-2 shows the indicator panels at the tops of bays 1 and 2. Bay 2 displays the three main full-word registers: memory buffer MB, arithmetic register AR, and multiplier-quotient register MQ. Since the results of an instruction are stored in memory, these registers are useful primarily for single-step maintenance operation, and their contents at an instruction stop depend entirely upon the instruction just performed. If the instruction requires storage, MB always contains the last word stored. AR contains the word stored in an accumulator (if any), and for a double-length result MQ contains the word stored in a second accumulator.

On bay 1 are the indicators for flip-flops and control levels. Indicators for the flags described in 2.1c are at the right end of the panel. At the top of the third column from the right is the EXEC MODE light, which is driven from the 0 output of the user flag and is thus lit when the executive routine is running (or the executive system is not in use). The remaining flags are as listed in the text, although the names engraved on the panel are in many cases abbreviated from those used in the logic drawings. PDL OV is at the bottom of the sixth column from the right. The second light in the third column, CPA ILL OP, is the illegal operation flag, which indicates that a user program has attempted to address a location outside of its assigned core area and should not be confused with EX ILL OP just below. The latter flip-flop inhibits relocation when a UUO or an illegal user instruction is trapped, and the light is always off at an instruction stop. In the second column are the AR overflow and carry flags and the PC change flag. The carry flags should not be confused with the carry flip-flops above them: the flip-flops detect carries in AR and their states are transferred to the flags only in those instructions wherein the information is relevant to the program. The remaining flags are in the right column.

Besides the flags, the indicator panel also includes shift counter SC, floating-exponent register FE, several important control levels, and a multitude of flip-flops that govern the sequencing of the various processor cycles, special sequences, and subroutines. The following

indicators are of importance to the operator in normal operation (unless otherwise specified, the meaning of a light is given for an instruction stop):

KEY EX SYNC, KEY EX ST - If the operator presses EXAMINE THIS while the processor is running and it stops with both of these lights on, the desired key function has not been performed. If only the sync light is on, the operation may have been performed incorrectly (see 3.3b). If the key is pressed while RUN is off, the start light does not go on at all, but the sync light goes on and remains on until some other initiating key is pressed.

KEY DEP SYNC, KEY DEP ST - If the operator presses DEPOSIT THIS while the processor is running and it stops with both of these lights on, the desired key function has not been performed. If only the sync light is on, the operation may have been performed incorrectly (see 3.3b). If the key is pressed while RUN is off, the start light does not go on at all, but the sync light goes on and remains on until some other initiating key is pressed.

CHF7 - If this light (bottom, fifth column from left) is on following a PI cycle that executes a BLKI or BLKO, an interrupt has occurred between the two parts of a character operation. Following a JSR in a PI cycle, the light will be off even if a character operation was interrupted. If CHF7 is on following a JRST, the instruction is returning from a break and the processor is about to restart an interrupted character operation.

SPLIT SYNC - Indicates that if there was a read-pause-write call during the preceding fetch cycle, it triggered only a read request, and the subsequent restart triggered a separate write cycle.

STOP SYNC - Indicates that the preceding fetch cycle triggered a read-pause-write memory cycle.

PI OV - Indicates that a BLKI or BLKO performed in a PI cycle has overflowed.

PI CYC - This light goes on when a PI request is honored, but is still on at the completion of an instruction only if a second PI cycle is required for the interrupt (i.e., the instruction was a BLKI or BLKO that overflowed).

PI REQ - At the completion of an instruction this light is on whenever PI OV and PI CYC are on. If those two lights are not on, PI REQ indicates that the PI system is active and a request that has been synchronized has not yet been honored. This can occur if a previous instruction activated the system and some requests were already waiting. If the system was already active, either a request was made by the program or synchronized by a BLT, or several requests were synchronized simultaneously and the processor has just finished servicing one of those with higher priority.

A LONG - Indicates that the address cycle of the preceding instruction used an index register for address modification or used an indirect address.

MA = MAS - Indicates that the number displayed by the MEMORY ADDRESS lights is identical to that contained in the ADDRESS switches. This light is on whenever an address stop occurs but may be on at other times as well.

EX PI SYNC - Indicates that the main sequence just completed was a PI cycle. This light remains on even when PI CYC goes off before the instruction is completed.

RIM SBR - Goes on when the operator presses the READ IN key and remains on until an instruction is retrieved from a location above 17. The light is always on if the system includes no fast memory, or if it is not in use.

PIA 33, 34, 35 - Indicate the PI channel assigned to the processor. If all three lights are off, no channel has been assigned.

At the completion of an instruction, SC and FE may have any configuration. Besides the flags and the control indicators discussed above, the following lights may be on at an instruction stop: MC WR, NRF2, NRF3, DSF7, MPF2, SC = 777, UUOF1, EX UUO SYNC (always on),

MQ36, CRY0 ⊻ CRY1, AR CRY0, AR CRY1. None of the remaining lights should be on at an instruction stop. At a memory stop MC RD will be on if the memory cycle was used to retrieve information, and at least one other light will be on to indicate the point at which a time chain is stopped awaiting the return from the memory subroutine. For example if KEY RD/WR in the left column is on, the memory stop occurred in an examine or deposit operation initiated at the console. Refer to 9.1 for further information on the use of these indicators for maintenance purposes.

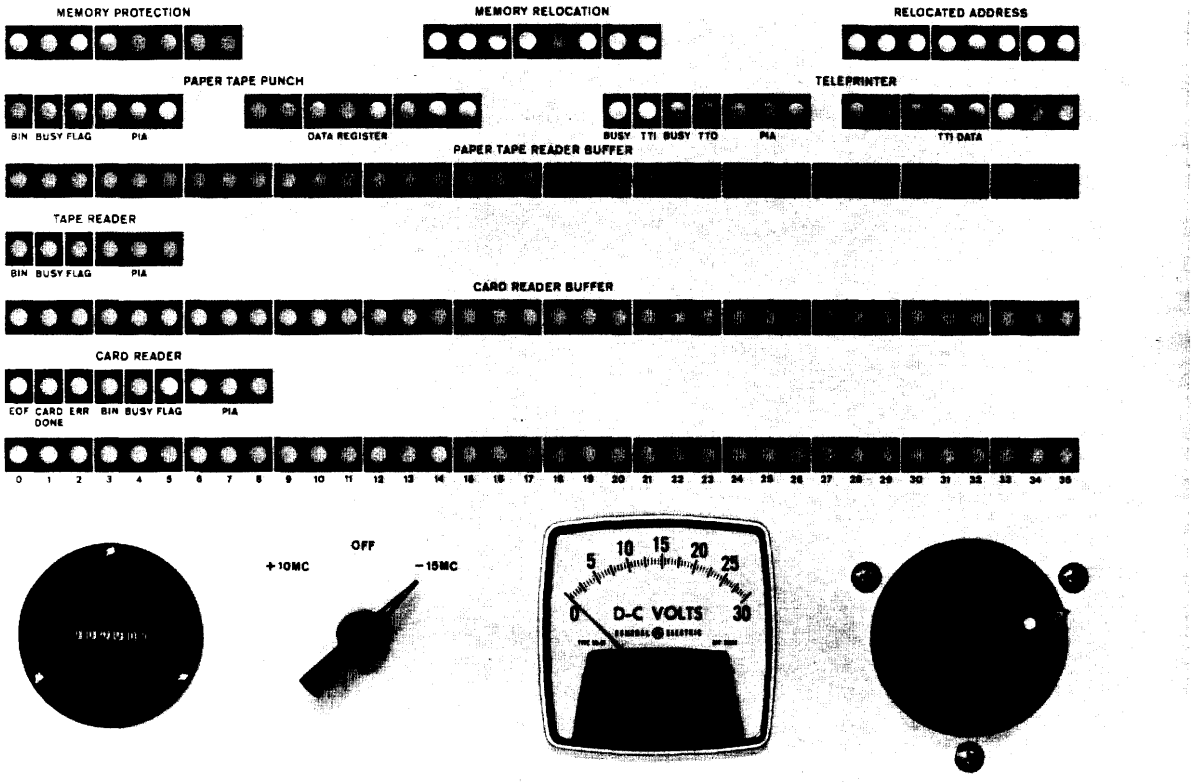


Figure 3-3 In-Out and Marginal Check Panel

c In-Out Indicator Panel

Figure 3-3 shows the panel that contains the in-out indicators and the marginal check controls (the latter are described in 9.1). At the top of the panel are three 8-bit indicator registers that are associated with the executive system rather than the in-out equipment. These are as follows:

MEMORY PROTECTION (PR) - Defines the size of the block in core available to a user program. The number of locations in the block is 2^{10} times the number one greater than that contained in the register. Each program must use addresses from zero to one less than the block size. If a user attempts access to an address greater than the number in PR followed by ten ones, i.e., to an octal address greater than C(PR) x 2000 + 1777, the processor skips the remainder of the current instruction, and the CPA ILL OP light at the top of bay 1 goes on causing a priority interrupt on the processor channel.

MEMORY RELOCATION (RLR) - Specifies the position of a user block in core. The address of the first location in the block is the number contained in RLR followed by ten zeros. Each user address other than for fast memory is relocated to the assigned block by adding C(RLR) x 2000 to the number displayed by MEMORY ADDRESS.

RELOCATED ADDRESS (RLA) - These lights display the most significant eight bits of a user address as it is placed on the memory bus (the least significant ten bits come directly from MA).

The bottom row of indicators on the panel displays the contents of the 36 data lines in the I/O bus. Since the bus is reset following every data transfer, the lights should always be off when the processor is stopped. The remaining lights are the buffers and some of the control and status bits for the reader, punch, keyboard-printer, and card reader. Each device has a 3-bit PIA register that contains the number of the PI channel assigned to it. Whenever the FLAG light for any device goes on, a PI request is made on the channel specified by the associated PIA (if all three PIA lights are off, there is no channel assignment). The lights labeled TELEPRINTER are actually for two distinct devices, keyboard and printer. Both share a common PI assignment but have duplicate control bits. The data buffer shown is actually for the keyboard; the printer buffer is not shown because it automatically clears as each character is being transmitted.

PAPER TAPE PUNCH -

DATA REGISTER (PTP1-8) - Contains the last character punched. The buffer bits are numbered 1 to 8 from right to left and correspond to a frame of

tape viewed with the feed hole near the right edge. The buffer receives information from bus lines 28-35, with line 35 supplying the information for bit 1.

BIN (B) - While this light is on, any punch operation always punches hole 8, never punches hole 7, and punches hole 6-1 according to the information on bus lines 30-35. While the light is off, the information on lines 28-35 is punched.

BUSY - Indicates that the punch is in operation.

FLAG - Causes a PI request upon completion of a punch operation. FLAG goes off when the program supplies another character.

TELEPRINTER -

TTI DATA (TTI1-8) - Contains the last character received from the keyboard. The buffer bits are numbered 1 to 8 from right to left so that when a character is shifted in at the left, the first bit received ends up in buffer bit 1. The character is transferred to the processor over bus lines 28-35 with bit 1 on line 35.

BUSY, TTI (BUSY, FLAG) - This pair of lights (at the left) is for the keyboard. BUSY goes on when a key is struck. When the entire character is assembled in TTI DATA, BUSY goes off and TTI goes on, requesting an interrupt. TTI goes off when the program retrieves the character.

BUSY, TTO (BUSY, FLAG) - In the right pair of lights, BUSY is on while a character is being transmitted to the printer. When transmission is complete, BUSY goes off and TTO goes on, requesting an interrupt. TTO goes off when the program supplies a new character.

PAPER TAPE READER -

BUFFER (PTR0-35) - Contains data read from tape but no yet retrieved by the program. The 36 buffer bits are numbered to correspond to the bus lines. Characters of six or eight holes from tape are brought in at the right end of the buffer, with bit 35 receiving hole 1.

BIN (B) - When this light is on, each reader operation reads hole 6-1 of only those characters in which hole 8 is punched and assembles six such characters into a 36-bit word. When each character is brought into the buffer at the right, the previously read characters are shifted left. When BIN is off, each read operation retrieves a single character, sensing all eight holes.

BUSY, FLAG - When BUSY goes on, the reader goes into operation retrieving information in the manner specified by BIN. When the required number of characters is retrieved, BUSY goes off and FLAG goes on, requesting an interrupt. FLAG goes off when the program retrieves the information from the buffer. FLAG is also set when the operator turns the reader motor on or off.

CARD READER -

BUFFER (CR0-35) - The 36 bits of this buffer are numbered left to right to correspond to the bus lines. Six-bit characters from cards are brought in at the right end of the buffer over reader signal lines 1, 2, 4, 8, A, B, with buffer bit 35 receiving information from line 1. The program specifies whether information retrieved from a card is to be placed on the bus in units of one, two, or six characters. If more than one character is to be read per bus transfer, previously read characters are shifted left in the buffer as new ones come in.

BIN (B) - While this light is on, a card is read in binary mode wherein each column is read as two characters. The first character is from the lower half of the column (holes 4-9) with hole 9 on reader line 1; the second character is the upper half (holes 12, 11, 0, 1, 2, 3) with hole 3 on line 1. If the light is off, reading is in alphanumeric mode in which the reader converts the Hollerith character in a column to the Burroughs 6-bit code, and six columns are required to fill the buffer.

BUSY - This light goes on when the program requests that the reader begin a card cycle, and the light remains on until the entire card is read.

FLAG - Each time the buffer contains the amount of information specified by the program, this light goes on, requesting an interrupt. Retrieving the data turns FLAG off and clears the buffer, which is cleared automatically if the program does not respond to the request before the next column is read. FLAG may light after each column or only when the buffer is full; depending on whether BIN is off or on respectively, a single column is one or two characters, and a full buffer contains six or only three columns. FLAG also signals that the reader has finished an entire card regardless of the number of characters in the buffer, or that a card jam has occurred.

CARD DONE - Lit from the time the reader completes one card cycle until it starts another. It is possible for CARD DONE and BUSY to be lit at the same time. The program turns on BUSY to cause the reader to begin a card cycle, but CARD DONE remains on until the cycle actually starts.

EOF - When the card hopper is empty, pushing the END OF FILE button on the reader turns on this light. It also turns on FLAG, requesting an interrupt.

ERR (CREL) - Indicates a validity check or read check error in the reader.

3.2 OPERATION OF IN-OUT EQUIPMENT

This section describes the normal operation of the photoelectric perforated tape reader, paper reader, paper tape punch, keyboard-printer, and card reader. Information for other devices is included in their maintenance manuals and in the operator manual, PDP-6 Operation. DEC also supplies manufacturer manuals for all devices included in a PDP-6 system.

a Tape Reader

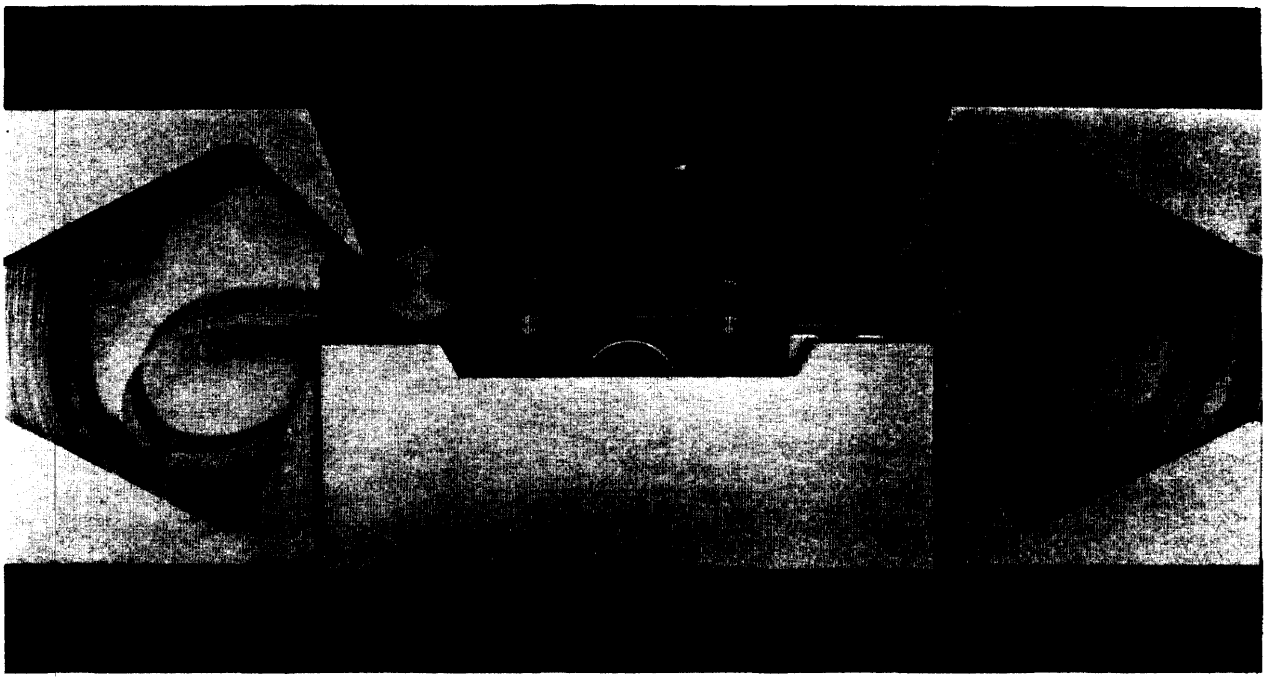


Figure 3-4 Paper Tape Reader Type 760

Before loading a tape in the reader, turn off the reader motor by pressing the READER OFF key. This releases the brake so that tape may be inserted, and it also requests an interrupt on the reader channel to inform the program that the reader is unavailable. Place the fanfold tape stack vertically in the bin at the right with the tape oriented so that the front end of the tape is nearest the read head and the feed holes are nearer the reader mounting panel, i.e., away from the operator. Take three or four folds of tape from the bin and slip the tape into the reader from the front so it is threaded as shown in Figure 3-4. Make sure that the part of the tape in the left bin is placed to correspond to the folds, otherwise it will not stack properly.

Once the tape is properly loaded lift the READER ON key to start the motor and energize the break. This also requests an interrupt to inform the program that the reader is on. When using the readin mode loader, always turn on the reader before starting the loader. The program makes use of the reader by sending signals to the clutch, which moves the tape past the sensing photocells. After the program has finished reading the tape, run out the remaining leader by lifting the READER FEED key, or turn the reader off so the tape may be slipped out directly.

b Tape Punch

The punch is located in a drawer at the top of the left console bay. The punch mechanism faces the right side of the drawer. Fanfold tape is fed from a box as shown in Figure 3-5. After punching, the tape moves into a storage bin from which the operator may remove it through a slot on the front of the drawer.

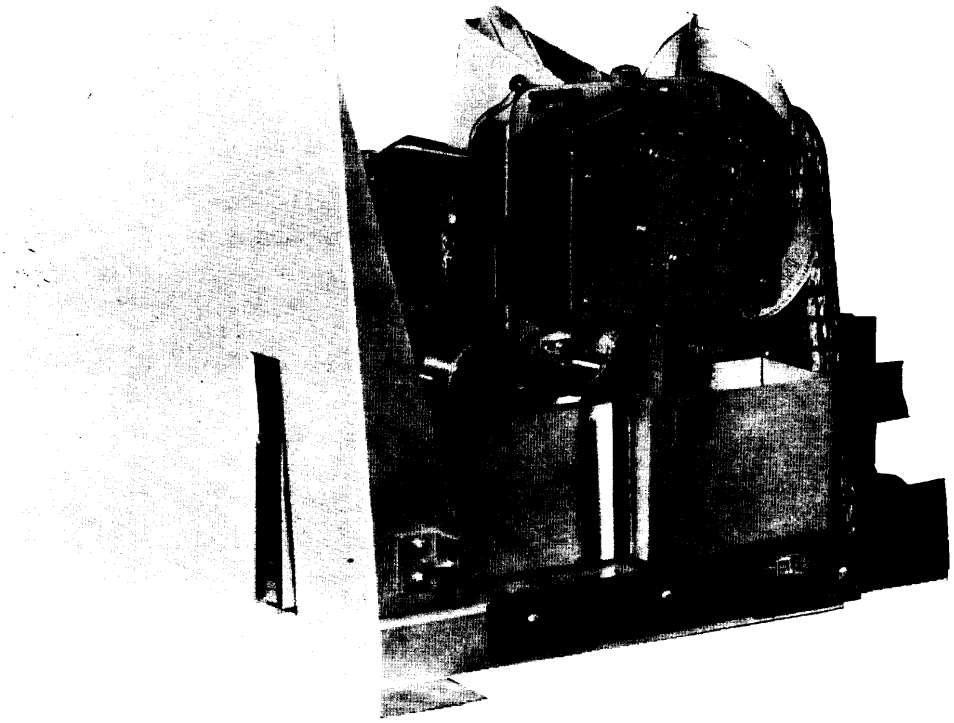


Figure 3-5 Paper Tape Punch Type 761

To load the punch, first empty the chad box below the punch mechanism. Then tear off the top of a box of fanfold tape (the top has a single flap; the bottom of the box has a small flap in the center as well as the flap that extends the full length of the box). Set the box in the frame at the right side of the punch and thread the tape through the mechanism as shown in Figure 3-5. The arrows on the tape should point in the direction of tape motion. If they point in the opposite direction, the box was opened at the wrong end; remove the box from the frame, seal up the bottom, open the top, and thread the tape correctly. After loading the tape, hold down the console PUNCH FEED key long enough to feed approximately 18 in. of leader. Make sure the tape is feeding and folding properly in the storage bin.

To remove a length of perforated tape from the bin, first hold down PUNCH FEED long enough to provide an adequate trailer at the end of the tape (and also leader at the beginning of the next length of tape). Remove the tape from the bin and tear it off at a fold within the area in which only feed holes are punched. Make sure that the tape left in the bin is stacked to correspond to the folds; otherwise, it will not stack properly as it is being punched. After removal, turn the tape stack over so the beginning of the tape is on top, and label it with both name and date.

c Teletype Keyboard-Printer

The teletypewriter (Figure 3-6) provides two-way communication between operator and computer. It is actually two independent devices, keyboard and printer, which may be operated simultaneously. The equipment operates at speeds up to ten characters per second, with 8-bit characters plus start and stop control signals transmitted serially. Located at the right front of the unit is a 2-position rotary switch, OFF/LINE. When this switch is set to LINE, the unit is on line and it goes on and off with system power.

The keyboard resembles that of a standard typewriter with four rows of keys and a space bar. Striking a key transmits a character to the Teletype control unit connected to the bus, but the character is printed or the function executed only if the processor sends it back to the printer. The line feed moves the carriage only vertically with a spacing of six lines to the inch. The return moves the carriage to the left margin but does not feed a line: to start a new line the operator must strike both return and line feed. Codes for the characters on the lower parts

of the key tops can be transmitted merely by striking the keys. Codes for printable characters on the upper parts (punctuation, ampersand, percent sign) are transmitted by holding down the shift key when striking the character key. Control codes are transmitted by holding down the control key, CTRL, when striking the appropriate character key. Codes for all characters listed on the keyboard and some that are not can be transmitted to the computer, but codes for some of the control functions have no effect on the printer when sent back. Table A4-2 lists all codes, their ASCII assignments, and the key combinations required to transmit them. Because of recent changes in the code, there may be slight differences in the printing characters associated with certain key positions. In such cases alternate characters are listed in parentheses.

Figure 3-6 Keyboard-Printer Type 626

In line with the space bar below the keyboard are four red buttons. At the right end is the repeat button REPT. Pressing this button and striking any character key causes repeated transmission of the corresponding code so long as REPT is held down. Characters that require the shift key may also be repeated in this manner, but there is no repetition of control characters.

The red button on the left, BRK RLS, is not connected in the console teletypewriter. The remaining two buttons, LOC LF and LOC CR, are the local line feed and carriage return. These buttons affect the printer directly and do not transmit codes over the bus.

Paper installation, ribbon replacement, and the procedure for setting horizontal and vertical tabs are described below. All references are to figures in typing unit section 574-220-100 in Vol. 1 of Teletype Bulletin 281B (Technical Manual 35 Keyboard Send-Receive (KSR) and Receive-Only (RO) Teletypewriter Sets).

Paper

The unit has a sprocket feed and uses 8-1/2 x 11 fanfold form paper. The supply is held in a tray at the back of the unit, and printed forms can be torn off against the edge of the glass window in front of the platen. To replace the paper first remove the upper cover by pressing the cover release button on the right side. To free the remaining old paper for removal, lift the paper guides by pushing the handle marked PUSH at the right of the platen. To insert new paper from the tray, offer it up below the platen at the rear, lining up the holes at the edges of the paper with the sprockets, and press the local line feed button to draw the paper in under the platen.

Ribbon

Replace whenever it becomes worn or frayed or when the printing becomes too light. Disengage the old ribbon from the ribbon guides on either side of the type block, and remove the reels by lifting the spring clips on the reel spindles and pulling the reels off (the ribbon feed mechanism is called out in Figure 4). Remove the old tape from one of the reels and replace the empty reel on one side of the machine; install a new reel on the other side. Push down both reel spindle spring clips to secure the reels. Unwind the fresh ribbon from the inside of the supply reel, over the guide roller, through the two guides on either side of the type block, out around the other guide roller, and back onto the inside of the take-up reel. Engage the hook on the end of the ribbon over the point of the arrow in the hub. Wind a few turns of the ribbon and make sure that the reversing eyelet has been wound onto the spool. Make sure the ribbon is seated properly and feeds correctly in operation.

Tabs

The horizontal and vertical tabulator mechanisms are also called out in Figure 4. Each is a slotted wheel surrounded by a spring on which are mounted a number of tab stops. The horizontal tab mechanism is shown in detail in Figure 47. The slotted wheel is mounted on the spacing drum, and a tab can be set by inserting a tab stop in a groove where it catches the tabulator pawl when the type block carriage is in the desired position. With needle-nose pliers or equivalent, lift the tab stop out of the slot in the wheel against the spring tension. Slide the stop along the spring in the desired direction, and reinsert it into the slot at the new location. A stop may be removed from use by turning it so that it does not catch the pawl. Figure 49 shows the vertical tabulator mechanism. The slots in this disc allow vertical tabs at any desired line, but adjacent tabs must be at least 1 in. apart.

d Card Reader

The B122 Card Reader handles 200 cards per minute and has a hopper and stacker capacity of 500 cards. With a trivial change in the control unit logic, the processor can control the B124 Card Reader. Its operation is similar to that described here, but the maintenance information given in Chapter 9 applies only to the B122. The B124 handles 800 cards per minute and has a hopper and stacker capacity of 2000 cards. In both machines the cards are read lengthwise and sensed photoelectrically. In Figure 3-7, the hopper is at the right, the stacker at the left; in the center is a console that contains the operating buttons and indicators. Of the following four indicators, the first is white, the other three are red error indicators.

NOT READY Indicates one or more of the following:

START button has not been pushed

Hopper empty

Stacker full

Card jam (feed check)

Read check

Validity check (only when VALIDITY ON button is lit)

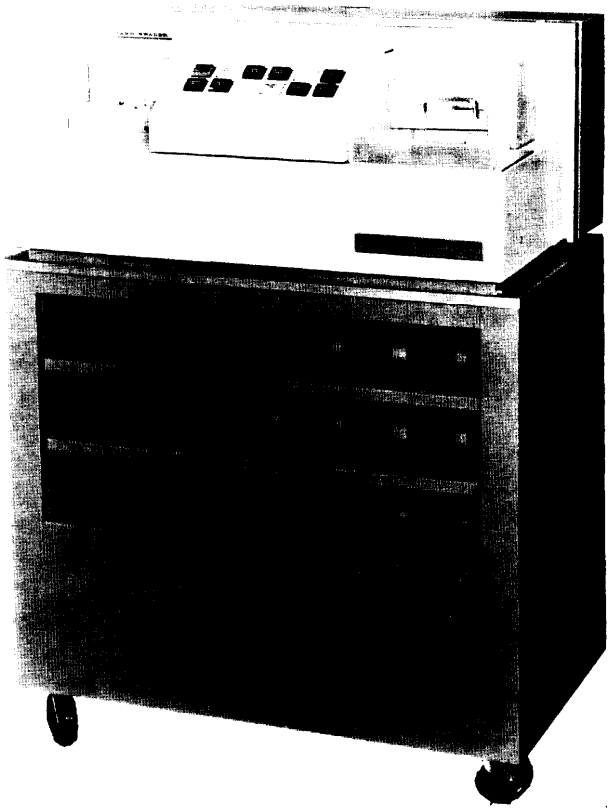


Figure 3-7 Card Reader Type 461

The reader cannot respond to the program while NOT READY is lit. The program may check a not-ready status bit to determine if the reader is available. This status bit is 1 on any of the above conditions and also when the reader power is off (the console cover is interlocked to turn power off when not in place).

READ CHECK - Indicates a NOT READY condition due to a malfunction in the read circuits, e.g., exciter lamps, solar cells, photo amplifiers. There is no read checking during any card cycle; otherwise, read checking is continuous. When this light is on, the ERR light on the processor in-out panel is also on and the reader error status bit is 1.

FEED CHECK - Indicates a NOT READY condition due to a jammed card or failure to select a card. The drive motors stop as soon as the light goes on. A signal to the processor turns on the FLAG light on the in-out panel, requesting a priority interrupt on the channel assigned to the reader; the signal also supplies a status bit to the program.

VALIDITY CHECK - This light functions only if the VALIDITY ON button is lit. It then indicates that an invalid punch combination has been read in alphanumeric mode. When this light is on, the ERR light on the processor in-out panel is also on and the reader error status bit is 1.

Located on the left side of the stacker is the main power switch that controls power to the reader auxiliary power supply. With this switch on, the reader may be turned on either at the reader console or by turning on system power at the processor console. The reader console contains seven buttons, three of which light; the first six below are momentary contact, the last is alternate action.

POWER ON - Green button which turns on the main power supply, the reader motors and, after a 3-sec delay, the reader control logic. This action is duplicated by turning on system power at the processor console. Button is illuminated when power is on.

POWER OFF - Turns off the main supply, but the auxiliary supply remains on.

START - Turns off the NOT READY indicator provided no other not-ready condition exists.

STOP - If the reader is in operation, this button turns on the NOT READY light and the reader stops when the current card runs out to the stacker.

RESET - Turns off the three red check lights: READ CHECK, FEED CHECK, VALIDITY CHECK.

END OF FILE - Pressing this white button when the hopper is empty, lights the button and turns on the EOF and FLAG lights on the processor in-out indicator panel, requesting a priority interrupt on the channel assigned to the reader; the button light goes out when cards are placed in the hopper. The signal generated by the button can be checked as a status bit by the program.

VALIDITY ON - This is an alternate-action yellow button that is lit when on. An invalid punch combination read in alphanumeric mode is sent to the reader control unit as an all-zero character; when the switch is on, and invalid punch also lights VALIDITY CHECK and stops the reader.

In addition to the above there are also interlocks in the hopper and stacker and a LOCAL RUN toggle switch under the cover; the interlocks generate the NOT READY condition when the hopper is empty or the stacker is full. Raising the toggle switch (local) causes the reader to feed cards continuously until the hopper is empty, and then stop with the FEED CHECK light on. With the switch in remote (down), card cycles can be started from the processor whenever the NOT READY light is off.

For operation off line, turn on the main power switch and press POWER ON; after 3 sec NOT READY should light. For normal operation on line, the main power switch is turned on at system power turnon and the reader is left on with NOT READY lit whenever it is not in use. Cards should be placed in the hopper face down with the 12 edge toward the operator. Place the plastic weight on top of the deck to prevent jamming as the last few cards are read. If any of the red check lights are on, push RESET. Push START to turn off NOT READY, and the reader is then available to the processor. STOP may be pushed at any time to generate a NOT READY condition, causing the reader to stop at the end of the current card. To continue, push START.

When reading is in alphanumeric mode, every column is checked for a valid Hollerith character. If an invalid punch is encountered and the VALIDITY ON button is lit, a VALIDITY CHECK error is indicated and the reader stops at the end of the current card. If a second attempt to read the card fails, check it for improper punches. Table A4-3 lists the Hollerith character codes and all invalid punch combinations (note that the Burroughs code is incompatible with that used by the IBM 029 Card Punch).

If the reader stops with READ CHECK lit, the self-checking circuits have detected a malfunction in the read circuitry. Usually this is either the failure of a lamp or solar cell and may be temporary. If both READ CHECK and FEED CHECK go on together, the reader is in need of adjustment. There is one section of the read circuitry in which a failure is not detected as a read check error. However, a failure in this particular part would cause incorrect timing resulting in a validity check error if reading is alphanumeric. If no invalid punches can be found on a card, but several attempts to read it result in a validity check error, it is likely to be a malfunction in this unchecked part of the read circuits. For particulars refer to the manual for the Burroughs B122 Card Reader.

When a card cycle begins, the card is first contacted by a knife at the bottom edge of the hopper and is pushed into the read station where the feed operation is taken over by rollers. The most probable point for a feed malfunction to occur is at the entry into the read station. If the card is bent, it may jam in the feedways; if the trailing edge had been damaged by frequent handling, the pickup knife may fail to move the card through the rollers. When a card fails to appear at the read station in the prescribed time, the FEED CHECK light goes on and the drive motors stop. Do not attempt to reread a worn or damaged card that has caused a feed check error, but put a duplicate in its place.

In the unlikely event that a card should jam inside the read station, no FEED CHECK is indicated, but no cards are processed either (if reading is in alphanumeric, a card stuck in the read station may produce a validity check error). To check for jammed cards, remove the head cover by lifting it at its base and pulling it out horizontally (an interlock removes power if it has not already been turned off). If necessary, the photocell head can be removed by moving the knurled-head sliding bolts to their vertical positions and squeezing them toward the center. The head can then be lifted straight up. Be sure when replacing the head that it is seated properly and the knurled-head bolts are fully engaged.

3.3 PROCESSOR OPERATING PROCEDURES

After turning on system power at the processor console, check the memories and peripheral equipment connected to the memory and in-out busses. In general all memories should go on with system power unless a single unit has been taken off line deliberately. Whether a particular peripheral device is on depends upon its own organization. Most in-out control units go on and off with system power; however, in some instances power supplies must be turned on and off independently of the processor.

a Read In

In order to allow initial information to be brought into memory, a readin loader is usually kept permanently in the part of core that is ordinarily inaccessible because of the fast memory. To use the readin loader, set the appropriate starting location in the ADDRESS switches and lift the READ IN key. This turns on the RIM SBR light at the top of bay 1, and while the light is on,

any memory address from 0 to 17 provides access to core instead of fast memory. The light goes out and the processor leaves readin mode whenever an instruction is retrieved from any location above 17. Read in can be single stepped using the CONTINUE and STOP keys, but any other key takes the processor out of readin mode.

To deposit a loader in the bottom of core, the operator must make use of a small toggle switch labeled RIM MAINT, which is mounted behind the double doors on bay 2, on a bracket at the left end between mounting panels 2L and 2M. Putting this switch up holds the processor in readin mode regardless of any action taken at the console, so the operator may deposit the loader. Place the first word in the DATA switches, set the ADDRESS switches to the first location (0 is most convenient), and lift DEPOSIT THIS. MEMORY displays the word deposited. If the remaining words are in consecutive locations, they may be deposited in order by setting them in the DATA switches and pressing DEPOSIT NEXT for each. Although all words are displayed when deposited, it is a good idea to check the entire loader by going through it first pressing EXAMINE THIS, then EXAMINE NEXT. After the loader has been deposited, turn RIM MAINT down.

b Operating Keys

The operator should check material accompanying each program for information on halts, tape requirements, and so forth. Every program is begun by either START or READ IN. On a halt the operator should make note of the console lights, particularly PROGRAM COUNTER, and do whatever is requested in the program operating instructions; the operator may restart by pressing INSTRUCTION CONTINUE.

To debug programs, INSTRUCTION STOP may be latched on and the program single stepped using INSTRUCTION CONTINUE. Or with the REPEAT switch on, and both INSTRUCTION STOP and INSTRUCTION CONTINUE latched down, the program speed can be varied by the SPEED controls. By similar use of the MEMORY STOP and MEMORY CONTINUE keys, a program may be single stepped from one memory call to the next; low-speed operation can be effected by using REPEAT.

The keys for the reader and punch may be operated at any time whether the processor is running or not. The stop keys may also be pressed at any time, but ordinarily these are used only for

single-step or low-speed operation (for special stop considerations see c below). The remaining ten keys use the key cycle to initiate some operation, if only a clear function as is the case with IO RESET. For seven keys, entry into the key cycle is gated by the 0 state of RUN so that inadvertent key manipulation can have no effect while a program is running. Even through RUN remains on throughout memory single stepping (unless of course a program halt should occur), the program can be restarted by pressing MEMORY CONTINUE; at a memory stop MEM STOP is on. Of the initiating keys, only EXAMINE THIS and DEPOSIT THIS have any effect while the program is actually running, i.e., there is neither an instruction stop nor a memory stop. Either of these keys inserts a key cycle between two instructions without stopping the processor. However, do not use these keys if there is any chance of a program halt occurring. The halt instruction stops the processor by clearing RUN at the beginning of the execute cycle, and it is thus possible for a key cycle to be triggered between the time RUN is cleared and the instruction is completed (of course the same caution holds for any wanton key manipulation while the processor is running). If a program halt should occur (RUN goes off) at the same time that EXAMINE THIS or DEPOSIT THIS is operated, check the corresponding pair of SYNC and START lights in the left column on the bay 1 indicator panel. If both lights are on, the corresponding key function was not performed; if neither light is on, the key function was performed prior to the halt instruction. If only a SYNC light is on, the key function was performed but there is no way of knowing whether it was executed during the halt or after, and thus the console lights are meaningless. If EXAMINE THIS or DEPOSIT THIS is pressed with RUN off, the SYNC light does not go out but the key function is performed.

There are also special precautions that must be observed while user programs are sharing processor time. While RUN is on, relocation and protection are inhibited during a key cycle so the operator may use EXAMINE THIS and DEPOSIT THIS with all of memory available to him. However, when the processor is stopped, as between instructions in single-step operation, the operator must make sure his actions at the console are compatible with the operating mode. In user mode any address supplied for a key function, including addresses in an instruction initiated by the EXECUTE key, must be smaller than the block size as indicated by the MEMORY PROTECTION lights at the top of the in-out panel, or a priority interrupt for an illegal address will occur. The operator should also understand that unless the address is for fast memory, it is relocated to the block specified by MEMORY RELOCATION (the RELOCATED ADDRESS is

displayed at the right). Furthermore, an illegal instruction executed from the console will take the processor out of user mode. The operator must observe the lights at the top of bay 1 to determine what he can do. All addresses and instructions are legal if EXEC MODE is on. If this light is off but PI CYC is on (this can happen only between two PI cycles required for the same interrupt), there is also no relocation or protection. However, in this circumstance not all instructions are legal; an IOT may be executed from the console, but any other illegal user instruction or a UUO will return the processor to executive mode. If both EXEC MODE and PI CYC are off, the operator must observe all user restrictions. The operator can switch from either mode to the other by executing the appropriate instruction from the console. The switch may be made from user to executive mode by pressing IO RESET, but this also clears most of the in-out equipment including all PI assignments.

Care should be exercised in the use of the EXECUTE key whenever priority interrupts are allowed while a program is being single stepped. In addition to observing the user-executive restrictions associated with priority interrupts, the operator must be aware of the following. An interrupt has priority over any instruction including one executed from the console. If an interrupt request is waiting when EXECUTE is pressed, the processor performs the PI instruction instead of the one in DATA switches. If PI CYC is on (indicating that the preceding instruction was a block IOT that overflowed), a non-IOT executed from the console will be taken by the processor to be the jump to the break routine. This will turn off PI CYC, and the JSR to the subroutine will be skipped when the program is continued. Furthermore, the processor will "hold" the break—i.e., PI REQUEST goes off, PI IN PROGRESS goes on, and both the channel on which the interrupt was requested and all lower priority channels will remain disabled as the program continues.

CAUTION

> Never under any circumstances press more than one initiating key at a time because the processor will try to perform both functions at once. Note that in low-speed operation one of the continue keys is always on, so although EXAMINE THIS or DEPOSIT THIS can be used while the processor is running, they cannot be used in low-speed operation.

c Emergency Stop

Ordinarily INSTRUCTION STOP is used for single step operation and maintenance procedures, but it can also be used for an emergency stop if the processor should get caught in a loop. For this purpose the turnon of the key triggers a pulse generator whose output triggers a one-shot that temporarily inhibits certain pulses in the instruction and address cycles; then the processor can stop at one of these points if a loop prevents its reaching the end of a main sequence. The pulse generator output also clears RUN in case a hardware malfunction should disable the normal clearing in the execute cycle. Both features apply only to the turnon of the key so that once it has been latched down it will not interfere with single step operation. Once RUN is clear, IO RESET can be used to clear the computer.

If the processor should hang up or be running without seeming to accomplish anything, do not call a DEC Technical Representative until certain routine checks have been made, as it is possible for an inept programmer to hang up the machine. Although it is recommended that only BLKI, BLKO, or JSR be used in a PI cycle, nonetheless the processor will perform a PI cycle correctly for any non-IOT instruction. But a condition IOT will cause the processor to hang up in the PI cycle. When this happens, the processor repeats the instruction over and over: PC is static, the PI REQUEST light for the channel is on but the PI IN PROGRESS light never goes on, and PI CYC and PI REQ at the top of bay 1 remain on indefinitely (the AC lights display the IOT instruction code).

If the program attempts to retrieve an instruction from a memory that is not connected to the bus (and the DISABLE MEMORY switch is off), the lack of any instruction retrieved is interpreted by the processor as a UUO. The attempt to address a nonexistent memory usually results in an interrupt on the processor channel, but if the JSR for the break should attempt to go to the same memory, the processor would go into a loop.

Both of the above loops include the complete main sequence, so pressing INSTRUCTION STOP will cause a stop at the completion of an instruction. There are other program failures, however, that never allow the processor to finish an instruction. If a program should include an XCT that executes itself or if a programmer puts a UUO in location 41, the processor goes into a loop that keeps jumping back to the instruction cycle without ever completing a main se-

quence. The key thus halts these in the address cycle. If DISABLE MEMORY is on and the program attempts any access to a memory that is not on line, the processor hangs up in the memory subroutine. Interrupting the time chain cannot affect this situation, but INSTRUCTION STOP clears RUN, and the machine can then be freed by IO RESET.

Hardware malfunctions can cause loops that the time chain inhibit cannot stop. For example if PI CYC fails to set, an interrupt request will cause a loop in which the processor keeps trying to honor the request without succeeding. If MQ0 refuses to clear in a block transfer, the processor will loop forever, returning on each step to the fetch cycle. For any loop that includes a memory call, do not free the processor merely by clearing RUN and press-IO RESET. If the processor is within the memory subroutine when the reset occurs, it will very likely hang up a memory while freeing the processor. For this situation it is preferable to press MEMORY STOP and then check the lights to be sure the processor has not stopped following the read part of a read-write access. If it has, hold on MEMORY STOP and press MEMORY CONTINUE, so that it will stop following the subsequent write. Once the processor has stopped with no chance of hanging up a memory, press INSTRUCTION STOP to clear RUN, and then press IO RESET.

CHAPTER 4

DRAWING CONVENTIONS AND FLOW CHARTS

Accompanying each PDP-6 is a complete set of drawings, consisting primarily of D-size flow charts, logic drawings (block schematics), and wiring diagrams. Every drawing is labeled with both a DEC drawing number and a type code. The drawing number is in four parts separated by dashes (e.g., D-166-4-EX): the first part is a letter indicating size; the second is the type number of the equipment (usually three digits); the third is the drawing serial number (see next paragraph); and the last is a number or a mnemonic letter code specifying the individual drawing (the code may end with a number, sometimes preceded by another dash, if more than one drawing is required to treat a section of the logic). If a drawing includes several sheets, both the sheet number and the number of sheets are written at the lower left of the drawing number. If a drawing is revised after being signed by the project engineer, a revision letter is written to the right. To the left of the number is a type code; some typical codes are block schematic BS, system diagram SD, flow diagram FD, timing diagram TD, interconnection diagram ID, cable diagram CD, wiring diagram WD, power wiring PW, module list ML, utilization module list UML, master drawing list MDL, cable list CL, wiring list WL (the last three are usually A size).

In general, the only drawings included in the manual for a given piece of equipment are the associated flow diagrams and logic drawings. The maintenance chapter of each manual does, however, describe the other types of engineering drawings and their use. Drawings in the manual are intended for instruction purposes only; personnel working at the machine should use the prints for the equipment rather than the figures for the manual. Drawings that are reduced to B size and printed in this and other manuals for the PDP-6 system are serial 0, corresponding to the standard production machine. Although every unit of a given type is assigned a different serial number, most of the prints accompanying the equipment have drawing serial 0. But if a particular unit differs in some way from standard, those drawings that reflect the difference have the same serial number as the lowest numbered machine that is so modified. Therefore, although each manual contains the drawings for that portion of the standard system that it describes, maintenance personnel should use the prints for work on the equipment because they show any variations peculiar to the installation.

All drawings included in a manual are assigned figure numbers by chapter. These numbers are also written on the prints in the lower right, above the drawing legend. To differentiate drawings associated with one manual from those of another, a code designating the manual appears in front of the figure number (the code may not be included on the figures reproduced in the manual). The letters "AP" indicate a figure for this manual, which includes not only the Arithmetic Processor Type 166 but also the control units for four in-out devices Types 461, 626, 760, and 761. Figures for the memory manual are prefixed "M" and show both the fast and core memories. Figure numbers on drawings for in-out devices described in separate publications are usually prefixed by the appropriate type number.

The complete system logic for the arithmetic processor and the common in-out devices is shown in a series of flow charts for Chapter 4 and a series of logic drawings for Chapters 5, 6, 7, and 8. This chapter describes the conventions and notation used in these drawings.

4.1 LOGIC DRAWINGS

The logic drawings are block diagrams that show the function of every logic element used in the computer. They also show the type of signal present at any module connector pin that carries a logic signal or some special voltage level. The standard power and ground pins (A to D on every module) are not shown. In addition to showing the function of every logic element, the drawings indentify every circuit by type and by physical location. Circuit type is always identified by the type number as given in the DEC module catalog. Below the type number is a location code made up of one digit, one letter, and one or two digits. For example, the location code 2F10 represents plugin unit connector 10 in mounting panel F in bay 2 (in the lettering on the logic drawings, each numeral "0" has a slash through it to distinguish it from letter "O"). Pin designations may be formed merely by adding the pin letter to the module location code, e.g., 2F10H.

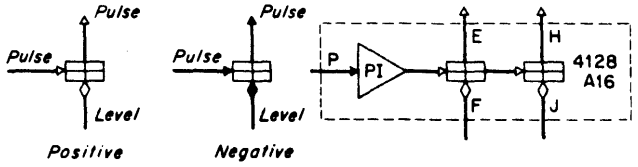
The frame containing the arithmetic processor and the console includes bays 1 to 4. Each memory, or peripheral device requiring a major portion of a bay, is designated as bay 1. If an in-out device control unit requires only a few logic mounting panels, it has no bay number; the panels are designated A, B, ..., even though they may be mounted in any position. For the smaller block symbols, such as those representing single inverters and capacitor-diode gates, the circuit type number and location code are written near the symbol, and the inputs and

outputs are labeled by connector pins. With all larger blocks, the circuit and location information are written inside the block. If several logic elements from the same module appear together in a drawing, they may be enclosed in a dashed line: the location and type number are then written only once within the module boundary and the pin letters are written just inside the boundary where the signal lines cross it.

Some modules have connectors on both front and back; pins on the rear connector are identified by the prefix "R." Some modules are double height, with two front connectors. The location of such a module is given by two panel letters, e.g., 2DE17, and the front connector pins are prefixed by the appropriate panel letter (pins on this module would be designated 2DE17-DT, 2DE17-EB). Only the upper connector receives the power lines, so pins A to D of the lower connector are available for logic signals. Such modules usually also have two rear connectors, the upper one identified by the prefix "R," the lower by "S." Thus for a double-height module mounted in 2DE, the R connector is at the rear of panel D, the S connector at the rear of panel E.

On the logic drawings, the type of signal present at a pin connection is shown by a triangle or diamond. In DEC convention, timing is provided by pulses whose polarities are shown by open and closed triangles. These polarities depend only upon input requirements and represent no logical difference. Similarly, gating levels are represented by open and closed diamonds that represent the assertion polarity that satisfies the gate; neither voltage level categorically represents 1 or 0, true or false. A given logic function may have different assertion levels in different places depending upon gate input requirements. For example, if a function has a negative assertion level, the function is considered true when the line corresponding to it is at −3 vdc; for ground assertion, the function is considered true when the line is at ground. Sometimes a line carrying a logic level is shown connected to the input of a pulse amplifier or capacitor-diode gate, which produces a pulse output. In these cases, the output is triggered by a level transition at the input. If the input is shown as a diamond, triggering occurs at the leading edge, i.e., the diamond shows the assertion polarity of the logic function immediately after the triggering transition. An event triggered on a trailing edge is indicated by a composite symbol with a diamond showing the assertion polarity of the level, and a triangle showing the opposite polarity required for the input pulse (i.e., the triangle indicates the direction of the transition when the logic level is negated). Sometimes a leading edge is shown by a composite diamond and triangle of the same polarity, but this is not necessary. Occasionally in the in-out equipment,

a wide pulse is used to produce a delay by triggering events on the pulse trailing edge. The composite signal for this is a pair of triangles, the first showing the pulse polarity, the second the polarity of the triggering transition. Any nonstandard signal is shown merely by an arrow pointing in the direction of signal flow.



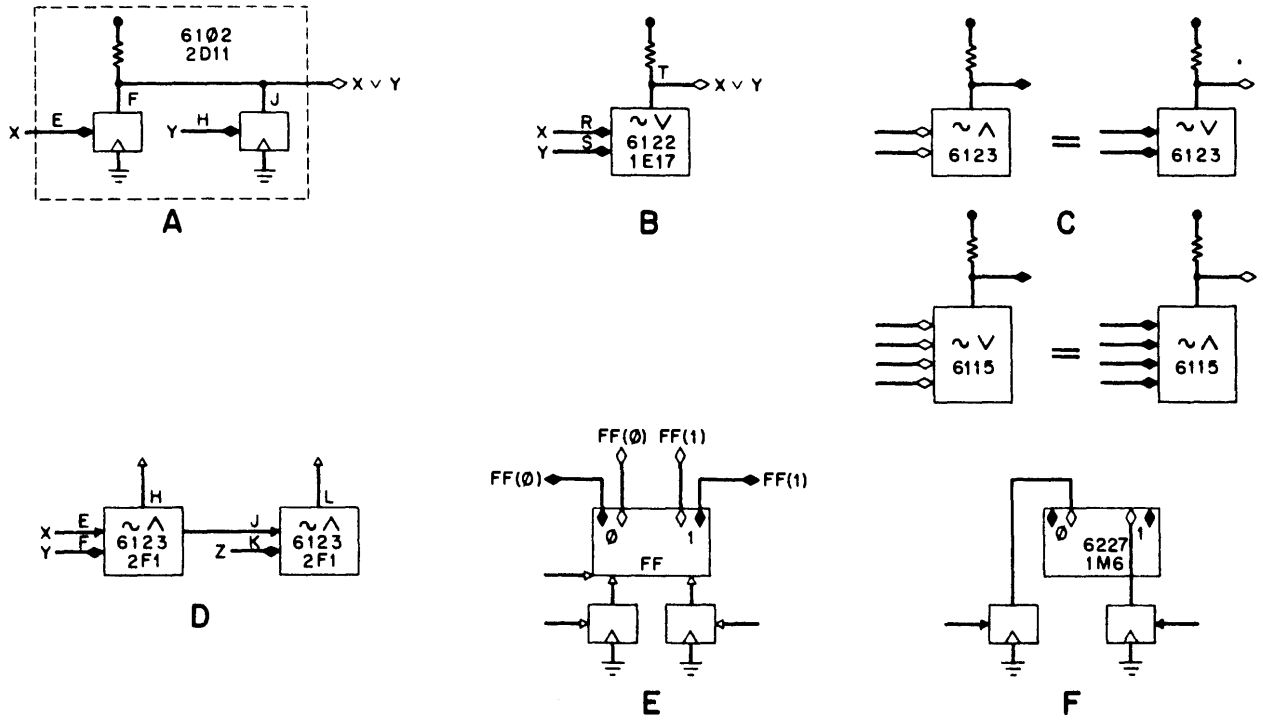


Figure 4-1 Logic Symbols

The upper part of Figure 4-1 shows the signal designations and the symbols for inverters and capacitor-diode gates. These are described fully in the introduction to the DEC module catalog. With the symbols in Figure 4-1 are examples of type numbers, location codes, and connector pin letters. The remaining lettered sections of Figure 4-1 show other conventions for the PDP-6 drawings.

The logic drawings show the function of every logic element in the simplest way consistent with the requirement that every pin connection be shown. Thus if two single inverters are connected to form an AND or OR gate, the individual inverters are shown in the drawing as indicated at A in Figure 4-1. However if the gate is produced by connections internal to a module, such as a pair of inverters with a common collector pin and internally grounded emitters, then it is shown by a block labeled for the appropriate logic function as shown at B. Blocks are used to represent inverter gates with as many as four inputs and diode gates with as many as eight (the in-out equipment uses capacitor-diode modules in which the gates have common pulse connections, but the individual gates are always shown). Since all such nets invert (a diode gate includes an inverter output) the blocks are always labeled "$\sim\wedge$" or "$\sim\vee$." The tilde ($\sim$) in these labels has no actual connection with logical function and may be ignored when learning the logic. Because an assertion level may be inverted without affecting its truth value, all gates are AND ($\wedge$) or OR ($\vee$) gates, and the tilde merely indicates that the output assertion level is of opposite polarity to the inputs. Each block also contains a location code and a type number; input and output pin connections are labeled in the usual way. Since the logical function of a gate depends both upon its logical configuration and the assertion polarities of its inputs, a given net may be used as either an AND or an OR gate—the equivalence is shown at C. Of course, a single level input may be replaced by a pulse in an AND gate, and all levels may be replaced by pulses in an OR gate. Inputs to a gate are generally at the left, outputs at the top, and a signal shown passing through a gate from left to right implies no logical change: D shows a pair of AND gates in which the pulse X, though labeled only once, is an input to both gates, whereas levels Y and Z each enter only one.

All other circuit elements except flip-flops appear as blocks that contain a mnemonic abbreviation of the circuit function. Some examples are delay DLY, pulse amplifier PA, pulse generator PG, clock CLK, bus driver BD, solenoid driver SD, majority gate MAJ, dc adder DCA.

Clocks, pulse amplifiers, pulse generators, and some one-shot delays have a pair of transformer-coupled pulse outputs (usually shown at the right of the block). When the input to such a circuit is triggered, a positive-going pulse appears at the positive output if the negative output is grounded, or a negative-going pulse appears at the negative output if the positive output is grounded. One-shot delays other than the 4303 have a logic level output that is asserted negative during the delay period. The symbol that represents a flip-flop is shown at E (the same type of symbol also represents the 4303 Integrating Delay and the 6131 DC Adder). In this rectangle, terminals S and T are drawn twice, showing the polarities associated with either state of the flip-flop. In normal convention the "0" is at the left and the 0-out terminal is represented by the left diamond in both pairs. Some flip-flops have a separate output terminal, represented by a fifth diamond, to drive an indicator.

The principal advantage in showing the two states at both assertion levels is that there is never any need to invert the name of a signal that appears as an input to a logic net: all logical conditions appear in the drawings with correct truth values. When a flip-flop output is used as the input to a logic net, the signal name indicates the enabling state of the flip-flop. To determine the physical source of the signal (the output terminal to which the signal line is connected), one must know both the signal name and the assertion level. For example, the signal FF(1) at negative assertion originates at the 1-out terminal of flip-flop FF; at ground assertion this signal actually originates at the 0-out terminal.

Two gatable inputs are shown at the bottom of the rectangle at E, with the 0-in terminal at the left. Direct pulse inputs (i.e., those that are not gated) are at the sides of the flip-flop: in the example a direct clear input is shown at the left. A flip-flop may also have a complement input, which is drawn at the bottom center. An unbuffered flip-flop may be set or cleared by grounding one of the flip-flop collectors; such a function may be represented in either of the ways shown at F.

Connections between flip-flops are shown in various ways on the logic drawings, always in the simplest way consistent with showing all pin connections. The clear line for an entire register is usually shown entering the lower left corner of the leftmost flip-flop, then out of the lower right corner on to the next flip-flop, and so on through the register. If the flip-flops in a

counter have count inputs and outputs, the count pulse is usually shown going from right to left, entering the center of the right side of each flip-flop and leaving at the center of the left (e.g., see the program counter, Figure 5-11). If flip-flops from different modules are connected for shifting, all of the shift gates must be shown; however, if the flip-flops within a module are connected internally as a shift register, the shift signal is shown in the same way as a count (e.g., the shift register modules in Figure 8-7). In all processor registers, most transfer input gates for a given flip-flop are included on the module containing the flip-flop. These gates are therefore shown as small rectangles with logic symbols and pin connections but without location codes or type numbers (refer to AR, Figure 6-5). In many instances no pin numbers are included because the input connections are internal to the module: e.g., the AR, MB, and MQ flip-flops are on the same modules and the connections shown between them have no pin numbers. If these internal logic gates are similar to the regular inverter and diode gates, i.e., if all inputs to a given gate are of the same polarity and the output is inverted, the block is labeled with the appropriate symbol, either "$\sim\wedge$" or "$\sim\vee$." However, in many cases there are nonstandard gates, e.g., one in which a ground level is gated by a negative pulse to produce a ground output. Such gates are labeled merely by logic function, "$\wedge$" or "$\vee$," and no attempt is made to indicate signal polarities other than the diamonds and arrows used for inputs and outputs. In some cases pulse inputs to individual bits of a register are made through NPN emitter followers. Since these perform no logical function and do not even change signal polarity, each is shown on the logic drawing merely as a small circle at the pin to indicate that the signal at the flip-flop input terminal through other input gates is not available at that pin.

State changes in the 10-mc flip-flops take place more rapidly than the duration of the input pulses. To compensate for this, many gated inputs to these flip-flops are made through delay elements, which are not shown on the logic drawings. Since in many instances the outputs of a register flip-flop condition its inputs, the flip-flop state change is delayed until the termination of the input pulse to prevent logical race problems.

In addition to the many modules containing flip-flops, pulse amplifiers, etc., there is also a hybrid module, the subroutine card SBR, which includes three circuits each containing a flip-flop and a gated pulse amplifier. The flip-flop 1 output is an input to the AND gate at the PA input; the PA output, besides being available at a connector pin, is connected internally to the

flip-flop clear input (see the SBR at the left in Figure 5-2). An SBR (or its equivalent) is used whenever a subroutine is called from any time chain. The same pulse that triggers the subroutine also sets the flip-flop in an SBR, enabling the input gate to the PA; the other input to the PA gate is the return pulse from the subroutine. At the completion of the subroutine, the return pulse triggers the PA, whose output both clears the SBR and restarts the time chain.

All logic drawings are laid out with rectangular map coordinates, numbered 1 to 8 from left to right and lettered A to D from top to bottom. Because a single drawing may contain a number of logic elements, coordinates are often included in figure references. For example, a reference to the circuit in "Figure 5-6B6" would mean the circuit located in block B6 of logic drawing 5-6.

4.2 SIGNAL NOTATION

All signal names in PDP-6 are mnemonics that indicate both the function of the signal and its source. Each register with associated logic and each control system, whether it occupies several drawings, one drawing, or only part of one, has a single mnemonic code of one to three letters, which appears in the drawing title and at the beginning of the name of any signal originating in this part of the logic. This source code may appear naturally as part of the signal name; if not, it is merely prefixed to the name. For example the arithmetic register AR and its associated transfer logic, flag logic, and AR subroutines require a number of drawings all with prefix code AR, and the pulse that shifts the contents of AR to the left is AR SH LT. On the other hand, the readin mode subroutine flip-flop, which is associated with read in and is shown on one of the drawings for the key logic, is designated KEY RIM SBR. All prefix codes and corresponding figures are listed at the left in Figure 4-2.

The name of a signal that transfers information from one register to another includes the names of the two registers with an arrow between them. The arrow invariably points to the left because the transfer logic is always associated with the receiving register and its name must therefore appear first in the signal designation. The name of a transfer signal specifies not only the registers and the direction of transfer, but also the type of transfer and the register bits involved if the signal acts on less than the entire register. Numerals representing register bits are merely appended to the register name; bit 8 in AR is AR8 and bits 0-7 in MB are MB0-7. Operations

that affect only half of a full-word register are indicated by appending LT or RT to the register name. The state of a flip-flop is represented by a numeral in parentheses, e.g., the 1 state of bit 8 in AR is AR8(1). Since transfers in effect transfer states, the type of transfer is also indicated by a symbol in parentheses following the register name. For example the signal that transfers all zeros into the bits of AR, i.e., the signal that clears AR, is AR ←(0). This action of course is not usually referred to as a 0 transfer but rather as the clear function. The actual transfer of zeros from register A to register B is as follows: the transfer pulse clears a given bit of B if the corresponding bit in A is in the 0 state. The pulse that produces this effect is B ←A(0). Since B now contains ones only in bit positions that originally contained ones and also correspond to ones in A, the 0 transfer therefore produces in B the AND function of A and B. Similarly, the transfer of ones from A to B, B ←A(1), produces the inclusive OR function. If B is cleared before the transfer, then after the transfer A and B both contain the same information and the pair of pulses B ←(0) and B ←A(1) transfer the contents of A to B. The same effect would be produced by setting all bits in B and then transferring zeros. If both the zeros and ones of A are transferred to B simultaneously so that B bits corresponding to zeros in AR are cleared and those corresponding to ones in AR set, no prior setting or clearing is necessary. This is a jam transfer and is written B ←A(J). Since the jam transfer occurs at a single point in time, it is possible to switch the contents of two registers: the outputs from A can provide the gating levels for the transfer into B while the B outputs gate the transfer into A. The signal that triggers both transfer pulses at once is labeled B(J) ↔A(J).

There are other types of transfers, such as B(0) ←A(1), which clears bits of B that correspond to ones in A, i.e., transforms ones into zeros. The pulse that complements a given bit in B if the corresponding bit in A is 1, produces in B the exclusive OR function of A and B, and is written B ←A(∀). Most other types of signals have fairly obvious names: MQ SH RT shifts the word in MQ to the right (a shift is a jam transfer from one bit to another in the same register); AR COM complements the word in AR. We have been discussing pulse signals so far but the names of gating levels are also quite descriptive: PC+1 INH inhibits incrementing the program counter; FC(E) PSE causes memory control to fetch the word in location E and then pause to wait for a restart for subsequent storage; SAC2 causes the store cycle to deposit a second accumulator. AC0 always refers to accumulator 0, but AC2 refers to the accumulator following the one addressed by the instruction (if the instruction AC address is 17, AC2 refers to accumulator 0). No more than two sources can have the same signal name, and any pair must have opposite

polarities so that the source of every signal is uniquely identified by its name and the associated polarity symbol. Whenever two logically equivalent signals have the same polarity, they are differentiated by adding an extra letter or number to one of the signal names; for example, MR CLR and MR CLR A are equivalent pulses. If one logic signal produces an equivalent signal through a bus driver, the buffered signal is indicated by the letter "B" at the end of the signal name. If the outputs of a flip-flop are buffered externally to the flip-flop module, the buffered signals are indicated by a "B" between the flip-flop name and the state numeral.

The pulses in the time chains for the various main cycles and subroutines all have the same three-part format: first the prefix code naming the chain, then the letter "T," followed by a number or number and letter combination specifying the pulse. For example, FT3 is pulse number 3 in the fetch time chain; DST13 is pulse 13 in the divide subroutine time chain. These pulses are not always in exactly the order that one might expect, and the reader should always consult the flow charts to determine the proper sequence. For example the first three pulses in normalize return are NRT0.5, NRT0, NRT0.1. In the execute cycle, ET0 and ET0A are logically equivalent and together are the first pulse in the execute cycle—the two labels indicate separate but equivalent pulse lines. The next pulse is ET1 but this is followed by ET3. In character operations CHT8B follows CHT8 but precedes CHT8A. But in most cases a letter following the number in a pulse name indicates the next pulse in the chain, usually following return from a subroutine. Thus the first pulse in the floating multiply chain, FMT0, calls the exponent calculate subroutine, and the next pulse, triggered by the subroutine through an SBR, is FMT0A. The SBRs and most control flip-flops that govern the time chains also have similar three-part names in which the "T" is replaced by an "F."

4.3 INSTRUCTION DECODING

Figure 4-2 is a tree which shows the decoding of instructions from the instruction register. The output signals in the figure appear as the gating levels in the flow charts that are described in the following sections. The purpose of the tree is to allow the reader to gain familiarity with these logic gates and to correlate them with the instruction codes. No attempt is made here to give a detailed explanation of every logical function; for this the reader should use the tree in conjunction with the discussion of the decoding hardware in 5.3.

Any code placed in IR is converted into gating levels to govern events that must occur in the fetch, execute, and store cycles and various special sequences to execute the particular instruction. The decoding begins at the left in the figure with the three most significant IR bits, which are converted into signals representing the eight instruction classes. These primary outputs act as gates to enable decoding of the remaining bits. In some cases the output enables a second decoder for several more bits whose outputs in turn act as gates for further decoding. In others the first output represents a single instruction group and it gates the decoding for all remaining bits. For the former type, the line extends only part way across the figure and then generates a number of branches at a single position; for the latter the line extends the length of the tree, and branches appear at several positions. Most of the groups of two, three, or four bits shown together are decoded by binary-to-quarternary or binary-to-octal decoders.

Groups of bits are actually decoded into individual outputs only where signal names are shown for each bit configuration. Where only parenthetical items are listed, the coding as shown produces the mode or action listed, but the hardware does not decode the bit configuration into individual outputs.

When IR receives an instruction code in the instruction cycle, the first three bits are decoded to generate one of eight primary command levels shown at the left in the figure. At the top is the decoder output for a UUO, which corresponds to three zeros in bits 0-2. The decoder output is conditioned by a flip-flop to prevent the generation of the actual command level while IR is clear awaiting an instruction code. Other conditions also generate the command level to perform a UUO when a user program attempts an illegal instruction. At the bottom of the drawing is the decoder output for an IOT, corresponding to three ones. If it is not a user IOT, the command level is generated and causes the decoding of bits 10-12 into the eight IOT instructions. Some of these individual instructions are ORed to generate composite functions. At the same time IR bits 3-9 are placed on the I/O bus to select the device.

Between the top and bottom entries in the figure are the decoder outputs for the six instruction classes that use the basic format (these are not in numerical order, but are instead listed so that classes with common decoding are adjacent). The configuration 001 indicates the floating-point instructions and character operations. If the second octal digit in the code is 0, 1, or 2, there is no further decoding as these 24 codes are not used. If bits 3-5 contain 111, IR6-8 are

decoded for two unused codes, the five character operations, and the single instruction floating scale. A 1 in bit 3 indicates the floating-point instructions and for these, bits 4 and 5 are decoded for the specific instruction, bits 7 and 8 for the mode. The logical condition for rounding (NR ROUND) is dependent upon bit 6, but is not dependent upon any command level because the gate is used only by the normalize return subroutine, which is called only by floating-point instructions.

The second basic instruction class corresponds to the primary command level IR 2XX, which represents a number of small instruction groups. Either 0 or 1 in the second octal digit specifies a full-word transfer for which bits 7 and 8 are decoded for the mode, and specific configurations of bits 5 and 6 generate the levels that control swapping and negating of the word. Second octal digits of 2 and 3 correspond to fixed multiply and divide, respectively. The 01 configuration of bits 3 and 4 (shown as 01-) generates a composite multiply-divide level to enable the net that determines the necessary fetch and store operations from bits 7 and 8. Further fetch and store operations are determined separately for multiply and divide from the state of bit 6, which specifies whether the operands are to be treated as integers or fractions. For division bit 6 is actually decoded into a further pair of control levels, but for multiplication the outputs of IR6 act as gates directly on the multiply sequence. The next three configurations of IR3-5 enable the decoding of IR6-8 for the shift operations, a group of miscellaneous instructions, and the pushdown and jump instructions (the absence of a signal name by any configuration of IR6-8 corresponds to an unused op code). The decoder output for JRST, like IOT, generates the true command level only if it is not executed as a UUO. For the JP group there are two additional control levels corresponding to the six instructions that actually execute a jump, and the three instructions that store the miscellaneous bits. The last group of instructions in IR 2XX are fixed addition and subtraction wherein IR6 determines the instruction, and bits 7 and 8 are decoded for the mode. These modes are equivalent to those for the Boolean instructions, which make up an entire class with the configuration 100 in IR0-2. IR BOOLE also enables the decoding of IR3-6 to determine which of the 16 instructions is specified.

Another single instruction group, the half-word transfers, is specified by 101 in the first three IR bits. For these instructions bit 3 determines which part of the destination register shall receive the half word. Bits 4 and 5 determine the effect on the nontransfer half: a 1 in either bit clears AR, and further decoding enabled by bit 3 determines whether that half shall then be

set to all ones. A 1 in bit 6 causes the operand to be swapped before the transfer so that the destination register receives the half word from the opposite half of the source register. Bits 7 and 8 are decoded for the mode.

The remaining two IR0-2 configurations, 011 and 110, are for the arithmetic compare instructions and the logical compare instructions (ACBM). The former class includes the ACCP group in which an accumulator is compared against either C(E) or E, and the MEMAC group in which either AC or C(E) is compared against zero. For the arithmetic class, 00 in IR3-4 specifies the ACCP type, and a 1 in bit 5 indicates a direct comparison, i.e., against C(E); the other three configurations of bits 3 and 4 specify the types of MEMAC instructions, any of which generates the composite MEMAC command level. The state of IR5 determines whether the comparison tests AC for a jump or memory for a skip. In the ACBM group, bits 3 and 4 determine the action on the masked bits. A 1 in bit 5 specifies that the mask is C(E), and a 1 in bit 8 swaps the mask before the test. The remaining bits in the two classes determine the skip or jump condition as shown; for the logical comparison the condition is determined by bits 6 and 7, for the arithmetic comparison by bits 6-8. The level for ACCP or MEMAC appears in the term for bit 8 because this part of the condition is not used by the ACBM instructions; otherwise no instruction levels need appear because the test level generated is used only by these instructions.

4.4 FLOW CHARTS

The remaining figures in this chapter are flow charts of all operations that can be executed by the arithmetic processor and the four common in-out devices. These figures show every event, and in so far as possible, show the flow of operations in a manner that is equivalent to the actual gating and timing in the hardware (the terminology is from the logic drawings unless italicized). Certain intermediate pulses are shown only implicitly (for example ET1 is shown clearing AR, when actually it triggers AR ←(0), which in turn triggers ARLT ←(0) and ARRT ←(0), which together clear both halves of AR). If an event is prefixed by CFAC, the time pulse is routed via the subroutine interface to the register gating.

Each flow chart is based on a sequence of time pulses shown along a vertical line. Except for insignificant intervals, such as the delay across a pulse amplifier or inverter, time between pulses is shown by breaks in the line. Pulses always appear in ellipses and events in rectangles.

A pair of single horizontal lines breaking a vertical line indicates a delay; between the lines is listed the delay time, or the signal that must be asserted to continue the flow. A break shown by double lines indicates a subroutine call; the upper term identifies the subroutine, the lower term names the pulse returned by the subroutine to restart the calling sequence. A flow line that terminates with an arrow indicates that the flow continues with the pulse listed below the arrow (numbers in parentheses are figure references); a line that terminates in an ellipse indicates that the flow along this path ends with the events associated with the pulse. In a rectangle, a condition written to the left of a colon must be satisfied in order for the event written to the right to occur. A condition written on a line must be satisfied for flow to continue along the line. When several vertical lines branch from a horizontal line, the conditions are written above the vertical lines.

The key cycle, through which initial entry into processor operations must be made, is shown at the left in Figure 4-3. Key functions that require a sequence of events are shown in columns associated with the key time pulses. Some console operations, such as examine and deposit, require only the key cycle; others, such as start and read in, use the key cycle to provide entry to the main sequence. Figure 4-4 and 4-5 show the main sequence. The flow charts for the instruction, address, fetch, and store cycles show all possible events; the execute flow chart shows the main time chain for the cycle, including all exits to special instruction sequences and all operations that are not produced by any instruction, e.g., clearing AR CRY0 at the beginning of the cycle. The basic memory and AR subroutines can be called from the main sequence and are shown, respectively, in Figure 4-3 and at the right in Figure 4-8.

The remaining figures show the execution of instructions following the fetching of the operands. Those instructions that are timed by the execute cycle are shown in tables in Figure 4-6, 4-7, and at the left in 4-8; instructions, other than IOTs, that are executed by special sequences are shown in Figures 4-8 to 4-10. These special sequences may call the basic SC subroutines (Figure 4-8, right) and the special arithmetic subroutines shown in Figures 4-11 and 4-12. Figure 4-13 shows the IOT instructions, some of which require the entire execute cycle as well as the special IOT sequence. In-out operations that take place outside of the processor are shown in Figure 4-14; for each of these an IOT is required at the beginning or the end, but otherwise the sequence goes on independently of processor operations.

For every instruction, Figures 4-6 to 4-10 and 4-13 list the complete decoding, the instruction action, the initial registers, and the initial and final gates. The decoding is shown as performed by the hardware; whenever the decoding does not correspond exactly to the meanings of the bits in the instruction codes, further explanation is given in italics. The action is given in the programming sense (i.e., in terms of accumulators, memory locations, control registers, flags), but in the notation of the logic drawings: AC may mean either an accumulator or its contents, E may mean either an address or the addressed location; the meaning is evident from the context. The states of the processor registers are shown following the fetch cycle; MA always contains the effective address and is not shown. The initial gates are those that control the fetch and execute cycles. It should be understood that PC is incremented at ET0 whenever the PC inhibit is not shown as an initial gate, unless there is some special situation (e.g., the processor is in a priority interrupt cycle or the instruction is being executed from the console). For all instructions that are executed by the execute cycle, or return to it, the final gates (i.e., those that control the store cycle) are listed at the end of the sequence. In all cases, registers and gates are listed in terms of the instruction modes even though the modes for fixed multiply and divide are not decoded individually.

Since the complete timing for the execute cycle is defined in Figure 4-5, those instructions whose events are timed entirely by the execute pulses are shown in tables. In a table entry for ET3, double vertical lines indicate the call of an AR subroutine; the subroutine is at the left, the return pulse at the right. The pulses from ET6 to ET9 are omitted when the instruction does not use the second half of the cycle.

4.5 EXECUTE CYCLE FLOW

Every main cycle, subroutine, and special instruction sequence is explained with the description of the logic that generates the time chain and gating levels for it. However, for instructions that are performed by the execute time pulses it is impossible to consider the multitude of possible events in the description of the execute cycle. For most such instructions the reader can easily determine how the specific sequence of events produces the desired result by inspecting the tables in Figures 4-6, 4-7, and 4-8. A few of the less obvious sequences are, however, described here (in the following it is assumed that the reader is familiar with the action of each instruction).

In a half-word transfer (Figure 4-6, left) the positions of the source and destination words after the fetch cycle depend upon the mode; i.e., the fetch cycle always places an accumulator in AR and a word from memory in MB, and the mode specifies which of these is the source and which the destination. The execution of the instruction requires that the source word be in MB and the destination in AR, so ET0 switches MB and AR for the memory mode and transfers MB to AR for the self mode. If the transfer is to be from one half of the source to the opposite half of the destination, then ET1 swaps the two halves of the source in MB. If the instruction is to perform any operation on the unused half of the destination, ET1 also clears AR. Then ET4 jams the specified half word into the appropriate half of AR and completes the action on the other half by setting it, if required. At ET10 the result is transferred from AR to MB for either the memory or self mode, i.e., those modes that require the store cycle to deposit the result in location E. In most instructions the result is in AR at the end of the execute cycle and thus any mode requiring storage in E usually includes the AR to MB transfer at ET10. Exceptions include instructions such as EXCH in which the correct words are already in MB and AR at ET0.

In a full-word transfer, ET0 performs whatever transfer is necessary so that the word to be moved is in both MB and AR. Then if the halves are to be swapped, the swap is made by ET1 in MB; if the word is to be negated—which may happen for either MOVN or MOVM—the AR negate subroutine is called at ET3. By ET4 the result is in MB only for MOVS, so for this instruction MB and AR are switched. Of course only a transfer from MB to AR would be necessary, but the gate for the bidirectional transfer is required for other instructions (no instruction requires a transfer limited to one direction). The other instructions in Figure 4-6 are quite straightforward with the possible exception of the logical compare group, ACBM. At the end of the fetch cycle, the word to be used as the mask is in MB, and the word to be tested in AR. For convenience let us refer to the latter as the data word, and the AND function of it with the mask as the test word. ET0 completes the construction of the mask by swapping the two halves of MB, if this is necessary. ET1 then produces the test word by transferring zeros from AR to MB, and at the same time it adjusts the data word for the complement and set actions by complementing bits of AR corresponding to ones in MB (the exclusive OR function) or transferring ones from MB to AR. ET4 then moves the test word to AR and the data word to MB. At this point the clear action has not been handled and it would seem that the mask has been lost. However, any ones now in AR must be a subset of the ones in the original mask, and these correspond exactly to all the ones

within the masked bit positions in the data word in MB. For the clear action the next three time pulses clear these bits by complementing AR, transferring zeros to MB, and complementing again to restore the test word. At ET9 the test is made according to the contents of AR, and the adjusted data word is transferred into AR at the same time so that it is available for subsequent storage in AC. The bidirectional transfer listed in the table is not necessary, but is available because of the requirements of other instructions. The store cycle deposits the data word in AC unless it has not been changed from the original.

In JRST (Figure 4-7), the second half of the execute cycle performs a series of operations that seems to be pointless, since the store cycle does not deposit anything. Pulses 5 and 6 transfer the current program address from PC to MB, and after the jump address has been transferred to PC, pulses 9 and 10 transfer MB to MA. If the program does continue, these operations are pointless, but the JRST can halt the processor and may be used for error halts in maintenance programs. When the processor stops, PC displays the jump address and MA displays an address one greater than that of the location that contained the JRST.

CHAPTER 5

MAIN SEQUENCE CONTROL

This chapter describes the processor response to console inputs and explains the way in which the main control elements operate on instructions. Of the five major cycles, four are described in detail here: retrieval of an instruction from memory and decoding it from the instruction register; calculation from memory and decoding it from the instruction register; calculation of the effective address; retrieval of the operands; and storage of the result. The execute cycle is discussed only to the extent of basic operations, and timing events required for the execution of individual instructions are described in other chapters. The many arithmetic and data subroutines are described in Chapter 6 with the arithmetic registers; Chapter 7 describes the memory subroutine and memory address and data transfers; Chapter 8 includes the decoding of IOT instructions and control over transfers between the processor and input-output equipment. The various test conditions that affect the program sequence are discussed here, but any arithmetic or logical operations necessary to produce them are described in Chapter 6.

5.1 CONSOLE CONTROL

In general, the console switches provide continuous control levels that gate certain operations within the processor; whereas the keys are momentary contact switches that allow the operator to trigger specific sequences of events or to stop the processor. The hardware that provides control and timing for the various console operations is shown in Figures 5-1 and 5-2 (flow chart, Figure 4-3).

When the operator turns on computer power at the console, the 4303 Integrating Delay in the lower right of Figure 5-1 comes on in the 1 state, turning on a clock that provides a string of master power clear pulses. These pulses clear RUN so the processor cannot begin operations inadvertently, and they also trigger the master clear and the master start (Figure 5-2, right). MR START clears the flags and the in-out system including all flip-flops in the priority interrupt system and the processor I/O interface; MR CLR clears the major control registers and flip-flops

in the processor proper including all SBR flip-flops. The master clear also occurs at the beginning of every new operating sequence, i.e., at the beginning of every console operation other than memory continue (see below) and at the initiation of any new instruction (IT0 or any equivalent pulse). The master start is generated only by the power clear and when the operator specifically wishes to clear the I/O system or place the processor in the executive mode. The flags and several special control flip-flops are cleared only by MR START because they must carry information over from one main sequence to another.

The left half of Figure 5-1 shows the logic inputs from the keys and switches, and nets that generate a number of composite functions to control events common to two or more console operations. Several of the nets AND key functions with a state of the RUN flip-flop. For example, a net in A3 allows the input from the EXECUTE key to affect the processor only if it is not in operation. Since examine and deposit may be performed while the processor is running, these two operations are governed not by levels from the keys but rather by two pairs of sync and start flip-flops (upper right). A pulse in the key time chain, KT0A, normally clears all four flip-flops, but for examine or deposit the appropriate sync flip-flop is set instead. Then, if the processor is running, the corresponding start flip-flop is set at the beginning of the execute cycle. The sync flip-flops provide gates for the events in these operations; the start flip-flops control the insertion of the operation between two main sequences. For this purpose, two of the nets at the left generate the AND functions of RUN (1) with functions of the start flip-flop states. If at the end of a main sequence, either flip-flop has been set and the processor is to continue, the signal from the net in C5 causes the processor to enter the key cycle at the end of the main sequence. However, if neither flip-flop has been set, the signal from the net in C3 causes the processor to begin a new main sequence.

Pushing any key other than the two STOP keys generates the level KEY MANUAL through the net in the top center of the figure (the entire net is disabled during the power clear period). The turnon of this level triggers a pulse generator in the upper left of Figure 5-2 to start the key time chain. This chain is broken between KT0A and KT1 by the condition RUN (0) so that KT0A triggers the following triplet of pulses only if the processor is not running. There are, however, two gates that bypass the RUN condition: if the operator selects EXAMINE or DEPOSIT while the processor is running, the appropriate start flip-flop allows the final pulse in a store cycle to trigger KT1; furthermore memory continue goes directly from KT0A to KT1

because it is assumed that the processor is "running"—i.e., RUN is 1—but that the operator has previously stopped it at the end of a memory cycle by inhibiting the memory subroutine return. This is why memory continue is the only operation in which KT1 does not generate the master clear: somewhere an SBR is legitimately waiting for the subroutine completion. Memory continue triggers the MC restart at KT1.

For read in, KT1 sets the readin mode subroutine flip-flop shown at the lower left. When the processor is operating in readin mode, the normally unused core registers at the bottom of memory replace the 16 flip-flop registers of the fast memory. In all other respects, read in is exactly equivalent to start; thus the operator may keep a loading program stored permanently in the readin area of memory, and the processor switches to normal operating mode whenever an instruction is taken from any location beyond 17 ($\sim$ MA18-31 = 0). For console operations other than read in, KT1 clears the flip-flop for normal operation, but the clear is inhibited by the CONTINUE keys so that READ IN can be single stepped (the clear net is in Figure 51D1).

A few of the functions produced by the key time pulses are shown on drawings associated with the appropriate function (such as operations involving PC) but most are shown in the center portion of Figure 5-2. Start, read in, examine, and deposit all clear MA and transfer an address into it from the console ADDRESS switches. Examine next and deposit next, instead of loading MA, increment it. The three operations that send data into the computer, deposit, deposit next, and execute, clear AR and load it from the DATA switches. Also shown in the key drawing is the gate through which the DATAI for the processor loads the DATA switches into AR. CPA indicates that the IOT has selected the processor and the assertion of the DATAI level triggers the transfer PA (see 8.3).

Following the various setup operations, start, read in, and instruction continue all generate KEY GO (upper right). This pulse places the processor in operation by setting RUN (Figure 5-2) and starting the instruction cycle. As long as RUN remains set, the completion of every store cycle triggers a new instruction cycle so that the processor performs one instruction after another in the program. Either the operator or the program may stop the computer at the end of a store cycle by clearing RUN at ET0. The executive program clears it by a 1 in bit 10 of a JRST; a user program cannot stop the processor. The operator clears it by pressing the INSTRUCTION STOP key, but the logic gate is bypassed by a pulse generator that is also connected

to the key. The action through the PG is slow compared to the processor time chains, but this ensures that RUN will be cleared even if some malfunction should interrupt the normal procedure.

The other console functions do not place the processor in normal operation. Execute transfers AR to MB at KT3 and enters the instruction cycle at the point at which an instruction would normally be in MB after retrieval from memory but it does not set RUN, so the processor stops at the completion of this single instruction. In the four examine and deposit operations, KT3 requests a memory read or write as required and sets KEY RE/WR (Figure 5-2 B6, C1). To deposit information in memory, KEY WR also transfers AR to MB. Upon completion of the memory subroutine, the MC restart generates the read/write return clearing the SBR.

The upper right portion of Figure 5-2 shows the gates that generate KT4 for the repeat mode. If the REPEAT switch is on, KT4 triggers a repeat delay (upper left): the 4303 state change at the end of the delay period then retriggers the key time chain as long as any initiating key is held on. To use the memory repeat function, the processor must already be running when the operator presses the STOP key for memory; then KT4 is generated every time the memory stop flip-flop is set. To repeat start, instruction continue, or read in, the operator must hold on both the initiating key and the INSTRUCTION STOP key; then at the completion of each instruction, ST7 triggers KT4. Since execute does not start the processor, the operator need not use the STOP key to repeat it; in this case, KT3 triggers KT4. Similarly, for the four examine and deposit operations, the read/write return triggers KT4. Furthermore, examine or deposit may be repeated while the processor is running: if one of these operations interrupts the normal transition from store cycle to instruction cycle, the appropriate start flip-flop causes KT4 to trigger KEY GO. This pulse does not affect RUN, which is already 1, but it does retrigger the instruction cycle and clear the start flip-flops.

5.2 PROCESSOR CYCLES

There are five cycles in the main sequence: instruction, address, fetch, execute, and store. All but the execute cycle perform only operations that are common to all instructions or to groups of instructions, and these four are described in detail in this section. Instruction, fetch, and store can call only the memory subroutine; whereas the address cycle can call both the memory and the add subroutines.

The execute cycle performs some general control operations but is limited primarily to the specific operations necessary for the execution of individual instructions, including entry into special instruction sequences. For this reason, the description of the execute cycle included here is limited to timing, general control functions, the calling of the simple AR subroutines, and the entry and return for special sequences.

In the fetch, execute, and store cycles, the gating levels that govern the sequence of time pulses and the calling of subroutines are OR functions of instruction conditions. An input condition may be an entire instruction class or a single instruction, or it may be a single mode either within an instruction or in an instruction class. The reader is assumed familiar with the standard outputs from the instruction decoders (4.3) and only special conditions are discussed here.

a Instruction

The lower half of Figure 5-3 shows the time chain that controls the retrieval of instructions from memory (flow chart, Figure 4-4). When the operator starts the processor or causes examine or deposit to interrupt normal processor operations, KEY GO begins a new instruction cycle by triggering IT0. If the processor is in normal operation and no examine or deposit has been synced by key timing, the final pulse in the store cycle of an instruction starts the instruction cycle of the next one. IT0 generates the master clear, clears the memory address register, and after a slight delay sets IF1A, the SBR for the memory subroutine that will subsequently retrieve an instruction. IT0 also generates PI SYNC (8.2b) which strobes the priority interrupt system provided the processor is not already in a PI cycle. If the PIR strobe produces no request or if the processor is already in a PI cycle, the sequence continues to IT1 directly. If the strobe does produce a request, the sequence continues instead to IAT0, which triggers the master clear, places the processor in a PI cycle by setting PI CYC, and then triggers IT1, which calls the memory subroutine and supplies a memory address for instruction retrieval.

In a normal sequence, i.e., PI CYC(0), the address comes from the program counter; if the processor is in a PI cycle, however, the appropriate PI channel address is transferred into MA. Furthermore, if PI OV is 1 (this can occur only in the second consecutive PI cycle when a block IOT is completed), MA is incremented by 1 at the same time that the channel address is

transferred in, so the processor performs the instruction in the second location associated with the channel. When the instruction is available in MB, the memory restart triggers IT1A, which transfers the instruction code and AC address from MB into IR. If the instruction was retrieved from a location above 17, IT1A also clears the readin mode flip-flop in case the processor has been in read in.

The instruction cycle may also be entered late for several situations. If a UUO appears in the program, UUO T2 starts a new instruction execution by setting IF1A and making a read request to retrieve the instruction in location 41; the memory restart then continues the chain automatically by triggering IT1A. After retrieval of the operand in the fetch cycle of the execute instruction, XCT T0 returns to IT1A so that the operand is executed as an instruction. Similarly, the console operation execute triggers IT1A after the contents of the DATA switches have been transferred to MB via AR.

b Address

The calculation of the effective address for an instruction is governed by the logic shown in the upper half of Figure 5-3 (flow chart, Figure 4-4). In a normal sequence, IT1A starts the address cycle by triggering AT0. However, the cycle is also triggered by CHT9 for the second part in a character operation: in this case, the processor has already handled the pointer and must now calculate the effective address for the operand as specified by the pointer. AT0 transfers the address portion of the instruction word from MB to AR, transfers the indirect bit and the index register address to IR, and clears MA in preparation for subsequent memory access.

AT0 also generates PI SYNC (8.2b) which strobes the priority interrupt system provided the processor is not already in a PI cycle. If the strobe discovers any request, PI SYNC generates IAT0 which returns the processor to the instruction cycle. If there is no request or if the processor is already in a PI cycle, the sequence continues the address cycle by triggering AT1 provided the operator has not pressed the INSTRUCTION STOP key within the preceding 100 μsec (IF1A being 0 guarantees that the return is to the address cycle rather than the instruction cycle.) If the instruction specifies no index register (IR14-17 = 0), AT1 jumps directly to AT4. However, if address modification is to be performed, AT1 triggers AT2, which transfers the index register address to MA, makes a memory read request, and sets A LONG. The return from

memory triggers AT3, which sets AF3A and requests the add subroutine to add the contents of the index register to the address specified by the instruction. The subroutine return transfers the calculated address from AR back to MB and moves the cycle on to AT4, which clears the left half of AR. If the instruction specifies an indirect address (IR13 is 1), the cycle continues to AT5 which transfers the calculated address from MB back to MA and makes another read request to retrieve the new address. AT5 also sets A LONG, clears the indirect bit and index register portions of IR, and sets AF0. The last action allows the memory restart to trigger AT0, restarting the address cycle.

If the instruction specifies a direct address, the processor goes on from AT4 to the fetch cycle. At the completion of address calculation, MA, MQ, and ARLT are all clear; the effective address calculated for the instruction is contained in the right halves of both MB and AR. The contents of MBLT are equal to the last word retrieved from memory, provided the final address cycle did not index.

c Fetch

Figure 5-4 shows the logic that controls the retrieval from memory of the operands necessary for the execution of an instruction (flow chart, Figure 4-4). The lower part of the figure shows the generation of the control levels for the cycle. In all instruction groups, individual instructions or specific instruction modes which do not require an accumulator, the net in the lower left generates an inhibit for that function. The bottom two inputs, CH INC OP and CH ∿ INC OP, both apply to the first part of character operations and indicate whether or not the pointer shall be incremented after it is retrieved. The level IR 254-7 is asserted for the four included instruction codes: the first two are jumps; the third is XCT; the fourth is not used. All other inputs that generate FAC INH are standard instruction situations (5.3). There are a few instructions which, after retrieving the accumulator, require the retrieval either of a second accumulator or of a word addressed by half of the already retrieved accumulator. The three levels that govern these situations are shown in the center of the figure. The cycle fetches AC2 for any double word shift operation and for fixed point division, which uses a double-length dividend. POP and POPJ both pull out the word addressed by ACRT, which keeps the final address for the pushdown list. Fetching of the word addressed by ACLT

is required by JRA and BLT: the former to restore AC as the return from a JSA which stored AC in E and saved E and PC in AC; the latter to retrieve a word for subsequent storage in the location addressed by ACRT.

Two levels govern the retrieval of a word according to the effective address, FC(E) and FC(E) PSE. The first reads and releases the memory so it can rewrite automatically. The second makes a read/write request: the memory subroutine thus pauses after fetching so that information can later be written into the same memory location (usually in the store cycle and controlled by the same level). For the first part of character operations, FC(E) retrieves the pointer if it is not to be incremented; whereas the pause is required if an incremented pointer is to be written back in. Similarly, in the second part, a character deposit requires the pause while a load does not. An IOT BLK also retrieves a pointer which is automatically indexed and thus requires the pause. The remaining inputs to both nets are all standard instruction situations, but the reader should take special note of two inputs to FC(E): IR MD FC(E) and IR FP. The first represents all fixed-point multiplication and division operations that do not use E as an operand; the second includes all floating instructions except floating scale. All of these instructions have modes that store the result in E; however, all require significantly greater execution time than other instructions and they request read to release the memory.

In the upper half of the figure are the delays and SBR flip-flops that provide the fetch time chain. The normal entry is from the address cycle at AT4 when no further deferring is required. If the address calculation requires no indexing or deferring at all, A LONG is 0 and the entry into FT0 is delayed; otherwise the entry is immediate. There is also an entry into FT0 for a block IOT: after the pointer has been indexed and stored, IOT T0A triggers the fetch cycle for the data instruction that follows (8.1). FT0 performs no outside operations but continues the proper fetch sequence. If the instruction requires no accumulator, the sequence goes directly to FT5. To fetch AC, the sequence goes to FT1, which transfers the AC address from IR to MA and requests a memory read. The memory restart triggers FT1A, which is also produced directly by BLT T6. In a block transfer, each word is retrieved by a fetch cycle; but the store and index operations are performed by the special BLT subroutine. The subroutine returns to FT1A bypassing FT1, because the instruction must fetch AC, which contains the initial source and destination addresses, only for the first transfer.

The sequence from FT1A and the operation performed by it depend upon whether additional AC operations are necessary. All sequences include a switch between MB and AR so that AC is in AR and E in MB. If there are no extra AC fetch operations, FT1A clears MA and switches MB and AR. To fetch a word addressed by AC, FT1A clears MA; and if the address is in the left half, swaps the halves in MB. It also triggers FT3, which loads MA with the selected half of AC, makes the MB-AR switch, and goes on to FT4. If a second accumulator is required, FT1A adds 1 to MA, switches MB and AR, and skips to FT4 which triggers the memory read for either type of extra AC fetch. FT4 also sends MB to MQ to save E. The memory subroutine returns to FT4A, which clears MA, and switches MB and MQ so that MQ contains the extra word fetched and MB again contains E. FT4A also triggers FT5, which results directly from FT1A if no additional AC operations are required. FT5 transfers E from MB to MA and then triggers FT6 or FT7 depending upon whether the instruction requires a memory read or read/write. The memory return then completes the cycle by triggering FT6A, which results directly from FT5 if the effective address is not used for memory access.

At the completion of the cycle, MA contains E; and AR and MB contain AC and C(E) if both words were fetched, although the AC halves are swapped if the cycle also fetched the word addressed by the left half. If there was no AC fetch, AC contains E with the left half clear; and if there was no C(E) fetch, MB contains E with the left either clear or in the same state as at the end of the address cycle (the latter case occurs only if no fetch operations were performed at all). MQ is clear unless an additional AC operation was required, in which case it contains the extra word.

d Execute

After fetching the operands, the processor performs the operations necessary for the execution of the instruction. Figure 5-5 shows the logic for the execute time chain, the flow chart is Figure 4-5. Only the simpler instructions are actually executed by the ET pulses; for the more complicated ones, the pulses trigger appropriate subroutines.

The lower portion of the figure shows the generation of three levels that control the execute sequence. Two are inhibit levels that break the chain for subroutines; reentry into the main sequence is made from the subroutine usually—but not always, at ET10 (XCT and UUO, for

example, both return directly to the instruction cycle). The subroutines for all IOTs and for FSB start at ET4 and thus inhibit ET5. All other instructions requiring execution by subroutine inhibit ET4 although this does not mean that the subroutine entries are made at ET3. In fact, most of them are at ET0 but the execute chain continues to ET3. There is also a group of instructions (including fixed-point add and subtract and some of the data transmission and executive instructions) whose transfer and logical operations are triggered by the ET pulses but which require an AR subroutine. These instructions all generate the level AR SBR which inhibits ET4 but only for a pause: AR SBR causes ET3 to trigger the SBR at the left at the same time that it triggers an appropriate subroutine. The subroutine return then triggers ET4 to continue the chain. AR SBR includes several instructions which generate both inhibits: floating subtract requires the negate subroutine at ET3 and enters the floating add subroutine at ET3 and enters the floating add subroutine at ET4; both block IOTs index the pointer at ET3 and switch to an IOT subroutine.

Almost all logical and transfer operations triggered by ET pulses occur in the first half of the chain; most instructions using ET6 to ET9 affect the program counter. These instructions generate E LONG (right) which causes ET5 to go to ET6 instead of skipping to ET10.

In the execute cycle flow chart, the only events for which complete conditions are listed are general operations independent of specific instructions and operations involving PC, MA, and the flags. The many logical and transfer operations on AR and MB that actually execute the Boolean, data transmission, and other instructions occur almost exclusively at pulses 0, 1, and 4; to determine which events are required for a specific instruction, refer to the appropriate instruction flow chart. The following description of the execute cycle discusses only the sequencing of the cycle, entry to and return from subroutines, and the more general operations. For events involving individual registers, refer to the appropriate section of the logic in this and following chapters.

To begin the execute cycle, the final pulse from the fetch cycle FT6A, triggers ET0 and ET0A simultaneously and also triggers a delay for a subsequent ET1; ET3 follows automatically from ET1. The two 0 pulses are logically equivalent but two pulse lines are required because of the many events they trigger. Usually ET0 increments the program counter so that the next instruction will be taken from the next memory location in sequence. The only circumstances

that inhibit PC+1 are those in which ET0 will occur again before the next instruction in normal sequence is to be performed. ET0 also clears the AR carry flip-flops for use during the cycle, synchronizes those console operations which may interrupt the normal sequence from store cycle to instruction cycle (5.1), and clears RUN for the halt instruction (JRST with a 1 in bit 10). The first three execute pulses also handle hold and dismiss operations for the priority interrupt system (8.2b), restoration of flags, and entry into user mode. ET0 provides subroutine entry for character operations, both fixed and floating multiplication and division, and floating addition; ET3 starts the subroutines for BLT, FSC, UUO, and all shift operations. XCT also stops the chain at ET3 in order to return to the instruction cycle.

ET4 follows immediately from ET3 only if there is no inhibit. For the instructions that generate AR SBR, ET3 triggers the appropriate AR subroutine and sets ET4 AR PSE. The subroutine return, ART3, clears this SBR and triggers ET4, which in turn continues to ET5 unless there is an entry to the FSB subroutine. If ET4 follows from ET3 without pause, ET5 follows immediately except for the in-out transfer instructions: for these ET4 triggers the IOT subroutine, but at its completion time pulse IOT T3 returns to the execute cycle by triggering ET5.

Most computational and data transmission instructions skip the second half of the execute cycle by having ET5 go directly to ET10. However, if E LONG is asserted, the chain continues through all the remaining pulses without interruption. ET7-9 perform the necessary program control operations (5.4); the final two pulses in the cycle regulate the flags (6.2e, 8.3). Since most instruction results that are to be deposited by the store cycle are produced in AR, most instructions switch AR and MB either at ET9 or ET10. ET10 sets up MB and triggers the store cycle not only for most instructions executed in the execute cycle but also for many of those executed by subroutine. The return for fixed-point division is at ET9 (provided the division could be performed); return is made at ET10 for character operations, all floating-point instructions (all of these except floating scale are made from the normalize return subroutine), fixed-point multiplication (which also returns via NRT6), and the block transfer routine if the block is complete.

e Store

Information resulting from the execution of an instruction is deposited in memory by the store cycle (logic, Figure 5-6; flow chart, Figure 4-5). The sequencing of the store time chain is controlled by four levels: FC(E)PSE from the fetch cycle, which gates in the write restart for any instruction that used a fetch and pause before execution; and three levels that gate in write requests and are generated by the store logic. SC(E) is generated by three types of instructions: those which merely clear location E and deposit information in it without having required a fetch; those which store C(E) in a location other than E; and those which require so much execution time that they requested only the read rather than the read and pause in the fetch cycle to free the memory during execution. SAC INH is generated by instructions that address no accumulator, by computational and data transmission instructions in the memory mode (that mode which stores the result only in memory), or by test instructions that have no result to store. Instructions that compare a memory word against zero store the word (which may be indexed) in both memory and an accumulator, but AC storage is inhibited if AC0 is addressed (the net in C6 decodes IR9-12 for no AC selection). The input BLT LAST inhibits AC storage in the cycle following completion of a block transfer. If a priority interrupt stops the block before completion, the current addresses are stored in AC so that the block may be continued after the break. Some arithmetic and shift instructions generate SAC2 to store MQ in a second accumulator. This event occurs for all double-length shifts, floating-point instructions that store the low-order halt, and fixed-point operations that have a double-length result.

To generate a write request for SC(E), ET10 triggers ST1; whereas, it triggers ST2 to provide a read/write restart for SC(E) PSE. Either time pulse sets SF3, and at the completion of the memory subroutine the return triggers ST3. If there is no storage according to E (or whatever address has replaced E during the execute cycle), ET10 triggers ST3 directly. If there is also to be storage in an accumulator, ST3 clears MA and ST5 loads the AC address into it from IR9-12, transfers AR to MB, and makes the write request. The memory restart triggers ST5A which goes on to ST6 if a second AC storage is necessary. In this case, ST6 increments MA to address the second AC, transfers MQ to MB, and calls another write. ST6 is also delayed slightly to generate ST6A which sets SF7, the SBR for the memory subroutine call. This delay is necessary to guarantee that the return from the previous subroutine cannot trigger both

ST5A and ST7. The return from this storage then triggers ST7 which also follows directly from ST3 if there is no AC storage at all, or from ST5A if there is no second AC storage. ST7 is the return time for the divide subroutine if the division could not be performed, and for a character operation that terminates after the first part (i.e., an instruction which operates not on a character but only on the pointer). Since all FP/CH codes inhibit ET4, those that are not used for instructions cause ET3 to trigger ST7; thus all unused codes (except the UUOs) are interpreted as no-ops. Furthermore, if at any time a user program attempts to address a protected area of memory, a priority interrupt is requested on the CPA channel and the sequence jumps directly to ST7. If RUN is 0, the processor stops at this point. Otherwise, ST7 returns to a new instruction cycle unless the operator has requested examine or deposit, in which case, it returns to the key cycle.

5.3 INSTRUCTION CONTROL

For each instruction to be performed, the 18-bit instruction register receives the instruction code, indirect bit, and AC and index register addresses from MB (Figure 5-7). Although information may be transferred into IR only from MB, the IOT time pulse T0A directly sets IR12, which changes a block IOT instruction into the corresponding data instruction after the pointer has been indexed. For effective address calculation, the indirect bit IR13 controls the repetition of the address cycle, and IR14-17 provides the address of an index register to be used in the calculation. Although there are 16 accumulators, only 15 of them can be used as index registers. A 0 address in IR14-17 indicates no index register selection and the address cycle performs no indexing. The generation of the appropriate address cycle gate, IR14-17=0, is shown in the lower right of Figure 5-8. For the fetch and store cycles, the AC address is supplied by IR9-12. For IOT instructions, IR0-2 are all ones, and IOT control decodes the instruction specified by IR10-12. A device is selected according to the code supplied by IR3-9; the bus drivers for the in-out selection lines are at the top of Figure 5-7.

The lower portion of the figure shows the pulse signals that control IR. MR CLR clears the entire register at the beginning of every main sequence. In every instruction cycle, IT1A transfers the instruction code and AC address (or instruction and device codes for an IOT) into IR0-12 from MB; at the beginning of every address cycle, AT0 transfers an indirect bit and index register address into IR13-17 from MB. If the calculated address is indirect, AT5 clears

IR14-17 in preparation for reloading at AT0 when the cycle repeats. When the first part of a character operation is completed, the processor returns to the address cycle to calculate the effective address for the character from the pointer. Thus CHT8A clears IR13-17 in preparation for loading at AT0, which is triggered by the final pulse in the first part of the character time chain.

The next three logic drawings, Figures 5-8 to 5-10 show the main decoding of IR0-8 to control the execution of individual instructions. In this section of the processor, the decoding is carried down in many cases to individual instructions and instruction modes. There are, however, a few command lines which represent instruction pairs whose differences are minor; trivial additional decoding for these is shown with the hardware that actually executes the instruction. For example, the IR multiply and divide outputs represent both integral and fractional operations; the floating command outputs represent both the standard instructions and those that round. Figure 5-8 shows the primary decoding into major classes, decoding into command levels for fixed-point multiply and divide and for all floating-point instructions, and final decoding for all single instructions without modes that correspond to unique 9-bit codes. All of the Figure 5-8 outputs, although representing functions at several different levels in the decoding hierarchy, have the prefix IR. Outputs from the groups decoded in Figures 5-9 and 5-10 have the appropriate group prefix. Figure 5-10 also shows the timing for SCT and UUO.

The binary-to-octal decoder in the upper left of Figure 5-8 performs the first stage in instruction decoding by determining which of the eight primary instruction classes is specified by IR0-2. The decoder gating input P is grounded so that the decoder is always on. All codes in the UUO class produce exactly the same operations and require no further decoding. In the IOT class, the eight instructions specified by IR10-12 are decoded in IOT control. Outputs 1 and 2, IR FP/CH and IR 2XX, gate other decoders shown in this figure; further control in the classes represented by outputs 3, 4, 5, and 6 is shown in Figures 5-9 and 5-10. The primary command level IR FP/CH includes the instructions for floating-point arithmetic and character operations, and instruction codes 100 to 131 which are not used. IR FP/CH ANDed with the condition 011 in IR3-5 (i.e., codes of the form 13X) gates the lower left decoder, which produces the command levels for FSC and the five character operations. At the right, the

primary level gated by IR3(1) represents all floating-point instructions other than FSC: bits 4 and 5 are further decoded to determine the instruction, bits 7 and 8 to determine the mode. Bit 6 does not appear here but controls rounding at the normalize return subroutine.

The top center decoder is gated by IR 2XX to convert IR3-5 into eight command levels, each representing eight instructions or two instructions with four modes. The 0 and 1 outputs each represent two of the move instructions and are ORed to generate the command level for the group. At the left, IR SH is further decoded into the six shift instructions plus two unused codes (the two types of arithmetic shift are ORed in B4 for use by the overflow logic); at the upper right IR 25X is decoded into seven miscellaneous instructions plus one unused code. Just below, a gate generates the signal IR 254-7 to inhibit AC fetch and storage, which is not required for these codes.

Although the decoder gated by IR 2XX generates two outputs for fixed-point multiply and divide, the condition representing both of these outputs also generates a combined command level IR MD (Figure 5-8, center). Further decoding produces the appropriate fetch and store gates. All modes but the immediate (01 in bits 7 and 8) fetch C(E). Only fractional division uses a double-length dividend and fetches AC2. The two modes with a 1 in bit 7 store the result in memory; whereas only the memory mode inhibits storage in AC. For all modes that do store in AC, a second AC storage is required for the remainder in division and for the low-order half of the double-length product in fractional multiplication.

Note that three of the decoder outputs at the top of the figure have a suffix "A:" these are IR UUO A and IR IOT A in the upper left, IR JRST A in the upper right. These decoder outputs do not drive the command lines for the corresponding instructions. Instead, they are applied to the executive logic (5.5) to determine in the latter two cases whether the instruction is allowed or must be executed as a UUO. The command level for UUO is generated by the executive logic; the command lines for the other two instructions are driven by the gates in C4, each of which is enabled by the appropriate decoder output when the instruction is not being executed as a UUO. The gate just below ORs the 1 states of IR9 and IR10 for use by the executive logic in testing a JRST.

ACCP V MEMAC, ACBM (Figure 5-9)

For the ACBM group (upper right), bits 3 and 4 are decoded for the action on the masked bits: do nothing clear, complement, set. IR5(1) selects C(E) for the mask direct, as against immediate which uses E. A 1 in bit 8 swaps the AC halves before masking. The net in the upper left decodes bits 3 and 4 for instructions in the class that includes both ACCP and MEMAC: one fourth of the instructions do an arithmetic comparison of AC against either E or C(E); the other three types compare C(E) or AC against 0 and either merely test, or add or subtract 1 before testing. Any instruction in the latter three types generates the main control level MEMAC; the state of bit 5 determines whether the test word shall come from memory or an accumulator. In ACCP, a 1 in bit 5 specifies a comparison with memory (direct); otherwise the test is made against E (immediate).

The logic nets at top center in the figure test the skip or jump condition for all three groups, ACCP, MEMAC, and ACBM. The condition is determined by the pair of gates at the left: bit 7 selects the condition that AR is 0; bit 8 that the test word is less than the standard, be it 0, C(E), or E. The upper gate functions only for ACCP and MEMAC: the logical comparison employs an AND function and the only test is whether or not the masked bits are all zeros. In an arithmetic comparison, the function AR = 0 represents equality of the test word and the standard. The function representing the inequality is the exclusive OR of the AR sign bit and the overflow condition. This function is true when the subtrahend is greater than the minuend; but since the standard is subtracted from the test word, it is true when the test word is smaller. In MEMAC, the overflow condition is automatically false (for a complete description, see the flag logic, 6.2e) so the entire function is true when AR0 is 1, i.e., when the test word is less than 0.

FWT (Figure 5-9)

For a full-word transfer, the four standard modes are decoded from bits 7 and 8; a left-right swap is made for MOVS (bits 5, 6 = 01); and the word is negated either when moving the negative or when moving the magnitude while the word is negative, both situations being represented by the condition IR6(0) V AR0(1) in MOVN or MOVM.

HWT (Figure 5-9)

For a half-word transfer, the standard modes are decoded from bits 7 and 8; a 0 or 1 in bit 3 specifies whether the half word shall be transferred into the left or right half of the destination; and a 1 in bit 6 specifies that the source word shall be swapped before the transfer is made so that the left half of the source is transferred into the right half of the destination or the right into left. The other levels control the operation on the other half of the destination. If bits 4 and 5 are both 0, there is no action. If either bit is 1, AR is cleared; the word is constructed by loading a half word into one half and complementing the other half if it should be all ones. If bit 4 is 0, the other half is left clear; however, on IR4(1) the other half is set to all ones (i.e., complemented) if the instruction requires that it be set or that its bits be made equal to the sign of the transferred half and that half is negative. The gates to the left generate the appropriate functions. For a transfer to the right, the upper net generates HWT LT SET if instruction bit 5 is 0 or the sign bit MB18 of the right half word is 1. The lower gate performs an equivalent function for the transfer left.

BOOLE, AS (Figure 5-10)

Since BOOLE and AS have the same modes, the OR function of the two instruction levels gates a decoder for bits 7 and 8. For further decoding of AS, the command level is merely gated by the states of bit 6 to determine whether addition or subtraction is required. To decode for the 16 Boolean operations, bits 4, 5, and 6 are applied to binary-to-octal decoders, one gated by the 0 state of bit 3, the other by the 1 state. The table at the left lists the 16 operations by name, function, and number, shows the result for each of the four possible pairs of operand bits, and lists the basic functions which, taken together, produce the required result for a specific instruction. Each operation is performed by three of the execute time pulses and may require from one to three of the basic functions.

JP (Figure 5-10)

Bits 6-8 of the jump and pushdown group are decoded into eight individual instructions. For controlling transfers between MB and AR at the end of the execute cycle, there are two subsidiary levels—one that represents all JP instructions except JSP, and another that represents the pushdown and pullout instructions. For PC control, JP JMP represents the six instructions

that jump, i.e., all except PUSH and POP. Three of the instructions that save PC (PUSHJ, JSR, JSP) also generate JP FLAG STOR to save the miscellaneous bits (JSA also saves PC but it stores E instead of the miscellaneous bits).

XCT, UUO (Figure 5-10)

IR XCT causes ET3 to trigger XCT T0, which returns the processor to the instruction cycle at a point beyond the memory subroutine so that the processor then executes the operand in MB as an instruction, just as though it had been retrieved by the instruction cycle. For all codes in the UUO class, the execute clears MA and MBLT, then transfers address 40 into MA and the instruction code into MBLT. The last event is equivalent to UUO T0, which triggers a memory write to deposit the trapped instruction, with its address portion replaced by the calculated effective address in location 40. The memory return triggers UUO T1 to index MA, and UUO T2 makes a read request and sets IF1A in the instruction cycle logic. The memory return automatically triggers the remainder of the instruction cycle so the processor performs the instruction in location 41.

5.4 PROGRAM CONTROL

Each instruction in the program is retrieved from the memory location addressed by the 18-bit program counter (Figure 5-11). At the beginning of every execute cycle, the counter is stepped one position so that instructions are taken from consecutive memory locations. The program controls its own sequence by means of skip and jump instructions. Skip instructions cause the processor to skip one instruction in the normal sequence if a specified condition is satisfied; the skip is implemented by advancing PC one extra position at the end of an execute cycle. Jump instructions transfer program control to any chosen location, sometimes upon satisfaction of a condition, by loading a new address into PC. An address can be transferred in only from MA so any input from the console ADDRESS switches or a jump address from MB must be made via MA and the transfer must be preceded by a clear. The flip-flops are connected in a carry configuration so a pulse at the PC+1 input to PC35 adds 1 to the contents of the counter.

Figure 5-12 shows the control logic for the counter: three control pulses are produced in the upper half; the lower half shows the generation of the gates that control counting, skipping,

and jumping. Every program begins with the console operation start or read in, in which KT1 clears PC and KT3 loads the ADDRESS switches into it. For normal counting, PC is incremented at the beginning of every execute cycle. The circumstances which inhibit program counting at ET0 are those in which another execute cycle will occur before the next instruction in normal sequence is to be performed. Character operations generally require two main parts, the second beginning with the address cycle, so PC is not incremented during operations on the pointer except in the single instruction that affects only the pointer and has no second part (C3). XCT, UUO, and a block IOT BLT all involve execution of instruction pairs and counting occurs in the second execute cycle. Since a PI cycle interrupts the normal sequence between instructions, the count must be inhibited in it because the processor has not yet executed the currently addressed instruction. Counting is inhibited throughout a block transfer because the BLT subroutine returns to the fetch cycle to process each word. When the block is complete (MQ0 = 0), PC is counted directly from the subroutine at BLT T5A. The inhibit also applies to an instruction executed from the console.

For changes in the program location out of normal sequence, PC SET causes ET7 to clear and ET8 to load, whereas PC+1 (ET9) causes ET9 to count. The net for PC SET has as input all unconditional jump instructions and an enable level which is asserted for any conditional jump when the condition is satisfied. Conditions include JFCL when the addressed flag is 1, the add-1 jumps when AR has the appropriate sign, and those arithmeitc compare instructions that use an accumulator when the test is satisfied. An extra count occurs at ET9 on two unconditional subroutine jumps so that the subroutine begins one place beyond the storage location of PC or AC. Any other extra count is for skipping and is represented by PC+1 ENABLE. The skip conditions include the two IOT status test instructions when the appropriate condition appears in AR, and the satisfaction of the test condition for any logical compare instruction or those arithmetic instructions that compare memory against 0 or an accumulator against either E or C(E). There is another conditional skip for a block IOT that does not use the PI system: if the indexing of the pointer did not produce a carry into AR0, IOT T0A counts PC (the computer performs the next instruction in normal sequence only if the block is complete). If any jump or skip occurs (other than an IOT block skip), the OR gate at the right generates the level PC SET $\vee$ PC+1 for setting the PC change flag (6.2e).

5.5 EXECUTIVE LOGIC

The executive logic allows the executive routine to control the sharing of processor and memory by a number of programs. The executive routine selects a user program and the area in core assigned to it by loading the protection and relocation registers, placing the processor in the user mode, and jumping to a location appropriate to the selected program. The user program may be interrupted temporarily by a block IOT in a PI cycle (which is under control of the executive routine and is hence unrestricted); but if a JSR is performed in a PI cycle (such as following the completion of a block IOT or for servicing some other type of interrupt), the processor leaves the user mode with control returning to the executive routine. Other than for priority interruptions, the user program has control until it attempts to use a protected area of memory or to perform an illegal instruction. The former action causes the processor to go the end of the current main sequence and triggers a priority interrupt on the processor channel; the latter causes the processor to perform the illegal instruction as a UUO (all UUOs are unrestricted but automatically return control to the executive routine).

Figure 5-13 shows the executive logic except for the memory protection and relocation registers which are discussed with the memory address logic (7.1c). The pulse amplifiers at top center provide the clear and set pulses for PR and RLR when the DATAO clear and set pulses are gated by CPA, i.e., when a DATAO for the processor appears in the program. The only unprogrammed clear for these registers is the master start because their states must remain until deliberately changed by the program. Similarly, the user flag and the illegal operation flip-flop (A3, B5) are also cleared only by the master start because they must maintain control functions from one main sequence to another. The remaining three flip-flops in Figure 5-13 provide synchronization and are cleared by the master clear. However, since the set function for EX PI SYNC is a level output from the PI cycle flip-flop, it remains set as long as PI CYC is 1 even though there may be a master clear between a pair of PI instructions. After PI CYC is cleared, the sync remains set until the beginning of the next main sequence. EX UUO SYNC is set at the beginning of every address cycle and then cleared at the end of the main sequence. The purpose of this flip-flop is merely to prevent the generation of the UUO command level during the time that IR is clear in the instruction cycle.

To set up a specific user program, the executive routine loads PR and RLR with a processor DATAO. It then jumps to the location for the program with a JRST that also sets the executive

mode sync flip-flop (B3). The setting may be done either by programming a 1 in bit 12 of the JRST, or by restoring the flags (a 1 in bit 11) provided that a user program was running at the time the flags were stored. Instructions that store the miscellaneous bits with PC store EX USER in bit 5, but the restoring JRST does not act on the user flip-flop directly. Instead, it sets the executive mode sync if MB5 is 1. The processor does not leave the executive mode until the end of the main sequence in which the sync is set: at this time, the transition of the sync back to 0 sets EX USER.

The addresses in the user program are checked against PR to determine whether they are legal and relocated to the area assigned to the program. The net in the lower right generates a level that inhibits both relocation and protection; the inhibit is always asserted if the user flag is 0. The net also inhibits the relocation of fast memory addresses so that these locations are available to all programs (the protection inhibit is really unnecessary here because these addresses are ipso facto less than the minimum block size). There is also no protection or relocation of addresses that occur in a PI cycle because the instructions executed are under control of the executive routine and must be unrestricted even if a user program is running. Again only the relocation inhibit is necessary for the PI channel addresses, but both inhibits are necessary for the addresses given by the instructions. If the interrupt should require a jump to the executive routine, the JSR that calls the routine also stores and clears the user flag (B2).

The nets in the upper right monitor user instructions by receiving the IR decoder outputs for JRST, IOT, and UUO to generate the UUO command level. The JRST or IOT decoder output drives the corresponding command line only if the instruction is not executed as a UUO. When the system is in user mode, EX IR UUO replaces a JRST that attempts to dismiss a priority interrupt or halt the processor and replaces user IOT (a block IOT in a PI cycle is not part of the user program). Any UUO regardless of mode enables EX IR UUO (after the UUO sync is set); this thus provides a means by which a user program can communicate with and return control to the executive routine. EX IR UUO inhibits relocation since it must use addresses 40 and 41, and at ET1 it sets EX ILL OP. The 1 state of this flip-flop then continues the inhibit into the JSR in location 41. At ET7 the JSR clears the user flag and at ET8 the illegal operation flip-flop. Just in case a block IOT that does not overflow should interrupt the JSR, EX ILL OP is cleared by all block IOTs.

The illegal operation flip-flop is also used to inhibit relocation during an examine or deposit that is inserted between two main sequences while the processor is running. For this purpose EX ILL OP is set by ST7 at the same time that the key cycle is triggered and cleared by the read/write return.

CHAPTER 6

ARITHMETIC LOGIC

The first half of this chapter describes the registers used for arithmetic calculations. Three full-word registers, AR, MQ, and MB, are used for computations on full-word and half-word fixed-point numbers and the fractional parts of floating-point numbers. All data transfers and logical operations on computer words are also performed in these registers. Besides the full-word registers, there are two 9-bit registers SC and FE that are used for floating-exponent calculations and for various subsidiary computations such as calculating the size and position portions of the pointer in character operations and counting the number of steps required for fixed- and floating-arithmetic operations. Included with each register is a discussion of its input gating, its control logic, and any other hardware associated specifically with the register (e.g., the flag logic and AR subroutines with AR, the SC subroutines with SC).

The second half of this chapter describes the time chains that control the execution of data and arithmetic instructions outside of the execute cycle. The test discusses the generating conditions for the pulses in a given chain and also describes the events that occur at each step. In many instances, a number of pulses from the different time chains all must trigger the same operation in an arithmetic register. Many subroutine time pulses are therefore connected to the register control logic through OR gates in a subroutine interface (6.5). Lines from this interface are labeled by the functions they perform and all have the prefix CFAC (computer floating-arithmetic connection). When only one or two subroutine time pulses trigger a given event, they are supplied directly to the register logic from the subroutines. In the following discussion of the arithmetic registers, the significance of the generating conditions for an individual register operation is given only for the execution of those instructions performed in the execute cycle and for other events in the main sequence. To determine the significance of any event triggered by a subroutine time pulse, either directly or through the subroutine interface, the reader should refer to the discussion of the appropriate subroutine.

The three full-word arithmetic registers are contained in the same set of double-height modules, that is, MB_i, AR_i, and MQ_i are all on the same module (it also contains the ith bit of the

memory indicator register MI, which is described in the memory logic). Each 36-bit register is controlled as a pair of half registers and is shown in two logic drawings, the first for the left half in mounting panels D and E of bay 2, the second for the right half in panels H and J of the same bay (e.g., see MB, Figures 6-1 and 6-2). The flip-flops of all half registers are in panel locations 5 to 12 and 16 to 24. All control pulses are supplied to them by a pair of pulse amplifiers in locations 14 and 15. On the register drawings, these are shown merely as blocks with the pin connections labeled; the actual pulse amplifiers with input gates are shown in the drawings of the control logic associated with the registers. Each PA output drives 18 input gates and is connected to 18 register modules for ordinary 0 or 1 transfers, but to only 9 for jam transfers. All inputs to every register bit are labeled by signal name; but since all the registers are contained in the same set of modules, connections between corresponding bits are made internally and pin connections are listed only for external signals.

6.1 MEMORY BUFFER

Figures 6-1 and 6-2 show the left and right halves of the full-word memory buffer. All transfers to and from memory are made via MB, but the register is discussed here because it holds one of the operands in most arithmetic and logical instructions. Each MB flip-flop has a direct clear input (which receives the register clear pulse) and gatable clear and set inputs. The MB modules include six sets of internal gates, but the gatable inputs are also available at the connector so that transfers can be made with single-bit pulses from external gates. Transfers of either zeros or ones may be made into MB from either AR or MQ. The two halves of MB may also be swapped; i.e., the left may be jam transferred into the right and the right into the left. For various executive instructions, the program counter may be stored in the right half of MB; the instruction register in MB LT. Transfers of information into MB from the memory bus (7.2) are made via single-bit pulse set inputs shown at the bottom of the figures. Clear inputs are also available, but these are used only for bits 1-8.

The external gates for the single-bit inputs other than those from the memory bus are shown at the top in Figure 6-4. Besides the clear gates for bits 1-8, there are also set gates for bits 0-8 that parallel the memory bus inputs. The gates are controlled by three transfer pulses, two of which set or clear bits 1-8; whereas the third stores the miscellaneous bits in MB0-5.

The generating logic for these transfer pulses is shown in the upper right of Figure 6-3. The set and clear functions occur in floating-point operations to nullify the exponent part of the register and are conditioned by the MB sign bit (refer to the appropriate subroutine). Storage of the miscellaneous bits occurs at ET6 in several of the subroutine-calling jumps. These bits include the four AR flags which can be used by a subroutine but return to their original states following the interruption. The miscellaneous bits also include CHF7 and EX USER. The latter is saved so that the executive routine can return control to a user program by restoring the flags. Saving CHF7, which is set at the end of the first part in a character operation, allows the computer to return properly to the second part if there is a priority interrupt between the two.

The remainder of Figure 6-3 shows the logic that governs all of the regular MB transfers. The top section shows the pulse amplifiers that drive the register gates; these PAs are triggered through OR nets whose inputs come either from the subroutine interface (6.5) or from the transfer gates in the center section of the figure. Timing inputs to these gates are supplied by the various main sequence and subroutine chains; the gating levels for the execute time pulses are supplied by the OR nets shown in the bottom section.

The clear function (upper left) is not required for transfers to MB from AR or MQ since these are always jam transfers. For transfers in from memory, a signal from memory control clears MB before the single-bit pulse inputs arrive via the bus. For a UUO, ET1 clears only the left half of MB leaving the effective address in the right half, and the ET3 loads IR into MBLT (D2, B7). The transfer pulse also triggers the UUO subroutine (5.3). MB must also be cleared before the transfers from PC that occur in various jumps. These jump instructions generate the level MB PC STO (C3) which clears MB at ET5 (B1) and transfers PC into it at ET6 (A6, B6). All but the restoring jump JRST also save the miscellaneous bits along with PC (A8). There is one jump, JSA, that transfers PC at ET6 but is not included in MB PC STO; this instruction gates ET6 via the diode net in D6. No prior clear is required because the left and right halves of MB are swapped at ET0 (D7) placing E in the left half and leaving the right half clear.

The two diode nets in D7 provide six conditions for the left-right swap of MB. Three of these conditions are standard instruction modes and a fourth, JSA, is described above. For the

other two, CONO makes the swap so that E is available over both halves of the I/O bus, and BLT does it to restore the address pair to its original configuration in order to store the data word in the location specified by the right half. These two levels gate the appropriate execute time pulses at B2 to trigger the transfer PAs in A2, 3. The swap is also triggered at FT1A (B2) whenever the processor must fetch a word addressed by ACLT (5.2c). The fetch cycle swap includes BLT, which then requires the second swap to restore the original address pair at ET1.

The logic that controls transfers from AR to MB occupies the entire center portion of the drawing from top to bottom. Almost all the transfers are of zeros and ones simultaneously, but there are several cases in which zeros are transferred alone (B3). In the ACBM group, a 0 transfer is always made at ET1 (the group command level is inverted through the diode net in D5) to AND the data word with the mask; then at ET6 zeros are transferred again if the masked bits are to be cleared. The 0 transfer is also used in a character deposit to produce an actual transfer: the character portion of MB contains all ones and all bits outside of the character in AR are also ones. The other three control pulses trigger the transfer of zeros and ones together: they jam transfer AR to MB, but two of the signals (one from the subroutine interface) are also applied to AR control (6.2c) to trigger the transfer from MB to AR at the same time. The double transfer always occurs in the fetch cycle to move AC to AR and E to MB; FT1A does it (D3) if there is no fetch of an additional word addressed by either half of AC, otherwise FT3 makes the transfer (B3). The MB-AR switch is required at three points in the block transfer subroutine (6.6a); the remaining transfers are at execute times gated by levels generated in Figure 6-3 (ET0, C2; ET4, D8; ET9 and ET10, C6). In a few instances, the transfer is actually necessary in only one direction but additional hardware would be required to eliminate the superfluous one. For an explanation of these cases as well as any other events that are not immediately obvious in the flow charts, refer to 4.5 (also see below).

The single transfer from AR to MB (B5) follows index register modification in the address cycle at AT3A to move the calculated address back to MB; ST5 triggers the transfer in order to store the result of an instruction in an accumulator. Any deposit or instruction execution from the console uses the transfer because the contents of the DATA switches can be sent to MB only via AR; AR also goes to MB at the beginning of the multiply and divide subroutines (6.8b, c). In the execute cycle, the transfer may be made at either ET0 or ET10: the former involves only standard instruction situations (D5), but the latter is complex and requires some comment.

In most instructions, the result appears in AR; and if it is to be stored in an accumulator, the transfer from AR to MB occurs at ST5 (the MB-AR switch at ET9 in ACBM is made specifically to move the result from MB to AR because the transfer in the opposite direction occurs automatically for AC storage). However, if the result is to be stored in E, either by requesting a write or restarting a read/write, the result is transferred from AR to MB at ET10 in preparation for the store cycle unless a transfer inhibit is asserted. The net that generates the transfer gating level is in C7; the net for the inhibit is in D4. The conditions generating the inhibit represent situations in which the result is already in MB or is being moved there by a 2-way transfer. In EXCH, the switch is made at ET0 (C2); in a character deposit, the character is inserted into the data word in MB. The transfer is also inhibited for all instructions in the jump and pushdown group: instead these instructions use the 2-way transfers at ET9 and ET10 (C6). At ET9, the switch is made for all instructions except JSR, whose result is already in MB (flowchart, Figure 4-7 left). In JSP, JSA, and JRA, the ET9 transfer results in the appropriate configuration for the store cycle. In the other four instructions, represented by the logic level JP $\wedge$ IR6(0), the result is already correctly placed before ET9, so a second switch of MB and AR is made at ET10. This double switch is made so that the address from AC right is available to MA at ET10 in PUSH and PUSHJ (the transfers are unnecessary in POP and POPJ, but extra hardware would be required to eliminate them).

The remaining transfer is that of both ones and zeros from MQ (Figure 6-3, upper right) which is triggered by two signals from the subroutine interface, one for a single transfer, the other for a switch of MB and MQ. The conditions within MB control that trigger the transfer (B5) include FT4A to return E to MB following a second accumulator fetch operation, and ST6 prior to storing a second accumulator. The transfer also occurs at the beginning of the block transfer subroutine (6.6a). In the execute cycle, it is used at ET0 in three of the JP instructions (C1) to move to MB the word fetched from a location addressed by half of AC.

6.2 ARITHMETIC REGISTER

Figures 6-5 and 6-6 show the left and right halves of the full-word arithmetic register. Each AR flip-flop has a direct clear input (which receives the register clear pulse), gatable clear and set inputs, and two complement inputs, one of which accepts a positive-going pulse, the other a negative-going pulse. The AR modules include ten sets of internal gates, but the

gatable 0 and 1 inputs are also available at the connector so that transfers can be made with single-bit pulses from external gates.

The AR outputs are connected within the 6205 modules to the input gating of the memory buffer and are also available through the module connectors for connection to the shift gates in other AR flip-flops, the in-out bus, and various other places in the processor. The outputs of the sign bit AR0 are used throughout the processor for control purposes, such as in the program control test nets and in many of the arithmetic subroutines, so these outputs are buffered by the drivers shown in the lower left of Figure 6-5. The 0 outputs of all AR bits are also available at the connectors through diodes that are joined externally to form large AND gates as shown in the left of Figure 6-10. Assertion of the output from a single AND gate indicates that the corresponding portion of the register contains all zeros (note that bit 9 is not included among the four gates). The first stage decoder outputs are further ANDed in the two nets in C3: the lower net decodes AR for the condition that every bit is 0, the upper net for the condition that bit 9 is 1 and bits 10 to 35 are all zeros. The latter condition, represented by assertion of the signal AR = FP HALF, is necessary for normalizing in floating-point operations. The floating-point fraction -1/2 (AR bits 0 and 9 both 1, and zeros in bits 10 to 35) is considered normalized even though bit 9 is the same as the sign bit.

In addition to the arithmetic register gating and control, this section describes the addition algorithm, the AR subroutines, and the flag logic.

a AR Gating

The external gates for single-bit pulse inputs are shown at the bottom of the two AR drawings, Figures 6-5 and 6-6. The only inputs presently used are for bits 0-8 and the gates for these are shown in the lower part of Figure 6-4. The upper pair of transfer pulses merely clear or set bits 1-8; the lower pair (the two gates at the lower left are triggered by the same transfer pulse as the row above) provide jam transfers from SC1-8 to AR1-8 and from SC3-8 to AR0-5.

The bottom two sets of gates in the register drawings, Figures 6-5 and 6-6, provide 1 transfers into AR from the I/O bus and the console DATA switches. Both of these transfers are preceded by the clear pulse for the register. The next two rows are jam transfer gates for right and left shifting; gating levels for a given bit are the 0 and 1 outputs of the adjacent bits. Since

at the register extremities the connections vary depending upon the type of shift, there are special inputs for the left shift gates at AR35 and for both sets at AR0. The generating nets for these special shift inputs are shown in the upper portion of Figure 6-7. At the left are listed the different types of shifts with the time pulses at which they occur and block diagrams showing the shift configurations. These diagrams are also drawn at the appropriate places in the flow charts. Among the level inputs to the shift nets are several composite functions that represent groups of conditions, all of which require the same shift type. The arithmetic shift of AR and MQ combined (B4) is required by the corresponding shift operation, but it is also required in the normalize return and floating-add subroutines. Another composite function is SHC DIV, which is asserted by any type of division provided NRF2 is 0 (D4). The flip-flop condition does not apply to fixed division but is necessary in control over floating division so that the control level cannot affect shifting in the normalize return subroutine, which follows all floating-point arithmetic subroutines. Just above is another composite function (C4), which is asserted during a division or a combined logical shift.

The three shift input nets use these composite functions as well as individual instruction levels to determine the effect of any given shift on the AR extremities. In left shifting, AR0 receives the state of AR1 unless some type of arithmetic shift is being performed, in which case AR0 is unaffected. The center net controls AR0 whenever AR is shifted right. The upper two gates make MQ35 the source of data for AR0 on a combined rotation and AR35 the source on a single rotation. The gates at the lower right of the net disable the 1 input so that AR0 is automatically cleared in a character load or a logical shift; the gate at the lower left disables both inputs so that AR0 is unaffected by any right shift in multiplication or division, or a right arithmetic shift. The net at the right provides shift left input to AR35 from AR0 in a single register rotation, from MQ1 in any double length arithmetic shift, and from MQ0 in a combined logical shift, combined rotation, or division shift. The gate at the lower right disables the 1 input so that AR35 is automatically cleared in a single left arithmetic or logical shift, or a character deposit.

Above the shift gates are three rows of gates that use the outputs of the corresponding bits in MB. The bottom two rows supply 1 and 0 transfers that provide the OR and AND functions of MB and AR when used separately, but provide a jam transfer when pulsed together. The third set is connected to the AR complement inputs and is conditioned by ones in MB. Pulsing

this set of gates produces in AR the exclusive OR of MB and AR. The next row of gates above the MB gates provides a simple complement function, i.e., pulsing these gates complements all AR bits. The top two sets of gates generate carry pulses, both of which are applied to the flip-flop to the left of the module containing the gate that generates the pulse. For example, the gates on the AR8 module use AR8 and MB8 outputs as level inputs, but the carry output complements AR7 (this method of placing the gates saves pin connections because the level inputs are internal to the module). The two sets of gates provide related carry functions, a ripple carry and a carry initiate. The lower set, the ripple carry, is a serial function, i.e., there is no control pulse applied to all gates simultaneously. The chain starts at AR35 with the pulse AR+-1T1, which is applied both the the AR35 complement input and the lower carry gate. This pulse, which occurs only when adding 1 to the contents of the register, complements AR35; and if AR35 is 1, it also triggers the ripple carry to AR34. This second carry in turn complements AR34; and if that bit is a 1, carries on to AR33. The chain continues through the register in this manner except for a break between AR18 and AR17: the carry out of AR18 automatically enters AR17, but the latter may be pulsed independently in order to index two 18-bit words simultaneously.

The upper set of carry gates provides the full-register carry-initiating function for addition. This arithmetic operation is carried out in two stages, first a partial addition, then a carry function. The partial addition is the exclusive OR function of MB and AR (AR $\leftarrow$ MB($\forall$)). After the partial sum (the result of the partial addition) has been formed, the full-register carry-initiating pulse triggers the upper set of carry gates to change the exclusive OR into the arithmetic sum. At the end of the operation, the number in AR represents the sum of the contents of MB and the previous contents of AR.

For any given bit, the partial sum (the exclusive OR function) of two numbers is actually the correct sum if there is no carry into that bit. But if there is a carry for that bit, the partial sum is the opposite of the arithmetic sum. For each bit where both summands are 1, the carry initiate directly complements the next more significant bit. However, since the processor cannot sense the prior state of a flip-flop, it instead senses the corresponding configuration of the partial sum. If after partial addition, AR_i is 0 and MB_i is 1, both bits must originally have been 1 and the carry therefore complements AR_{i-1}. Anytime a bit is complemented, a ripple carry is initiated into the next more significant bit, but this carry is inhibited if the

bit was complemented from 0 to 1. At each stage, a carry produced through the upper gate goes not only to the complement input of the next more significant bit but also to its lower gate. Thus, a carry initiated by the partial addition of two ones ripples up the register until it terminates when a 0 bit is complemented. That this algorithm does produce the correct sum of two binary numbers is proved below.

b Addition Algorithm

Let A be the original contents of AR, B the contents of MB, PS the partial sum produced in AR by the partial addition, and S the arithmetic sum of A and B. For convenience, let A and B be positive binary fractions whose sum is less than 1, i.e., there is no overflow. A bit of the partial sum PS_i is equal to a bit of the sum S_i if there is no carry into S_i. If there is a carry, PS_i is the complement of S_i. Since there can be no carry into the least significant bit, $PS_{35} = S_{35}$.

To understand the operation of the two carry functions, divide PS into sections from the right so that the first section starts with PS_{35} and ends at the first bit PS_i that satisfies the conditions $PS_i = 0$, $A_i = B_i = 1$. The second section starts with PS_{i-1} and extends to the next bit that satisfies the same conditions as PS_i. Proceed in this way through the entire partial sum. Since there can be no carry input to the least significant bit of the partial sum, it must be correct. If this bit is 1 or if it is 0 resulting from the partial addition of two zeros, there is no carry out and the next bit of the partial sum is also correct. Proceed with each more significant bit of the partial sum until reaching the bit PS_i, which is 0 resulting from the partial addition of two ones. This bit is also correct; therefore, all bits in the first section are correct.

Because the partial sum in PS_i generates a carry, PS_{i-1} is not correct and a 1 from the first section is carried into it by the carry initiate. If PS_{i-1} is 1 (resulting from the partial addition of 0 and 1), there is a ripple carry into PS_{i-2}. The ripple carry propagates up the register until a 0 bit is encountered. If this 0 is the result of a partial addition of two zeros, no further carry is generated; all further bits are correct up to the next 0 that results from the partial addition of two ones, i.e., up to the end of the section. If the 0 that terminates the ripple carry results from the partial addition of two ones, there must be a carry into the next bit. However, the partial addition of two ones is the condition that ends the section and the

carry initiate begins a new ripple carry in the next section. Consequently, the carry complements all incorrect bits of the partial sum. At the completion of the carry operation, the result S is the correct sum of A and B.

The preceding example shows that the addition algorithm works for the special case of two positive numbers. Before proving the algorithm for the remaining cases, including negative operands, some further facts should be understood:

The sign bits are included in the partial addition, i.e., the partial sum of two minus signs (ones) is a plus sign (0).

Both carry functions are applied to the sign bit AR0, which is treated as though it were a next more significant bit of AR.

The sign bit conditions both carry functions: there is a carry out of AR0 if two negative numbers are added, or if there is a carry out of AR1 and the sign of the partial sum is minus. Carries into and out of the sign bit (i.e., carries out of AR1 and AR0) are used to detect overflow.

Assume that the binary point is to the left of the most significant bit, i.e., all positive numbers are 35-bit, fixed-point fractions. The computer representation of the positive number x is therefore +.[x] where the brackets enclose the number contained in AR1-35. The sign of this number is represented by the state of AR0. In 2's complement arithmetic, the negative of a number is produced by changing the sign and subtracting the magnitude from 1. The computer representation of the number -x is therefore -.[1 - x]. With this representation, there is no negative 0; the magnitude 1 - 0 overflows, changing the sign back to plus. Furthermore, the largest negative number is -1, represented by the configuration -.[0].

The four cases of addition of two positive 35-bit fractions are:

$$x + y$$
$$(-x) + (-y)$$
$$x + (-y), \quad y \leq x$$
$$x + (-y), \quad y > x$$

Since the 2's complement format allows a representation for -1, either x or y may be 1 in the second case and y may be 1 in the fourth case. In the first case, which is discussed above, the contents of AR after addition represent the number $+.[x + y]$. If $(x + y) \geq 1$, the carry out of AR1 changes the sign. Consequently, if the addition of two positive numbers results in a negative answer, it is apparent that the sum has exceeded the capacity of the register. The processor detects the overflow by checking the sign bit carries: there is a carry from AR1 but none from AR0. The contents of AR then represent the number:

$$-.[x + y - 1]$$

In case two, the addition of two negative numbers, the partial addition and all carries except that into the sign bit would be:

$$\begin{array}{l} -.[1 - x] \\ \underline{-.[1 - y]} \\ +.[1 + 1 - x - y] \end{array}$$

If $(x + y) \leq 1$, the AR1 carry changes the sign and the complete result is:

$$-.[1 - x - y]$$

which is the computer representation of $-(x + y)$. If $)x + y) > 1$, there is no carry into the sign bit, and its absence in the presence of a carry from the sign bit indicates overflow. AR then contains:

$$+.[1 - (x + y - 1)]$$

In case three, the addition of x and -y, where y is less than or equal to x, the partial addition and all carries except that into the sign bit would be:

$$\begin{array}{l} +.[x] \\ \underline{-.[1 - y]} \\ -.[1 + x - y] \end{array}$$

Since $y \leq x$, it follows that $(1 + x - y) \geq 1$. Hence the AR1 carry changes the sign and the complete result is $+.[x - y]$. Since the signs of the operands are different, the magnitude of the result cannot exceed the larger operand and there can be no overflow. Although there is an AR1 carry, the minus sign resulting from the partial addition allows it to ripple through producing an AR0 carry.

In case four, the addition of x and -y with y greater than x, the partial addition and the carry function are:

$$\begin{array}{l} +.[x] \\ \underline{-.[1-y]} \\ -.[1+x-y] \end{array}$$

Because $y > x$, it follows that $(1 + x - y) < 1$. Hence there are no carries from AR1 or AR0 and no overflow. The above result is the 2's complement representation of the number $x - y$, i.e., $-(y - x)$.

Addition is also used in fixed-point subtraction with the minuend x in AR, the subtrahend y in MB. The subtraction could be performed by taking the 2's complement negative of x, adding -x to y and taking the negative of the result. It is much simpler however to complement the word in AR, then add and again complement the result in AR. The complement of a word, which is produced by exchanging all ones for zeros and zeros for ones, is equivalent to the arithmetic 1's complement in which the sign is changed and the magnitude is subtracted from all ones, i.e., from $1-2^{-35}$. The complement of x is thus:

$$1 - x - 2^{-35};$$

adding y to $\sim x$ yields:

$$1 - x + y - 2^{-35}$$

which equals:

$$1 - (x - y) - 2^{-35}$$

which is the 1's complement of $x - y$. Overflow is indicated in the same way as in addition; that is, by an AR1 carry without an AR0 carry or vice versa. Overflow is properly indicated even for operations involving -1 because the 1's complement is 2^{-35} more negative than the 2's complement (i.e., when subtracting -1 from 0, the proper carries occur in the addition of -1 to the complement of 0).

c AR Control

Figure 6-8 shows the generating logic for the AR control pulses. The pulse amplifiers that drive the register gates are in the top section; these PAs are triggered through OR nets most of whose inputs come either from the subroutine interface (6.5) or from the transfer gates in

the left center section. Timing inputs to these gates are supplied by the various main sequence and subroutine chains, and the gating levels for the execute time pulses are supplied by the OR nets at the lower right. Six of these diode nets are used almost exclusively to gate the AR events necessary for the execution of the Boolean instructions. The flow chart (Figure 4-8) lists at each time the instructions that require each event, and also lists the specific sequence of events necessary for each of the 16 Boolean functions. Because there are no complement gates for MB, all functions that require the AND with the complement of memory use the OR and complement the result; the same is true for the converse. BOOLE 14, which places the complement of memory in an accumulator, clears AR first and then uses the exclusive OR as a 1 transfer. All other functions are quite straightforward. The other diode nets in Figure 6-8 gate the events necessary for the data transmission, IOT, and compare instructions; the functions that are not immediately obvious from the flow charts are described in 4.5.

The pulse amplifiers that clear AR are shown in the upper left corner of Figure 6-8. To leave the desired address in the right half, only the left half is cleared in the address cycle and the block transfer subroutine. For console operations, both halves are cleared by a signal from the key logic (5.1) and the subsequent transfer from the DATA switches is triggered through the PAs in B7. The other AR clear input is generated by the nets in B3 and C3. Subroutines that require the clear include divide, floating add, and multiply. The clear may also occur at the first two pulses in the execute cycle: it is required at ET0 for four of the Boolean functions (D2), at ET1 by any input IOT (D6) and in any half-word transfer that affects the other half of the destination location. For input instructions, IOT T3 triggers the subsequent transfer into AR from the bus (B8). For an HWT, there is no further action if the instruction is to clear the other half, or the transfer half is positive and the instruction is to extend it; however, to set, or to extend a negative half, the other half is set by complementing through the appropriate gate in B1.

All other complements affect the entire word. One input to the PA gates (A2) comes from the subroutine interface; another is the first pulse in the add-subtract time chain—this pulse occurs only if the subroutine is to subtract (d below). Corresponding time pulses for the index and negate subroutines trigger the complement through the top gate in C2. The gates below that provide the complement at ET0 or ET4 for Boolean functions, and at ET5 and ET7 for the ACBM

group in case the masked bits must be cleared at ET6. The bottom gate complements at the end of the carry function in an AR subroutine if the complement control flip-flop has been set.

The four PAs in A3 and A4 are triggered in several combinations to provide 0, 1, and jam transfers from MB into either or both halves of AR. All four must be triggered for a full-word jam transfer. The top two inputs provide for a simultaneous interchange of MB and AR; for any such transfers not made through the subroutine interface, the generating logic is shown with MB control (6.1). Half-word jam transfers include transfer of the address portion of the instruction word into ARRT at the beginning of the address cycle, and transfer of the selected half word in an HWT triggered by ET4 through the gates in B2 with gating levels supplied from the nets in C7. The left transfer also occurs in a CONO so that E is available on both halves of the I/O bus. All other pulses that trigger the PAs in A3 and A4 are for full-word, one-directional transfers. Transfers of zeros alone (B4) are required in both the deposit and load character sequences (6.6b), at ET1 in four of the Boolean functions (D4) and at ET6 for the two IOT status test instructions (D8). The 1 transfer (C5) is used to set the masked bits in an ACBM (D4) as well as to provide a 1 transfer in SETM (BOOLE 3) and an OR function in four of the other Boolean instructions. The gates for the jam transfer are located between B3 and B4 in the figure. This transfer is required in fixed-point multiplication (6.7a), at ET0 in various data transmission, compare and IOT instructions (D5), and in character operations in the first part and in the load sequence of the second part (6.6b).

The PAs for the exclusive OR function of MB and AR (A5) are triggered when ET1 is gated at B4 by the level from D5. This function complements the masked bits in an ACBM, provides a 1 transfer in SETCM (BOOLE 14) and an exclusive OR in XOR and EQV (BOOLE 6 and 11). The exclusive OR pulse amplifiers are also triggered by AR AS T1 in the add-subtract subroutine to provide the partial addition. The next pulse in the chain initiates the carry function through the pulse amplifier in B6. The remaining PAs in the upper right of Figure 6-8 provide for the left and right shifts, which are triggered from the subroutine interface.

The control pulses for the special AR gates that are external to the arithmetic register modules are shown at the right in Figure 6-9. To understand the significance of these transfers, the reader should refer to the description of the subroutines in which they occur. In the exponent calculate and floating add subroutines, AR bits 1-8 are cleared or set according to whether

the sign bit is 0 or 1 in order to nullify the exponent part of the register. The jam transfer from SC1-8 into AR occurs in the normalize return and floating-scale subroutines to insert the exponent in the result; a similar transfer of SC3-8 to AR0-5 occurs in the first part of a character operation to insert the new position if the pointer has been incremented.

d AR Subroutines

Figure 6-9 shows the logic governing the negate, index and fixed-point add and subtract subroutines; the flowchart for these is in the upper right of Figure 4-8. In the lower right of Figure 6-9 are the nets that provide the level gates to trigger and control the subroutine time chains. The add gate is asserted only for fixed-point addition, the subtract gate for both fixed-point subtraction and the instructions that arithmetically compare an accumulator against memory. The add 1 and subtract 1 levels are generated by instructions requiring upward or downward indexing of either the entire AR or its two halves independently. AR+1 is asserted in any MEMAC instruction that adds 1 to the test word, in the two add-1-to-both jump instructions, and to index the pointer in any pushdown instruction or block IOT. Similarly, AR-1 is asserted for the MEMAC instructions that decrement the test word and to decrement the pointer in the pullout instructions. Between the add 1 and subtract 1 nets is a net that generates the level AR+-1 LTRT, which causes any entry into the carry chain at AR35 to enter the chain at AR17 as well. The double indexing occurs in the two add-1-to-both jumps and for operations on the pointer in a block IOT or in any pushdown or pullout instruction (these last four are included in the level JPΛ IR6(0)). The net in the lower right corner generates the level that causes the execute cycle to pause after ET3 for an AR subroutine. AR SBR is generated by any of the four subroutine calling levels described above, and also for negation in a full-word transfer and for a floating subtraction (this last case is for negation of the subtrahend before entering the floating-add subroutine).

The AR subroutines are actually two separate time chains, each with multiple entries, that join for the return to the interrupted sequence. The chain in the upper left handles indexing and negation; downward indexing and negation both require a preliminary pulse to complement AR. Subtracting 1 is done by complementing, adding 1, and complementing again; negation by complementing, then adding 1. All subroutines in the execute cycle are triggered at ET3. AR-1 triggers AR+-1 T0, which complements AR and also sets the complement control

flip-flop (C5). The complement pulse for negation may be triggered by a signal from the subroutine interface (6.5) as well as by ET3 in a floating subtraction or the appropriate FWT. Following either T0 pulse, there is a delay to allow the register complement to function before triggering AR+-1 T1 which carries into AR35 adding 1 to the contents of the register. This latter pulse is also triggered directly by time pulses from block transfer, normalize return, and character operations as well as by the input for execute cycle incrementing. For indexing both halves of AR together (this occurs in a block transfer as well as in those instructions that generate AR+-1 LTRT), the same pulse that carries into AR35 is also gated to carry into AR17. Of course, a carry into AR17 occurs automatically whenever there is a carry out of AR18.

Just to the right of the index and negate chain is a separate time chain for addition and subtraction. The first pulse in the chain, which complements AR, is necessary only for subtraction and is triggered not only in the execute cycle, but also from the subroutine interface, and in a block transfer to determine whether the block is complete. AR AS T0 sets the complement control flip-flop, and after a delay sufficient for the complement to function, triggers the partial add pulse AR AS T1. The chain begins at this pulse for any execute cycle or subroutine addition and also when adding an index register in the address cycle. After the partial addition, the chain continues to the next pulse which initiates the full register carry.

There are three pulses that initiate carries in AR. These are AR+-1 T1 in the right half, AR17 CRY IN in the left half and AR AS T2 which triggers carries at any required point throughout the register. All three pulses are applied to the OR gate in B1 cutting off the transistor just to the right by grounding pin 2F5Y. ORed with these carry initiating pulses is a vast OR gate (lower left) that receives the carry outputs from all AR bits. Once any carry function is initiated, all further carries are from the ripple carry, i.e., a carry for any bit can come only from a contiguous bit. Successive carries overlap because of the carry speed and the OR gate is enabled until all carries die out: 2F5Y then goes negative triggering the carry completion pulse. If the complement of AR is not required (i.e., the subroutine is not performing a subtraction) the completion pulse triggers the subroutine return ART3. However if the complement control flip-flop has been set, the carry completion complements AR, and after an appropriate delay triggers the return. ART3 clears the complement control flip-flop and triggers the next pulse in the time chain that has called the AR subroutine.

e AR Flags

The jump-addressable flags and associated logic are shown in the right half of Figure 6-10. Although some AR flag control functions also affect the user flag and the flip-flop that differentiates the two parts of a character operation, only the four AR flags can be sensed or cleared directly by the program. The PC change, overflow, carry 0,and carry 1 flags are cleared initially by the master start rather than the master clear because their states must remain from one main sequence to the next. Program control over the flags is exercised by three instructions, JRST with a 1 in IR11, JFCL, and the CONO for the processor. The last instruction, which governs the many flags in the processor I/O interface (8.3), clears the PC change flag if a 1 is programmed in bit 29 or the overflow flag if bit 32 is 1. JFCL selects flags with bits 9-12 of the instruction word and clears the selected flags after sensing them to make the jump. The JRST, which must be addressed indirectly, clears all the flags and restores them according to the first four bits of the word taken from memory as the direct address. The flag clear and set pulses generated here also handle the restoration of CHF7 and EX USER.

The remaining gates include conditions for setting the flags so that the program may determine whether or not certain events have occurred. Any jump or skip sets the PC change flag provided the instruction in process is not a JRST that is restoring the flags. All other gates are for arithmetic conditions indicating overflow. Overflow in most fixed-point cases is determined from the pair of flip-flops in the lower right, which are cleared at the beginning of every execute cycle and then set by carries out of AR0 and AR1. Since overflow is indicated by the presence of one of these carries in conjunction with the absence of the other (as is shown in the discussion of the addition algorithm in b above), the level AR OV SET is asserted when the two flip-flops have oppositestates; i.e., the overflow level is the exclusive OR of the two carry flip-flops. In fixed-point addition or subtraction or in the MEMAC instruction group—the latter condition being relevant only for those MEMAC instructions that index the test word—ET10 transfers the states of the carry flip-flops to the corresponding flags, and if overflow has occurred, sets the overflow flag. The remaining gates in C6 set the overflow flag on an overflow signal from the subroutine interface (6.5); on the attempt by an FWT to form the negative of -1; and on the loss of a significant bit in an arithmetic shift to the left, i.e., if a 1 is lost in a positive number or a 0 in a negative number (the exclusive OR input for this gate is generated in E5).

The top nets at the far right in Figure 6-10 provide control levels for use in various arithmetic subroutines: the exclusive OR of AR0 and MB0, and the complementary function of AR0 and SC0. The remaining nets at the lower right test for the inequality conditions in arithmetic compare instructions (5.3). The lower net ANDs the overflow condition with ∿ MEMAC, and this output is exclusive ORed with AR0 in the upper net. In a MEMAC, the output of the lower gate is automatically false and the output of the upper gate is true or false as AR0 is 1 or 0. In this case, since the comparison is with 0, the function specifies whether the test word is negative or positive, i.e., less or not less than 0. In the ACCP group, a test word from AC is compared against a standard, either E or C(E). The comparison is made by subtracting the standard S from the test word T. The four possible configurations of T and S result in the following signs and overflow conditions:

$T \geq 0$, $T < S$: - and ∿ OV
$T \geq 0$, $T \geq S$: + if ∿ OV, - if OV
$T < 0$, $T < S$: - if ∿ OV, + if OV
$T < 0$, $T \geq S$: + and ∿ OV

Hence, the function AR0 ⊻ AR OV specifies the relation between the test word and the standard: it is true for $T < S$, false for $T \geq S$.

6.3 MULTIPLIER-QUOTIENT REGISTER

Figures 6-11 and 6-12 show the left and right halves of the full-word, multiplier-quotient register. Each MQ flip-flop has a direct clear input, which receives the clear pulse for the register, and gatable clear and set inputs. The MQ modules include four sets of internal gates, but the gatable inputs are also available at the connector so that transfers can be made by single-bit pulses from external gates (at present, these are used only for MQ0).

The upper two sets of gates provide transfers of zeros or ones from MB. The lower two rows are jam transfer gates for right and left shifting; gating levels for a given bit are the 0 and 1 outputs of the adjacent bits. Since the connections at the ends of the register vary depending upon the type of shift, there are special inputs for the left shift gates at MQ35, the right shift gates at MQ1 and both sets at MQ0. The generating nets for the special shift inputs are shown in the lower portion of Figure 6-7. At the left are the different types of shift with the

time pulses at which they occur and block diagrams showing the shift configurations. These diagrams also appear at the appropriate places in the flow charts. Among the level inputs to the shift nets are two composite functions that represent groups of conditions, all of which require the same shift type. The arithmetic shift of AR and MQ combined (B4) is required by the corresponding shift operation, but it is also required in the normalize return and floating-add subroutines. Another composite function is SHC DIV which is asserted by any type of division provided NRF2 is 0 (D4). The flip-flop condition does not apply to fixed division but is necessary in control over floating division so that the control level cannot affect shifting in the normalize return subroutine, which follows all floating-point arithmetic subroutines.

The four shift input nets use these composite functions as well as individual instruction levels to determine the effect of any given shift on the MQ extremities. In left shifting (C4), MQ0 receives the state of MQ1 unless some type of double-length arithmetic shift is being performed in which case AR0 is the source. The next net controls MQ0 whenever MQ is shifted right. AR0 is shifted into MQ0 by any double-length arithmetic shift or by the final multiplication shift, i.e., when the shift counter contains 777. In all other circumstances, AR35 provides the input. The next gate to the right causes information to be transferred directly from AR35 to MQ1 in any double-length right arithmetic shift (MQ0 is skipped in the shifting), otherwise MQ0 shifts into MQ1. The final net connects AR0 to MQ35 for any left combined rotation but causes MQ35 to receive the complement of AR0 in a division shift. The two gates below provide constant inputs to MQ35. The left gate grounds the 0 input so that ones are always shifted in the first part of character operation; similarly zeros are shifted in a combined logical shift, in any double-length arithmetic shift, and in the character deposit sequence. There is a single additional gate that affects MQ0 when MQ is not itself shifted. In the first floating division shift at DST10A, only AR is shifted but AR35 must enter MQ0. The pair of gates at the left in Figure 6-13 allow entry through the single-bit pulse inputs, leaving the remainder of the register unaffected.

For use in multiplication, the register actually has an additional bit MQ36 (Figure 6-13, right). This bit receives the state of MQ35 on any right shift but is of significance only in multiplication (6.8b). For the multiply subroutine, the net at the right supplies a level indicating when bits 35 and 36 are equal. Another net in C4 supplies the exclusive OR function of MQ35 and MB0 for use by the divide subroutine (6.8c).

At the top of the figure are the pulse amplifiers that control the register. MQ is cleared only at the master clear and the shift PAs are triggered only from the subroutine interface (6.5). The remaining PAs provide transfers of zeros and ones from MB. Transfer of ones occurs in the deposit character sequence (6.6b). The jam transfer occurs on a signal from the subroutine interface that switches MB and MQ, and at the four time pulses listed at the gate in B4. The transfer at FT4 saves E in MQ if there is to be a second AC fetch operation; FT4A transfers this additional word into MQ at the same time that it returns E to MB. The remaining transfers occur in the multiply and divide subroutines (6.8b, c).

6.4 ARITHMETIC SHIFT COUNTING

In addition to the three full-word arithmetic registers, the arithmetic logic also includes two 9-bit registers for use in auxiliary computations and counting steps in arithmetic operations. The shift counter and the floating-exponent register are shown in Figure 6-14. Each module in the figure includes one flip-flop from each register and all associated gating. FE is used only for storage of intermediate results. In floating multiplication and division, the exponent is calculated in SC and stored in FE while SC counts the number of shift steps in the operation on the fractions. FE also provides temporary storage for the position portion of the pointer in a character operation.

Since actual computations are performed in SC, there is considerable gating associated with its flip-flops, including a carry chain; but FE is used only for storage so it includes only a direct clear and two sets of transfer gates connected to the flip-flop collectors. These provide 1 transfers from SC and from MB0-5 to FE3-8.

a SC Gating

Below the SC flip-flops in Figure 6-14 are the gates that implement the transfers, partial addition, and carry logic. In addition to gatable clear, complement, andset inputs, each flip-flop also has a carry input and a carry output. These are connected from one flip-flop to the next so that a pulse at the SC+1 input to SC8 adds 1 to the contents of the counter. The clear input is used only for the register clear. There are three sets of complement gates: the top one provides a simple complement function and the other two provide the partial add and carry functions which are described below. The remaining gates are connected to the 1 inputs.

The top gate provides an ordinary 1 transfer from FE to SC. The middle gate provides a complement transfer from MB18, 28-35 into SC. The last gate is a diode net (not part of the 6203 module) which receives no level inputs, but instead control pulses are applied to individual gates in order to place a specific number in the counter. In multiplication and division, for example, SC receives the complement of the number of steps to be counted; it is then incremented until it contains all ones, terminating the operation.

At the bottom of the figure are two networks, the lower a carry chain, the upper a set of level gates that supply the data inputs to the partial add (exclusive OR) gates at the complement inputs to the SC flip-flops. The partial add gates are used not only for partial addition but also for transfers provided SC is cleared or set first. The source of information for the partial add gates varies depending upon the operation in which SC is being used. The nets that generate the data levels are sets of four AND gates ORed together; each AND gate receives an enabling level that is common to all nets and an input from a single bit of a source register. For example, the top set of AND gates places the complement of MB0-5 into SC3-8, i.e., with SC clear, a 0 in a given MB bit causes the partial add pulse to set the corresponding SC bit by complementing it. For this function, the first three data levels are automatically asserted. For the other three sets of AND gates, the data levels are asserted by ones in the source register: the second set enables input from MB6-11 to SC3-8 with the first three bits negated; the bottom two sets enable input to SC from AR0-8 and MB0-8.

To calculate the exponent in floating-point operations and the pointer in character operations, numbers must be added in SC. The addition is performed with essentially the same algorithm used in the arithmetic register (6.2b). First, the partial add pulse produces in SC the exclusive OR of the contents of SC with the number represented by the data inputs, then the carry pulse adjusts the partial sum to produce the arithmetic sum. The carry function for SC differs from that in AR in that no ripple carry is used. The carry connections from one flip-flop to the next are for indexing and are not associated in any way with the full-register carry function; no carries propagate from one bit to the next when a bit is complemented by the carry pulse. Instead, as soon as the partial sum is formed, a series of level transitions from right to left across the carry chain determines the carries for all bits, and all bits are adjusted simultaneously by the carry pulse. The conditions for a carry are the same as those in AR. There is no carry into

the least significant bit (D8) and a carry out occurs only when two ones are added—a condition that is indicated by a 1 in the data and a 0 in the partial sum. For the other bits, there are two carry out conditions, one dependent upon a carry in. For any bit, there is a carry out if both summand bits are ones, or if the partial sum is 1 and there is a carry in. After the level changes have propagated through the chain defining the carries for all bits, the carry pulse complements those bits that receives carries; the SC8 carry supplies the gating level for SC7, and so on through the register. Since there can be no carry into the LSB, the carry gate for SC8 is disabled (B8).

b SC Control

The various functions of SC and FE are triggered directly from the subroutine time chains including the SC subroutines. There are no connections through the subroutine interface and SC functions are triggered by execute time pulses only prior to entry into a subroutine. The logic governing the SC and FE functions is shown in Figure 6-15. The conditions governing FE, which is used only for temporary storage, are quite simple: it is cleared only on the master clear (B6), and may receive an exponent from SC in the exponent calculate subroutine for floating multiplication or division (B7), or may receive the position portion of the pointer in the first part of a character operation (B4). In all cases, the information is subsequently transferred to SC (B2).

The complement transfer of MB18, 28-35 into SC (C8) is made at ET0 in a shift operation or floating scale. The only other function triggered from the execute cycle is the complement pulse, which occurs at ET1 in FSC if AR is positive (B3). All other functions govern SC for shifting or calculating in the data and arithmetic subroutine instructions, and they are triggered by pulses from the special time chains. In most instances, these pulses are ungated although many of those that trigger the complement are gated by sign conditions. Read the appropriate subroutine description (all are included in the final three sections of this chapter) to determine the significance of each SC event in a given subroutine. SC is cleared through the net in the upper left (which includes the master clear), complemented through the nets in the top center, and its partial add function is triggered through the net in the upper right. This last function is used for addition only in the SC add subroutine in which it is followed by the carry (B3); in all other cases, the exclusive OR is used for transfers. The source of information for the data

levels to the partial add gates is determined by the enabling levels in the upper right of Figure 6-16. A flip-flop in the logic for the appropriate subroutine enables the required input for each transfer or partial addition. There are no flip-flops associated with the SC add subroutine; instead the enable level is derived from a flip-flop in the main subroutine that calls for the SC addition. SC may also be incremented by 1 through the nets shown below the enable levels.

In the lower left of Figure 6-15 are several control signals derived from the SC outputs. The function in D6 indicates that the first three bits are all ones and hence SC contains a 1's complement negative less than or equal to 63 in magnitude. The net in D1 decodes SC0-7 for all ones and its output is ANDed with SC8(1) for a signal indicating that a shift-count has been complemented, i.e., SC has counted to -0 which is all ones. The termination of the count may also be indicated by a pulse through the pulse amplifier in C4. When SC0-7 are all ones, the next count pulse in theshift-count, multiply or divide subroutine triggers the PA to produce a leading edge at output SC8B. The PA output is ORed with the -0 configuration, so SC8B remains asserted even after the PA output disappears.

c SC Subroutines

There are two subroutines associated with the shift counter, an add subroutine for use in calculations on exponents and pointers, and a shift and count subroutine that counts the number of steps required in an operation and shifts the intermediate result at each step. The logic for these subroutines is shown at the left in Figure 6-16 and the flow charts are in the lower right of Figure 4-8. Listed with the shift-count subroutine are the entry conditions, the control levels governing the type of shift, and the pulses to which the subroutine returns in the interrupted sequences. Similarly, the flow chart for add lists the entry, the source enabled for the partial addition, and the return for each call.

The time chain for add is in the upper left in Figure 6-16. The first pulse always clears the character control flip-flop CHF1, although this is of relevance only in character operations. SAT1 then triggers the partial add, and after a delay sufficient for all level transitions through the carry chain, SAT2.1 triggers the carry. SAT3 then returns to the interrupted sequence.

The shift-count subroutine is used only in character operations, shift operations, and the floating-add subroutine. All other shifting counted by SC is produced directly by pulses from the arithmetic subroutines. Entry into the shift-count sequence at SCT0, which performs no operation but provides a delay before the first shift. If SC does not contain -0 (indicated by the condition that either SC8 is not 1 or SC0-7 does not contain all ones), SCT1 increments SC and triggers the appropriate shift (6.5). If the counting is still incomplete, the output of the delay triggered by SCT1 again triggers SCT1 for a new shift and count. When the count is complete, the delay output returns to the interrupted sequence via SCT2.

6.5 SUBROUTINE INTERFACE

Because the same event is often required at many different times in the various subroutines, the processor includes a subroutine interface that collects signals from the subroutines to reduce the number of signals applied to the control logic for the arithmetic registers. For example, all shifts of AR or MQ are triggered through the nets at the left in Figure 6-17. The lower set of nets, which includes level gates, is for shifting on SAT1 in the shift-count subroutine. For a shift operation, AR and MQ are shifted left if bit 18 of the instruction word is 0, right if bit 18 is 1. Note that even though the program may request the shift of a single accumulator, the logic shifts both AR and MQ, the latter being empty. The level gates for other SC-controlled shifts are supplied by subroutine control flip-flops. The first part of a character operation requires an MQ left shift; this is followed in the second part by a shift left of both in the deposit sequence or an AR shift right in the load sequence. Floating addition requires a right shift of both registers. The upper set of gates allows pulses from the subroutine time chains to trigger shifts. The regular division shift is both registers left at DST14A, but two other divide pulses produce right shifts of AR alone and one left shifts MQ alone. Similarly the multiplication process shifts both to the right at MST2, but a final shift at the end of the subroutine moves MQ alone to the right. The normalize return begins by shifting both registers right in case there has been overflow in calculations with the fractions, and then the regular normalizing process shifts both to the left. In floating division, a 2-bit overflow is possible; so following the divide subroutine, the final pulse in the floating-divide instruction sequence (FDT1) shifts both registers right to supplement the single shift that begins the normalize return. The effects at the register extremities for all of these shifts are controlled

by the special shift inputs shown in Figure 6-7 and described with the gating for the registers (6.2a, 6.3). The exact configurations for all shifts are shown in block diagram form in Figure 6-7 and each shift is shown at the appropriate place in the flow charts.

The remaining nets in Figure 6-17 are mostly for gates that collect subroutine time pulses for transmission to the register gating. In some cases, the pulse inputs are gated by levels, particularly in the upper right nets that detect overflow. All connections from subroutine time chains, other than for functions listed at the top of the figure, are made directly to the control logic for the registers. All subroutines pulses that trigger the AR negate or add subroutine, the switch of MB and MQ, or the AR complement are routed through the subroutine interface. For the other functions, which include entry to the AR subtract subroutine, the transfer of MQ to MB, the switch of MB and AR, and overflow, most pulses are routed through the interface but some are connected directly to the register gating. For example, the net in the upper right of Figure 6-17 handles overflow for multiplication, division, and all floating-point operations, but overflow in an arithmetic shift operation is handled by a net included with the flag logic (6.2e).

Except for one special case, all SBR flip-flops and control flip-flops in the subroutine logic are cleared at the beginning of every main sequence. For this purpose, the master clear triggers several pulse amplifiers to drive additional clear lines. Two of these with prefix MP are in the lower right of Figure 6-21, a third with prefix DS is in the upper right of Figure 6-26.

6.6 DATA SUBROUTINE INSTRUCTIONS

Three types of data transmission instructions switch to subroutines for their execution. Flow charts for all three types are in Figure 4-9. The block transfer moves an entire block of words from one area in memory to another. The character operations handle single characters smaller than a word and can insert a character into a word in memory or retrieve a character from a word without affecting the rest of it. Shift operations move the bits of a word or pair of words to the left or right. There are several shift configurations differentiated mainly by the effects

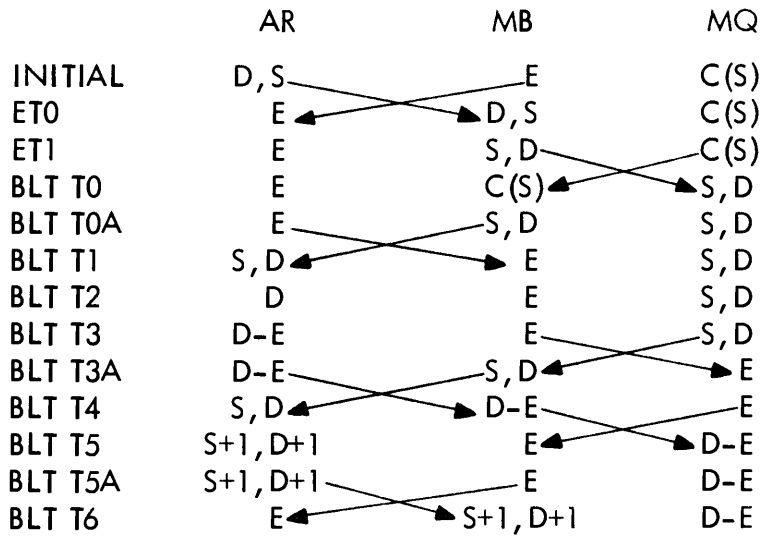
of the shift on the register extremities. The last group may be viewed as logical operations rather than data transmission; the arithmetic shift is equivalent to multiplying the word by a power of 2.

a Block Transfer

A flow chart of the block transfer instruction is at the right in Figure 4-9 and the time chain is in Figure 6-18. The left and right halves, respectively, of the accumulator addressed by the instruction provide source and destination addresses S and D. The first fetch cycle retrieves a word from location S. The subroutine then stores the word in location D, increments both S and D, and returns to the fetch cycle to retrieve and store a second word according to the incremented source and destination addresses. The entire sequence is iterated until D equals the effective address E.

The first fetch cycle retrieves AC, swaps its halves so that S is available to MA, and fetches C(S). At the beginning of the execute cycle, AR contains (D,S), MB contains (0,E), and MQ contains C(S). The first execute pulse switches MB and AR to save E and bring (D,S) to MB. ET1 then clears MA and swaps the MB halves so ET3 can transfer D to MA. ET3 also triggers the first subroutine pulse BLT T0 (Figure 6-18, upper left) which switches MB and MQ to save (S,D) in MQ and make C(S) available to memory from MB. It also requests a memory write to store C(S) in D.

Upon receipt of the memory return, the BLT time chain transfers MQ to MB so that the addresses are now in both registers. The next pulse then switches MB and AR so that E is now in MB and (S,D) in AR. BLT T2 clears ARLT and the next pulse calls the subroutine to subtract E from D. Following the AR subroutine, BLT T3A places E in MQ and returns (S,D) to MB. BLT T4 then moves D-E to MB and the two addresses to AR. The next pulse moves the subtraction result to MQ, bringing E to MB, and triggers the subroutine that adds 1 to both halves of AR, incrementing both addresses. Upon the return BLT T6 saves E in AR, moves the new addresses to MB and reenters the fetch cycle at FT1A. This is the point just following the retrieval of an accumulator, so the processor repeats the entire procedure using the incremented addresses in MB as though they had just been retrieved from AC. For convenience, the following table shows the contents of AR, MB, and MQ following each pulse in the sequence (or following a subroutine called by the pulse).



Since E is initially greater than D, the result of the address subtraction is negative until the cycle following that in which the indexing of D makes it equal to E; then the result is 0 so the sign is positive. The transfers at BLT T4 and BLT T5 move the result to MQ, and BLT T5A tests MQ0 (D4, B8) to determine whether the block is complete. If MQ0 is 0 at this time, the program counter is incremented (the normal program counting at ET1 is inhibited throughout the block transfer) and the subroutine returns to ET10 instead of going on to BLT T6. There are no operations in the store cycle and the processor goes on to the next instruction.

Since a block transfer may use many main sequences, the subroutine includes provision for strobing the priority interrupt system at every BLT T4. If a PI request is generated, the level BLT DONE is asserted even though MQ0 may not be 0 (if it is, the subroutine terminates in the usual manner). This prevents the final MB-AR switch at BLT T6, so the incremented addresses are still in AR and BLT T5A goes directly to ET10. Since BLT LAST is negated (D6), there is no store-AC inhibit and the current addresses are stored in the accumulator in place of the original ones. Following the store cycle, the processor returns to the instruction cycle ostensibly to repeat the same instruction but it is interrupted by the PI request. After all requests have been serviced, the program returns to the interrupted block transfer, fetches the new addresses from AC, and begins where it had previously left off.

b Character Operations

There are five instructions in the character operation group, four of which require two main sequences for execution: the first part fetches and if necessary increments the pointer, the

second handles the character designated by the incremented pointer. The flow chart for all character operations occupies the left half of Figure 4-9 and the logic for the two parts is shown in Figures 6-19 and 6-20. The top of the flow chart lists the different instructions, the main control levels governing their execution, and the configuration of the pointer. The first fetch cycle fetches the pointer according to the effective address calculated from the instruction; the address cycle in the second part calculates the effective address of the operand from the I, X, and Y portions of the pointer. Within the operand, the character is defined by the P and S portions of the pointer: S specifies its size; P specifies its position as the number of bits remaining to the right of the character in the word. Two of the instructions merely fetch the pointer in the first part and then one enters the load sequence in the second part, the other the deposit sequence. The load sequence retrieves a character of size S from position P in the word in location E and loads it right justified into AC. The deposit sequence fetches a character of S bits from the right end of AC and inserts it at position P in C(E). Two other instructions increment the pointer in the first part and use this new pointer in the load or deposit sequence that follows. The last instruction merely fetches and increments the pointer and then returns to the instruction cycle to continue the program, skipping the second part.

The major control levels for the first part are derived from the instruction command levels and two control flip-flops CHF5 and CHF7, both of which are set by the final pulse in the first part. The first flip-flop distinguishes the two parts, the second compensates for the fact that a PI request can interrupt the program between the two parts. The level CH INC is asserted during the first part of any instruction that increments the pointer (Figure 6-19A2). This level is ANDed with CHF7(0) to generate CH INC OP, whose assertion indicates that the first part must actually increment the pointer. If the pointer is incremented and a priority interrupt occurs, the program must not reincrement upon repeating the first part after returning to the interrupted sequence (the first part is repeated only to fetch the pointer that was lost). Unlike most control flip-flops, which are cleared by the master clear, the state of CHF7 (D3) must remain from the first to the second main sequence and is cleared by the master start via the flag clear (6.2e). When it is set by CHT9 at the end of the first part, the sequence returns to the address cycle at which point a priority interrupt may occur. If an interrupt merely executes a block IOT, CHF7 remains set for the return. However if there is a jump to a subroutine, the JSR saves CHF7 with the flags and clears it so that it may be used by the break

routine. When the routine is complete, the restoring JRST again clears CHF7 via the flag clear and restores its original state from MB4 at the same time that it restores the flags. Then when the program repeats the first part, CHF7(1) inhibits CH INC OP and instead causes CH INC to assert CH ∿ INC OP (C5). This level is also asserted in the first part of the two instructions that do not increment the pointer. Finally, CHF7 is cleared by either sequence in the second part.

The first part fetches the pointer from location E: CH ∿ INC OP allows memory to rewrite the pointer, but CH INC OP requests a fetch and pause so that the incremented pointer may subsequently be deposited. Since there must be only one program count for each instruction, PC+1 is inhibited by all but the IBP (CAO), which uses only the first part. If there is no pointer incrementing, ET0 triggers the subroutine time chain at CHT6 (Figure 6-19B4); otherwise it starts at CHT1 (upper left). The first pulse transfers the pointer to AR and sets CHF1, enabling the zeros of MB0-5 as data inputs to SC so that the partial add at the next pulse loads -P-1 (i.e. ∿ P) into SC. CHT3 then sets CHF2 and calls SC add. CHF1 is not cleared until the first pulse in the SC add chain to give the CHT2 partial add a little more time. CHF2(1) then enables the ones from MB6-11 for the partial add in the SC subroutine so that upon the return SC contains -P+S-1. If there are not enough bits left for another character; i.e., if $S \geq P + 1$, the result in SC has a positive sign and the instruction must go on to the next location for the character. Thus if SC0 is 0, the chain continues to CHT4, which clears SC and calls an AR subroutine to index the address portion Y of the pointer to the next location. Upon completion, CHT4A loads -37 (i.e., -P-1 for a character of 0 size) into SC and returns to CHT3 to call SC add again; this time the addition of S generates the complement of the position portion of the pointer for the new location. If SC0 is 1 at the junction following CHT3A, the chain skips to CHT5 (B3), which complements SC so that it now contains P-S, the position of the next character in the same location or the first character in the next location. Actually CHT4A loads -229 into SC so that an S larger than 36 cannot put the processor into a loop. The result is then interpreted mod 64 so the correct position results. Thus if the pointer must go to the next location and S is larger than 36, the new P is 100 -S rather than 36 -S.

The chain then continues to CHT6, which is the starting point for any nonincrementing instruction. CHT6 again sets CHF2 but there is no subroutine call: CHF2(1) merely enables the MB6-11(1) data inputs to SC so that the partial addition at CHT7 transfers S into it. For

CH ~ INC OP, the chain then skips to CHT8B. CH INC OP uses the CHT6 and CHT7 operations already mentioned because S is still available from the old pointer in MB; additional events specifically for CH INC OP are that CHT6 inserts the new P into the pointer by transferring it from SC3-8 into AR0-5 (dropping SC0-2 means SC mod 64 is transferred) and clears SC. CHT7 then moves the new pointer to MB after which CHT8 restarts the read/write memory cycle to deposit it. The return triggers CHT8B which clears both CHF6 and CHF2, transfers the new P from MB0-5 to FE and complements S in SC. If incrementing the pointer is the only operation required by the instruction, the subroutine terminates here and the sequence returns to ST7 for a new instruction cycle. For any character operation other than an IBP, CHT8B calls the shift-count (C2) which shifts MQ left S places loading ones in at MQ35. Upon the return, CHT8A clears SC and IR13-17, the latter in preparation for receiving the I and X portions of the pointer in the address cycle following the first part. Then CHT9 transfers P from FE to SC, sets CHF5 so that the next execute cycle will select the second part, and sets CHF7 in case there is a PI request at the beginning of the address cycle to which the sequence then returns.

The two chains for the second part are shown at the top in Figure 6-20. Both sequences start with MB containing the word retrieved according to the effective address of the pointer, and MQ containing a word made up of ones in the last S places at the right and zeros elsewhere. For the two load instructions, AR contains E. The initial gates include FC(E) as there will be no subsequent storage in E, and ET0 triggers the first pulse in the chain (A2). LCT0 moves the data word to AR and the mask to MB, complements P in SC, and calls the shift-count subroutine. The character is then right justified by right shifting AR P places. Following this, LCT0A transfers zeros from the mask into AR thus clearing all of AR except that part containing the desired character. The pulse also clears CHF7 and returns to ET10.

The two deposit instructions request a fetch and pause, and ET0 enters the deposit sequence (A4). DCT0 complements P in SC and calls a shift-count that moves AR and MQ left P places loading zeros in at the right in both registers. Upon subroutine completion, AR contains the character in the appropriate position and the ones in the mask are in the same position. The pair of pulses triggered by the return transfers the mask to MB, the ones from MB to MQ, and complements AR. Thus MB contains the mask, whereas MQ contains the data word other than in the character position which contains all ones. DCT1 then transfers zeros from the mask to AR clearing it other than in tne character position which contains the complement of the

character. At the same time, the data word with ones in the character position is moved back to MB. The next pulse again complements AR so that it contains ones outside of the character, and DCT3 then inserts the character into the appropriate position in the data word by transferring zeros from the character (all other bits are ones) into the all ones portion of MB. This pulse also clears CHF7 and returns to ET10. The subsequent store cycle restarts the waiting memory cycle to deposit the data word in E.

If the program specifies a size greater than 36, the character is at most the entire word. For $P \geq 36$, no character is processed. If both P and S are less than 36 but $P + S > 36$, a character of size 36 - P is loaded from position P or the right 36 - P bits of the character are deposited in position P.

c Shift Operations

The lower part of Figure 6-20 shows the logic governing the shift operations and the flowchart for them occupies the right portion of Figure 4-9. The three combined instructions generate the level SHAC2 (B6) which causes the main sequence to fetch and store a second accumulator. The direction of the shift is specified by bit 18 of the instruction word (0 left, 1 right), and the number of places to be shifted is specified by bits 28 to 35.

At the beginning of the execute cycle, AR contains AC; and for a combined shift, MQ contains a second AC. ET0 transfers the complement of MB18, 28-35 into SC. Since left shifting is considered to be positive and right shifting negative, it is assumed that if bit 18 is 1, bits 28 to 35 contain the 2's complement of the number of shifts desired. Thus if MB18 is 0, SC is already correct and contains the complement of a positive number; however if MB18 is 1, SC contains a positive number one less than the number of shifts. Thus MB18(1) gates ET1 to trigger SHT0 (B5), which adds one to SC. Then ET3 starts the subroutine chain for all instructions and the first pulse SHT1 complements SC on the condition MB18(1) so that SC now contains the correct complement. SHT1 also calls the shift-count, which counts SC up to all ones and at each count shifts AR and MQ left or right according to the state of MB18. The shift connections to the registers are made through the subroutine interface (6.5), and the special shift inputs that control the shift actions at the register extremities are shown in Figure 6-7 and described with the AR and MQ gating (6.2a, 6.3). Block diagrams below the flowchart show the configurations for all twelve types of shift.

Since an arithmetic shift multiplies fixed-point numbers by powers of 2, an overflow condition is included in case significant bits are lost in a positive shift. In a single- or double-left arithmetic shift, the overflow flag is set if a 1 is shifted out of AR1 in a positive number, or a 0 in a negative number (6.2e). The return from the shift-count triggers SHT1A which retruns to the main sequence at ET10.

6.7 ARITHMETIC INSTRUCTIONS

This section describes fixed multiply and the floating-point instructions. Each of these instructions goes from the execute cycle to a special sequence which may or may not call an arithmetic subroutine. Fixed add and subtract are both performed within the execute cycle and are described with the arithmetic register (6.2). Fixed divide enters directly into the divide subroutine from ET0 and is described with that subroutine in 6.8c.

a Fixed Multiply

A flow chart of the two fixed-multiply instructions is in the left part of Figure 4-8; Figure 6-21 shows the special time chain for them. Both integral and fractional multiplication use AC and either C(E) or E itself as operands, but the product in the latter case is a double-length fraction, whereas in the former it is assumed that the desired integer is in the low-order half of the double-length product. Both instructions enter the special sequence at ET0 by triggering MPT0 (Figure 6-21, upper left). This pulse sets the appropriate bits in SC to count 35 steps and sets MPF2 if both operands are negative. It also calls the multiply subroutine (6.8b) and waits until MST6 returns to MPT0A. If both operands are negative and the result is also negative, MPT0A sets the overflow flag. This can occur only if -1 is multiplied by -1, whose answer +1 overflows generating the representation for -1. If integral multiplication is being performed (IR6(0)) and the result is negative, MPT0A complements the high-order half in AR. At this point, the fractional process is complete and the sequence returns to ET10 via NRT6, the final pulse in the normalize return subroutine. In the store cycle, the low-order half is stored in a second accumulator for all but the memory mode wherein no accumulator is stored. For integral multiplication, MPT0A continues the chain to MPT1, which transfers the resulting integer in the low-order half from MQ to MB and sets the overflow flag if the high-order half is not clear. MPT2 then transfers the result to AR so that the sequence can make use of the standard transfer and store functions following the return to ET10 via NRT6.

b Floating Scale

This instruction allows the program to change the exponent of a floating-point number without affecting the fractional part. The number in AC is multiplied by 2^y where y is the number contained in bits 28 to 35 of the effective address. This number is interpreted as positive or negative in 2's complement notation as the sign, bit 18, is 0 or 1. The flowchart for the instruction occupies the left quarter of Figure 4-10, and the time chain is shown in the lower right of Figure 6-19.

The first pulse in the execute cycle transfers the complement of MB18, 28-35 to SC. The next two pulses then adjust SC according to the sign of the number in AR: if positive, ET1 complements SC; if negative, ET3 adds one to SC by triggering FST1. In either case, ET3 triggers FST0, which calls the SC add subroutine. During the subroutine, the 1 state of FSF1 enables the AR0-8(1) data inputs to SC. The return pulse FST0A transfers the new exponent from SC1-8 to the exponent part of AR; and if the signs of AR and SC are different, it sets the overflow flag.

To see that the above sequence of events produces the correct exponent and properly detects overflow or underflow, consider the various cases keeping in mind that the floating-point exponents from -128 to +127 are represented by the numbers 0 to 255 and that the scale factor in E is in 2's complement notation. Let x and y be the absolute values, respectively, of the exponent part of AR and the scale factor in MB. Thus if AR is positive, AR1-8 contains x; otherwise $\sim$ x, i.e., 255 - x. On the other hand if MB is positive, MB28-35 contains y; otherwise the 2's complement, i.e., 256 - y. At each step, C(SC) is a function of the signs of AR and MB as follows:

	AR+, MB+	AR-, MB+	AR+, MB-	AR-, MB-
ET0	-[255 - y]	-[255 - y]	+[y - 1]	+[y - 1]
ET1	+[y]	-[255 - y]	-[256 - y]	+[y - 1]
ET3	+[y]	-[256 - y]	-[256 - y]	+[y]
SC ADD	+[x + y]	-[255 - (x + y)]	+[x - y]	-[255 - (x - y)]

Hence with no overflow or underflow, SC and AR have the same sign and SC contains the proper representation of the new exponent. However, if in the first two columns $x + y > 255$ (overflow)or in the last two $x - y < 0$ (underflow), AR and SC have opposite signs.

c Floating Add-Subtract

Both of these instructions use the floating add time chain shown in the upper half of Figure 6-22; the flow chart occupies the center portion of Figure 4-10. For floating add, ET0 triggers the special sequence (Figure 6-22A1); but for floating subtract, ET0 switches MB and AR so that the subtrahend is then in AR, and ET3 calls the negate subroutine. Since the number to be subtracted has now been replaced by its negative, the operands may instead be added and ET4 triggers the floating add sequence.

If the signs of the operands are the same, FAT0 complements SC to all ones. It also sets FAF1 enabling the MB0-8(1) data inputs to SC so that the partial addition at FAT1 produces a 1 transfer if the signs are different, but transfers the complement of the MB sign and exponent if the signs are the same. In either case, the sign of SC is opposite that of AR. FAT1 also triggers SC add, and shortly after, FAT1B clears FAF1 and sets FAF2 (A3) to enable the AR0-8(1) data inputs to SC. Since the signs of AR and SC are different, at least one exponent is represented by its complement, and hence the result of the addition in SC is a 1's complement negative of the difference between the exponents unless there is overflow. Let x and y be the absolute values of the exponents of the numbers in AR and MB. The possible signs, exponent representations, and results are as follows:

AR, MB signs	AR0-8	SC Before	SC After	Result
AR+, MB+	+[x]	-[255 - y]	-[255 - y + x]	$y \geq x$: signs $\neq$
AR+, MB-	+[x]	-[255 - y]	-[255 - y + x]	$y < x$: signs $=$
AR-, MB-	-[255 - x]	+[y]	-[255 + y - x]	$y > x$: signs $\neq$
AR-, MB+	-[255 - x]	+[y]	-[255 + y - x]	$y \leq x$: signs $=$

Hence, if the MB exponent is greater than the AR exponent, the AR and SC signs differ after the addition. If the AR exponent is greater, the signs are the same and FAT1A switches MB and AR because the number with the smaller exponent is the one that is shifted. If the exponents

are equal, the signs may or may not be the same but it matters not whether the transfer takes place. Since the result in SC is a 1's complement, no further action on it is necessary if it is negative, and in this case FAT1A jumps directly to FAT4 (A6). However if overflow has produced a positive result, the number in SC is one less than the difference between the exponents (since a 2's complement addition was performed on numbers in 1's complement notation); in this case FAT1A triggers FAT2 which adds one to SC, and FAT3 complements it in preparation for the shift-count.

If the number with the smaller exponent must be shifted more than 63 places (a condition represented by at least one 0 in SC0-2), the addition can affect neither the fraction nor the low-order part so FAT6 clears AR (A8). However if fewer than 64 shifts are required, FAT5 ensures that all bits to the left of the fraction MSB are of no significance by loading the sign into AR1-8, sets FAF3(B4) to generate the correct arithmetic shift inputs to the registers extremities (6.2a, 6.3), and calls the shift-count subroutine. After the number has been right shifted in AR and MQ so that its bits correctly match the MB bits in order of magnitude, the return triggers FAT5A which follows directly from FAT6 if there is no shifting. FAT5A clears SC and again sets FAF1 to enable the MB0-8(1) data inputs. Then FAT7 changes SC to all ones if MB is negative, and the partial add at FAT8 loads SC with the MB sign and exponent or their complements depending upon the state of SC. Thus SC always receives a positive sign and the absolute value of the exponent. The next pulse in the chain then nullifies the exponent portion of MB, and FAT9 calls AR add. If AR was cleared at FAT6, the addition merely transfers MB to it. The return triggers FAT10 which clears those control flip-flops that are still set and enters the normalize return subroutine. The return via NRT6 is directly to ET10 for storage of the result.

d Floating Multiply and Divide

These two simple sequences do little more than call the three subroutines necessary for the execution of floating-point multiplication and division. The flowcharts are at the right in Figure 4-10 and the logic is shown in the lower half of Figure 6-22. Both chains are triggered at ET0 and the first pulse in each calls the exponent calculate subroutine. The return from FPT4 places in SC the complement of the number of steps required (27 for multiply, 30 for divide) and enters the appropriate subroutine. The return from multiply at MST7 triggers

FMT0B, which transfers the calculated exponent from FE to SC, sets NRF2 to set up the normalize shift gates, and enters normalize return. NRT6 returns directly to ET10 for storage of the result.

The divide subroutine first tests that division can be performed. If the divisor is less than or equal to half the dividend, the sequence sets the overflow flag and jumps directly from DST13 to ST7—the only normalized number that fails to satisfy this condition is a zero divisor. If the division is executed, DST21A returns to FDT0B, which transfers the calculated exponent from FE to SC and sets NRF2 to set up the normalize shift gates but does not enter the normalize return subroutine. This subroutine can compensate for only one bit overflow whereas floating divide, by doing 30 steps, deliberately generates two extra quotient bits, one for possible overflow, the other for rounding. The divide instruction sequence includes an extra time pulse, FDT1, which shifts AR and MQ right to compensate for the rounding bit. It then enters normalize return and NRT6 returns directly to ET10 for storage of the result.

6.8 ARITHMETIC SUBROUTINES

Besides the simple AR and SC subroutines, there are four arithmetic subroutines that are called by more than one arithmetic operation. Floating multiplication and floating division both begin with a subroutine that calculates the exponent. The multiply subroutine is used by both fixed and floating multiplication, the divide subroutine by both fixed and floating division. All floating-point instructions except floating scale call the normalize return subroutine to normalize the result of their arithmetic computations.

a Exponent Calculate

Floating multiply and divide call a subroutine to calculate the exponent before beginning operations on the fractions. The flow chart for exponent calculate is at the left in Figure 4-11 and the logic for the time chain is shown in Figure 6-23. The nets at the right in the logic drawing generate several levels necessary for execution of the subroutine. These are exclusive OR and equivalence functions of the operand signs and FMF1, the last term providing a distinction between multiplication and division. In multiplication the exponents are added, whereas in division the exponent of the divisor is subtracted from that of the dividend. Since

FMF1 is the SBR for the return to multiplication from the exponent calculation, it is 1 in multiplication but 0 in division; therefore, the sign functions for multiplication are exactly the opposite of those for division.

The first time pulse in either multiply or divide triggers the floating-point time chain (Figure 6-23, upper left). The first floating-point pulse sets SC1 thus loading 128 into SC. The next pulse complements SC if the sign of AR is different from the state of FMF1, i.e., the complement occurs if AR is positive in multiplication or negative in division. FPT1 also enables the AR0-8(1) data inputs to SC and calls the SC add subroutine. At the return FPT1A complements SC if the exclusive OR of AR0, MB0, and FMF1 is false; otherwise it adds one to SC. The latter action is triggered by FPT2, which is generated only if the appropriate condition holds (upper right); the gating for the complement is included in the complement net for SC. The next pulse then calls another SC addition, this time using MB0-8(1) as the SC data inputs enabled by FPF2(1). After this addition, the result is correct for two of the four cases; for the other two, MB0 and FMF1 the same, FPT1B complements SC. Then FPT3 transfers the calculated exponent to FE, clears SC, and nullifies the exponent portions of both MB and AR by loading the appropriate sign into bits 1 to 8. FPT4 returns to the interrupted special sequence.

First, consider multiplication. FMF1 is set, thus at FPT1 the complement occurs if AR0 is 0; then at FPT1A if the signs of AR and MB are not equal, SC is again complemented, otherwise it is indexed. Finally at FPT1B, SC is complemented if MB is negative. On the other hand, FMF1 is 0 for division, so the complement at FPT1 occurs if AR is negative; at FPT1A complementing occurs if AR0 and MB0 are the same, and indexing otherwise. Finally, FPT1B complements SC if MB is positive.

To see that these operations give the correct result, let x and y be the true exponents of AR and MB. Since excess-128 code is used, the exponent portion of AR is $x + 128$ if the number is positive; but if negative, it is $255 - (x + 128) = 127 - x$. Since the result must also be in excess-128 code, the sum of the exponents must be reduced by 128 for multiplication; whereas, in division the difference must be increased by the same factor. For multiplication, the above sequence of events operates in SC as follows (the table items show the sign of SC and the contents of SC1-8 as a positive number):

	AR+, MB+	AR+, MB-	AR-, MB+	AR-, MB-
FPT1	-[127]	-[127]	+[128]	+[128]
SC ADD	+[x - 1]	+[x - 1]	-[255 - x]	-[255 - x]
FPT1A	+[x]	-[256 - x]	+[x]	-[256 - x]
SC ADD	+[128 + x + y]	-[127 - x - y]	+[128 + x + y]	-[127 - x - y]
FPT1B		+[128 + x + y]		+[128 + x + y]

For every case, the result in SC is the correct exponent for the product (but expressed in positive form), unless there is overflow, as indicated by a negative result (SC0 = 1). The exponent remains in positive form during the multiply subroutine, and the subsequent normalize return subroutine checks for overflow and puts the exponent into correct form. For division, the sequence of events is:

	AR+, MB+	AR+, MB-	AR-, MB+	AR-, MB-
FPT1	+[128]	+[128]	-[127]	-[127]
SC ADD	-[x]	-[x]	+[254 - x]	+[254 - x]
FPT1A	+[255 - x]	-[1 + x]	+[255 - x]	-[1 + x]
SC ADD	-[127 - x + y]	+[128 + x - y]	-[127 - x + y]	+[128 + x - y]
FPT1B	+[128 + x - y]		+[128 + x - y]	

which again gives the correct result in positive form.

b Multiply

A single subroutine handles multiplication for both fixed point and floating point; the two types differ only in the number of steps—35 for fixed point, 27 for floating point. The subroutine flow chart occupies the center portion of Figure 4-11; Figure 6-24 shows the time chain. A pulse in the fixed- or floating-multiply sequence loads the complement of the number of steps into SC, sets an SBR and calls the subroutine. To control the AR and MQ extremities in multiply shifting, the two SBR flip-flops are ORed by the net at the right in Figure 6-24.

For fixed point, the subroutine time chain is triggered by MPT0, which is equivalent to ET0; for floating point, the entering pulse is FMT0A, which follows the exponent calculation. Either pulse triggers MST1 (upper left), which moves the multiplier from MB to MQ and the

multiplicand from AR to MB. AR is cleared shortly thereafter. Form MST1, the sequence may continue to either MST2, MST3, or MST4 depending upon the relationship between bits 35 and 36 of MQ, but every step includes the incrementing of SC and the shifting of AR and MQ right one place. Initially, MQ35 contains the LSB of the multiplier and MQ36 is 0. On each successive step, MQ35 contains the next more significant bit of the multiplier and MQ36 contains the bit MQ35 held on the previous step. Arithmetic operations are performed whenever there is a transition from 1 to 0 or vice versa in a pair of multiplier bits, rather than on the basis of a particular bit being 1 or 0. At the 3-way decision following MST1, there is no arithmetic action if the two bits are equal and in this case the chain goes to MST2. If there is a transition from 1 to 0 in the bit pair, the chain goes to MST3, which calls the AR add subroutine. For a transition in the opposite direction, MST4 calls AR subtract. Either return triggers MST3A, which goes to MST2. Thus MST2 appears in every loop whether any AR subroutine is called or not. This pulse increments SC and shifts AR, MQ right one place. It then returns the sequence to the 3-way decision to check the next bit pair (which has just been shifted into MQ35, 36).

At the 3-way decision, the equality branch has the additional condition that the step count is not complete, i.e., SC does not contain 777. In the final shift, MQ36 receives the multiplier MSB and MQ35, the sign. If these two bits are equal, MST2 leaves the loop, jumping to MST5 (lower left). If they are not equal, MST2 returns to the 3-way decision, only two of whose branches are now open—those which necessarily call AR subroutines. Since the step count is complete, the return jumps directly from MST3A to MST5. This pulse clears SC, shifts the low-order half of the product in MQ to the right, and makes its sign equal to that of the high-order half in AR. MST6 then provides the return to the appropriate instruction sequence.

The multiplication sequence is as follows: Initially, MB contains the multiplicand, MQ the multiplier, and AR is clear. AR and MQ are connected so that the low-order bits of the product are shifted into MQ as the multiplier is shifted out. At each step, the current bit of the multiplier is available at MQ35, and the effect of the multiplicand in MB on the partial sum in AR is one binary order of magnitude greater than in the preceding step because the partial sum was right shifted. Thus MB can be combined directly with AR. Since MQ36 is initially 0, the sequence shifts without calling an AR subroutine until a 1 is shifted into MB35.

At this transition, the sequence jumps to MST4 to subtract the multiplicand from AR, which is clear. The shifting then continues until the next transition (which will be from 1 to 0), at which time the sequence goes to MST3 to add the multiplicand to the previous partial sum. The process continues in this way, subtracting the multiplicand at every transition from 0 to 1, adding at every transition from 1 to 0. At the end, AR contains the correct sign and the high-order half of the double-length product. The low-order half is in MQ0-34 with the sign of the multiplier in MQ35. Thus, there is an additional right shift of MQ to move the correct sign from AR0 to MQ0 and to place the low-order fraction in the MQ magnitude positions.

To see that this procedure results in a correct product, consider the positive binary integer:

1 0 0 1 1 1 0 1 1

8 7 6 5 4 3 2 1 0

(The decimal digits below the binary digits are the powers of 2 corresponding to the bit positions.) This number is obviously equal to:

$$\begin{array}{rr} & 100000000 \\ + & 111000 \\ + & 11 \end{array}$$

Now an n-bit string of ones whose rightmost bit corresponds to 2^k is equal to $2^{k+n} - 2^k$, or equivalently $2^k(2^n - 2^0)$; i.e., $2^n - 2^0$ is a string of n ones and the 2^k shifts the string left k places. Thus:

$$\begin{array}{rcccc} 100000000 & = & 2^{8+1} - 2^8 & = & 2^9 - 2^8 \\ 111000 & = & 2^{3+3} - 2^3 & = & 2^6 - 2^3 \\ 11 & = & 2^{0+2} - 2^0 & = & 2^2 - 2^0 \\ \hline 100111011 & & = & & 2^9 - 2^8 + 2^6 - 2^3 + 2^2 - 2^0 \end{array}$$

In this last representation, each power of 2 that is subtracted corresponds to a transition from 0 to 1 (in the direction of increasing significance), whereas each that is added corresponds to the opposite transition. The largest term corresponds to the transition to the sign bit, which is 0 for a positive number. The multiplication algorithm in PDP-6 interprets the multiplier in this manner, alternately subtracting the multiplicand from the partial sum and adding it to the partial sum in the order-of-magnitude positions corresponding to the transitions. If a multiplier

of the same magnitude were negative, it would have the form:

1 0 1 1 0 0 0 1 0 1

- 8 7 6 5 4 3 2 1 0

in which the extra bit at the left represents the sign. The number is now equivalent to:

$$-2^9+2^8-2^6+2^3-2^2+2^1-2^0$$

wherein opposite signs correspond to opposite transitions. The algorithm may thus use exactly the same sequence for a negative multiplier: this time the subtraction of greatest magnitude is detected by the transition to the sign bit, which is now 1.

c Divide

The time chain for the divide subroutine is on two logic drawings, Figures 6-25 and 6-26; Figure 4-12 is the flow chart. Entry is at ET0 in fixed point, at FDT0A in floating-point. Either pulse loads SC with the complement of the number of steps required—30 for floating-point, 36 for fixed-point—through pulse gates at individual bits. For this purpose, ET0 gated by IR DIV generates DS DIV T0 (Figure 6-25A8). Although the subroutine is triggered only by ET0 or FDT0A, there are nevertheless six entries, depending upon the sign of the dividend in AR and upon whether the fixed division is integral or fractional. To make the latter distinction, the fixed-point command level is further decoded into a pair of levels according to the state of IR6 (B1).

For a negative dividend, entry is at DST0 (upper left) for integral or floating-point operations. The first subroutine pulse sets DSF7 (D6) to remember that the dividend was originally negative, and calls the AR negate subroutine. Following negation, the sequence jumps to DST10 for floating-point but continues to DST1 for division of integers. This is also the entry for integral division with a positive dividend. Integral division uses a double-length dividend just as does fixed-point fractional division, but only one accumulator is fetched and it is used for the low-order half. Since the fetch cycle brings AC to AR, DST1 moves the divisor from MB to MQ and the significant half of the dividend from AR to MB. DST2 then switches MB and MQ, placing the dividend in low-order position, and clears the high-order half in AR. The sequence then jumps to DST10 for computations.

The only other entry earlier than DST10 is for a negative fixed-point fraction, wherein ET0 enters at DST3 (A6). For all other entries, the dividend is either one word in length or is already positive. The sequence starting with DST3 is necessary to change a double-length negative dividend into positive form. DST3 sets DSF7 because the dividend is negative and interchanges MB and MQ. DST4 then switches MB and AR so the low-order half is now in AR, and DST5 calls AR negate. After the return (B3), the sequence branches depending on whether AR is clear. The 2's complement of a double-length number is formed by complementing the entire double word and adding one into the LSB. A carry into the high-order half is necessary if the 2's complement of the low-order half is null (the complement was all ones before the LSB addition). Thus if AR is clear, the sequence goes on to DST8, 9: this reverses the preceding interchanges, returning all words to their original positions, and DST9 calls AR negate to form the 2's complement of the high-order half (i.e., to complement it and then add one as required). If AR is not clear, the sequence goes instead to DST6, 7 which also returns all words to their original positions but merely complements AR.

The computational part of the subroutine begins at DST10, which is triggered through the net at the right in Figure 6-25. This pulse begins the subroutine for a positive dividend in floating point or fractional fixed point. Otherwise, it follows from one of the preliminary sequences discussed above. Entry is direct from DST2 or DST7; from DST0A, it is conditioned by ∿ DSDIVI, which in this case represents floating point; and if DST9 calls AR negate, entry is made at the subroutine return. For a floating division, DST10 triggers DST10A (lower left) which right shifts AR, moving the LSB of the high-order half into MQ0, so the double-length dividend now has the sign bit in both AR0 and AR1 and the 70-bit magnitude in AR2-35 and MQ0-35. This action closes up the hole between the two halves of the dividend but it also divides the dividend by 2. For fixed division the sequence goes to DST10B, which left shifts MQ, closing the hole but leaving the magnitude the same. The shift brings the complement of AR0 into MQ35 because of the division shift connections, but this is of no significance.

The next step in the process is to compare the divisor with the dividend to determine whether the division can be performed. Thus, the next pulse is either DST11 or DST12 depending upon whether the divisor is negative or positive. The former pulse calls AR add; the latter, AR subtract. Since the dividend is positive, the result of the computation is also positive if the magnitude of the divisor is less than or equal to that of the number in AR. There are actually

three different conditions being tested here for the three types of division. For a fixed integer, AR is clear and the result is positive only for a 0 divisor; if the divisor has anything of significance in it, the quotient cannot possibly be greater than $2^{35}-1$ which does not overflow. For a fixed fraction, the divisor is actually subtracted from the dividend and no overflow is allowed. For a floating fraction, the divisor is subtracted from half the dividend (the preceding right shift having divided it by 2), so there may be one bit overflow but the normalize return automatically compensates for this. Since all floating fractions are normalized, the only case that fails the test is that of a 0 divisor (a dividend of -1 could also fail but this number cannot occur as the result of any floating-point operation). After the return from the AR subroutine, DST11A tests the result. If it is positive, the sequence continues to DST13, which sets the overflow flag and jumps directly to the end of the store cycle. If AR0 is 1, the overflow cannot be greater than allowed in the particular type of division in progress, and DST11A continues the sequence to DST14A (Figure 6-26B7).

The next few pulses comprise the loop that performs the actual division. In division on paper, one subtracts out the divisor the number of times it goes into the dividend, then shifts the dividend one place to the left (or the divisor to the right) and again subtracts out. In binary computations, the divisor goes into the dividend either once or not at all at each step. The loop thus subtracts the divisor to generate a single bit of the quotient. If the subtraction does not overflow, i.e., if the dividend is larger than the divisor, the sign of the result is positive and a 1 is entered into the quotient. If there is overflow, a 0 is entered. To compensate for the overflow, one could add the divisor back into the dividend before going to the next subtraction step. However the PDP-6 algorithm instead shifts first and then adds the divisor back on in the new position. It then continues to shift and add putting zeros into the quotient until the result again becomes positive. This procedure generates the same quotient without ever going back a step.

The following processor operations correspond to the procedure outlined above. If the initial test subtraction produces a negative result, the sequence enters the loop at DST14A, which increments SC and triggers the division shift of AR, MQ to the left. This shift is equivalent to a combined rotation except that MQ35 receives the complement of AR0. The complement of the sign of the result is the next bit of the quotient: if the divisor does not go into the dividend, the result produces a 0 quotient bit. Each loop loads one bit of the quotient into

MQ35, and the low-order half of the dividend is shifted out from MQ as the quotient is shifted in. Following DST14A, the loop checks the previously generated quotient bit in MQ35 to determine what action to take next. If the quotient has received a 1, the divisor is subtracted; for a 0, it is added. The gate is the exclusive OR of MQ35 and MB0 (Figure 6-26, upper left) because the divisor may have either sign. If MQ35 is 1, DST14 subtracts the divisor if positive, or DST15 adds it if negative. A 0 quotient bit reverses the functions. Either DST14 or DST15 sets DSF5 (lower left) to gate the AR subroutine return which goes back to the beginning of the loop (A7). The loop iterates 36 times for fixed division, 30 times for floating. The first bit loaded into MQ35 is not actually part of the quotient but rather the sign bit to the left of the binary point. It must be 0 because the sequence enters the loop only if the divisor is larger than the dividend. The 36 fixed-point iterations generate the sign and 35 magnitude bits of the fractional or integral quotient. Floating-point requires 30 steps for the 27-bit fraction because there may be one bit overflow and an additional quotient bit is computed to allow running.

The test for termination of the process is made not at the end of the loop but in the middle. Thus each step is not shift and subtract, but rather subtract and shift, the first subtraction occurring before the loop. The test for completion follows DST14A. When this pulse generates the final shift and increments SC to 777, the assertion of the SC completion signal prevents the delayed DST14B from continuing in the loop (upper left) and it instead jumps out to DST16.

The remaining operations in the subroutine generate the correct remainder, adjust the signs of quotient and remainder, and place them in correct position. The final shift in the loop places the last quotient bit in MQ, but it leaves the remainder off one place to the left. Therefore DST16 right shifts AR. Since all operations have been performed on positive operands, the remainder should also be positive. If it is already, the sequence jumps directly to DST17A (C4). A negative remainder indicates that too much has been subtracted from the dividend, so either DST17 or DST18 adds the divisor back in (B4). Again this addition may call either AR subroutine, depending upon the MB sign. DST17A then checks the sign of the original dividend: if DSF7 is 0, the sequence jumps to DST19A (lower left); but if the dividend was originally negative, DST19 (A6) calls the AR negate subroutine. Thus at the end, the remainder has the same sign as the dividend. The setting of the SBR for the return from AR negate (lower right) is delayed slightly to prevent the return from the previous subroutine from getting

through both SBR. DST19A then moves the quotient from MQ to MB, and the divisor in the opposite direction. Next DST20 (A6) clears SC and switches the remainder presently in AR with the quotient in MB. The sequence then determines the proper sign for the quotient by checking the exclusive OR of DSF7 and MQ0, which is generated by the net in the lower left. If the operand signs are the same (A5), the result is already in correct form and the sequence jumps to DST21A. If the signs are different, the result should be negative and DST21 calls AR negate. DST21A then moves the remainder to MQ where it is available for storage in a second accumulator. The subroutine returns to ET9 for fixed division, FDT0B for floating division. The reader should note at this point that the remainder is the correct one for a fixed-point instruction; but in the floating case, the remainder in MQ9-35 is correct for a 29-bit quotient which cannot be stored in its present form. For further particulars, refer to the normalize return subroutine (d,p. 6-46).

As an example of the way this algorithm operates, consider a division of 3-bit fixed-point fractions with a dividend of +.100100 and a divisor of +.101. By paper computation, we obtain the quotient this way:

```
        .111
101 ) 100.100
      10 1
      -----
      10 00
       1 01
       ----
         110
         101
         ---
           1
```

Assuming the computer registers to be four bits in length, AR contains 0.100, MQ has 0.100, and MB has 0.101. Before starting the division, MQ must shift left to close the hole, giving MQ = 1.001. The sequence has four steps as follows:

```
        0.100 | 1.001
       -0.101
       -----

        1.111 | 1.001
1  ←    1.111 | 0.010
       +0.101
       -----

        0.100 | 0.010
2  ←    1.000 | 0.101
       -0.101
       -----
```

```
          0.011 | 0.101
3 ←       0.110 | 1.001
         -0.101
          0.001 | 1.011
4 ←       0.011 | 0.111
  →       0.001
```

The quotient is in MQ at the right, the remainder in AR at the left.

d Normalize Return

All floating-point instructions that operate on the fractional parts of the operands end by calling a subroutine to normalize the result. A floating-point number is considered normalized if the MSB of the fraction (bit 9) is opposite in state to the sign bit or if the magnitude of the fraction is 1/2 (the fraction -1/2 has the same magnitude representation as +1/2). The normalize return flow chart occupies the right half of Figure 4-11, and the time chain is shown in Figure 6-27.

Entry into the normalize return chain is immediate from FAT10, the last pulse in the floating add sequence (upper left). The first pulse in the chain, NRT0.5, sets NRF2 (C5), which enables the shift inputs at the AR and MQ extremities for a double-length arithmetic shift (6.2e, 6.3). NRT0 then adds one to the exponent in SC (which is in positive form, see 6.7c, 6.8a) and right shifts AR, MQ to accommodate any overflow that may occurred in the fractional computations. Since a right shift is equivalent to dividing by 2 and indexing the exponent is equivalent to multiplying by 2, the result is unchanged. The next pulse tests for a 0 result. Since the answer has already been shifted right, the net in the center of the figure generates the gating level as the AND function of AR = 0 and MQ1(0). The test can include all of AR because any operation on the fractions that produces a 0 result clears the exponent part of the register. If there is a 0 result, NRT.1 jumps directly to NRT6, the final pulse in the subroutine (B4). For a nonzero result, the chain goes to NRT1, which complements SC (so the exponent is now in negative form) and enters the normalizing loop (A5).

Multiply skips the right shift because it cannot overflow. The last pulse in the multiply sequence, FMT0B, sets NRF2 and after a delay that allows the exponent in SC to settle (C3) enters the chain at the 0 test. MQ1 is unnecessary for the 0 test in this case, but its inclusion merely

allows attainment of a significant result with certain unnormalized operands for which the result would otherwise vanish. For divide, there must be two right shifts, one to compensate for a possible overflow bit, the other because of the extra quotient bit generated to allow rounding (c above). FDT0B, the pulse in the divide sequence generated by the return from the divide subroutine, sets NRF2 and transfers the calculated exponent from FE back to SC. The final pulse in the sequence, FDT1, then triggers the required extra right shift of AR,MQ; the regular right shift occurs in normalize return. Divide enters at the beginning of the chain but is delayed slightly so that the total delay between FDT1 and NRT0 is ample to allow the preliminary shift-count to settle down. Even though there are two shifts, only MQ1 need be included in the zero test because MQ2 contains an extra quotient bit generated only for rounding.

In the normalizing loop, if AR9 and AR0 are the same and the fraction is not of magnitude 1/2 (the gating levels are generated respectively by the net in C2 and the decoding nets for AR, 6.2), NRT1 triggers NRT2. This pulse adds one to the negative exponent in SC thus decreasing its magnitude, and triggers an arithmetic left shift of AR,MQ. NRT2 also enters a delay whose output retriggers NRT2 if the number is not yet normalized. The loop continues until either AR9 and AR0 differ or the fraction has magnitude 1/2, at which time NRT2 goes on to NRT3. If SC is now positive, NRT3 sets the overflow flag. This does not necessarily mean that the normalizing loop counted from negative into positive exponents—the overflow or underflow may have occurred when the exponent was calculated before multiplication or division. Since the loop uses the negative representation of the exponent, it is already in proper form for insertion into the result if the result is also negative. However, if AR0 is 0, NRT3 complements SC.

At this point, the subroutine must determine whether any rounding action is required. The net that generates the round gate is in C3 and C4. The program specifies rounding by a 1 in IR6, but the gate is asserted only if a rounding action is necessary, i.e., if the MSB of the low-order part of the fraction is 1. The rounding sequence requires the repetition of the entire subroutine, so the assertion of NR ROUND first causes NRT3 to complement SC, making the exponent positive again for the initial right shift. Then NRT3.1 continues the chain to NRT5, which calls the subroutine that adds one to AR. The completion triggers NRT5A (lower right) which returns to the beginning of the subroutine (C2). Thus the entire routine is repeated in

case the rounding has overflowed (which can occur only if the fraction was all ones). NRT5A also sets NRF3 (D3) disabling the round net. Then when the chain again reaches NRT3.1, it automatically continues to NRT4, to which it goes on the first pass if no rounding is required. NRT4 transfers the exponent from SC to AR so the result is now complete and in proper floating-point form. This pulse also triggers NRT6 which returns to ET10 to store the result.

The rounding used here is in magnitude, i.e., away from 0—if the bottom part is $< 1/2$LSB of the top part then it is ignored, but if the bottom part is $\geq 1/2$LSB then the magnitude of the top part is increased by 1LSB. The reason for conditioning the round on a 1 in MQ1 is that the action both rounds and places the result in correct 2's complement form. The answer, if negative, is a 1's complement unless the low-order part is null and a 1's complement is one greater in magnitude than a 2's complement. Adding one because of a 1 in MQ1 increases the magnitude of a positive result when the lower order MSB is significant, but decreases the magnitude of a negative result when the low-order MSB is not significant. Of course, a 0 can be null in a negative number, but only if the entire low-order part is null, in which case the high-order part is already a 2's complement. The program should always round unless the low-order half is actually going to be used.

A further caution is necessary concerning division. The divide subroutine computes 29 quotient bits and leaves the remainder in MQ9-35. Following the two right shifts, MQ0 contains the true sign of the remainder, and MQ11-35 contain a truncated remainder—the two least significant remainder bits are lost in the shifts. If there was overflow in the division after normalization, MQ still contains two quotient bits, otherwise only one. In order to use the low-order part of the result, the program must either reconstruct the true remainder or save the extra quotient bits and append them at the left end of a lower order quotient calculated from the given remainder.

CHAPTER 7

MEMORY LOGIC

In a PDP-6 system, the core memories and fast memories are separate units connected to the arithmetic processor by a memory bus. A core memory may contain an 8K or 16K bank, and a fast flip-flop memory may replace the bottom 16 locations in any core bank. The internal operation of these memories, their control functions, their timing, and the way they respond to processor requests are described in a separate manual. This chapter describes only the hardware at the processor end of the memory bus: the logic elements that request access to memory, provide the necessary addresses, and control the transmission and receipt of data.

A time pulse in a main cycle or a subroutine may request access to memory by triggering appropriate operations in the memory control section of the processor. Memory control places a request signal on the bus and the processor must wait for a response from the memory addressed by the high-order address bits. Once the memory is free and available to the processor, the time required by the processor to transmit or receive data depends upon the type of memory. For a core memory, this time is usually much shorter than the memory cycle. The fast memory contains no buffer, so a read requires only slightly less time than a write. For reading from a core memory, the processor must wait until the data is available and the memory rewrites the word automatically; for writing the processor need wait only until the memory acknowledges the request, at which time the memory stores the data in its own buffer and continues with the clear and write cycle.

In addition to the standard logic for controlling access to memory, this chapter also describes the user mode registers. These govern the protection and relocation of areas in core and are included in the description of the memory address logic.

7.1 MEMORY ADDRESS LOGIC

The memory system appears to a processor as one homogeneous unit: the processor may address any one of 262,144 locations merely by providing an 18-bit address from the memory address register. The actual address put on the bus is the sum of C(MA) and C(RLR). The bus control

portion of each memory decodes four or five address bits (depending upon whether a given memory contains a 16K or 8K bank), and a given memory responds only to the address wired in. If the bottom of a core bank is replaced by a fast memory, an additional selection signal is required; this signal is generated by the processor.

a MA Register

Figure 7-1 shows the 18-bit memory address register. Transfers of addresses into MA, which must always be preceded by a clear, may be made from the memory buffer, the program counter, or the console address switches. For the transfer of short addresses of accumulators, index registers, and PI channels, individual pulses may directly set bit 30 and bits 32-35. The transfer gates that produce the pulses are included in the MA control circuits (b below). The MA flip-flops are connected in a carry configuration so that a pulse at the MA+1 input to MA35 adds 1 to the contents of the register. The carry chain is broken, however, between the fourth and fifth bits from the right. A carry out of MA32 cannot go into MA31 if the processor is fetching or storing a second accumulator; such actions occur for double-length shift operations and certain arithmetic instructions that use a double-length operand or produce a double-length result. When such an instruction addresses location 17 as an accumulator, the second AC is in location 0. In all other situations that increment MA, the second address is in the normal order.

To address a location within a single memory bank, address bits 21-35 are supplied over the memory bus with ones asserted negative. Bits 26-35 are supplied directly by MA, bits 21-25 from the sum with the relocation register RLR. Bus drivers for the direct bits are shown above MA; those for the relocated bits, RLA, are with RLR in Figure 7-5. The high-order relocated bits, which select a memory, must be supplied in both states at ground assertion. Figure 7-2 shows the required bus drivers and all control connections to the memory bus. Bits 21 and 35 are supplied over the bus in both forms; that is, they are supplied both as bits to select a memory bank and to select a location within a bank. A switch at each 16K memory allows the operator to select between the bits for the two uses. If bit 35 replaces bit 21, the memory locations are interleaved; i.e., all odd addresses are in one bank, all even in another. With an 8K bank, addresses must be interleaved because five bits are required to select the bank.

At the upper left in Figure 7-2 are the bus connections for the memory control section of the processor. At the right is a net that decodes MA bits 18-31 for all zeros, a condition which means that the address in MA is 17 or less. The decoder output generates the fast memory selection signals for the bus since a fast memory replaces the bottom 16 locations in core (fast memory addresses are not relocated). Additional fast memories require extra decoding. The signal that selects the fast memory is also conditioned by the 0 state of the readin mode flip-flop. This mode allows the operator to start a program in the area of core that is replaced by the fast memory and is ordinarily inaccessible to the program. The ungated decoder output is used by readin mode control to determine when the program leaves the readin area.

In addition to a flip-flop and three transfer gates, each MA module contains a logic net that compares the MA bit with the corresponding console address switch (these are shown below the input gates in Figure 7-1). The outputs of these nets are ANDed (Figure 7-2, left) to assert a level when the address in MA is identical to the address in the switches. This signal is used by the memory indicator logic (7.2).

b MA Control

Figure 7-3 shows the circuits that control the transfer of addresses into MA. Every transfer requires a pair of pulses: the first triggers the clear PA through the net in the upper left; the second triggers the transfer, usually through one of the PAs at the top of the figure. Two of the transfer pulses, those for MBRT and PC are applied to the gates shown with the register. Figure 7-3 shows no pulse for the transfer from the address switches because that signal is generated by the key logic and applied directly to the register gates. The other transfer PAs in Figure 7-3 set individual MA bits through the gates in the lower right. A UUO loads address 40 into MA by setting MA 30. The transfer of a PI channel address sets MA30 and loads the channel number into MA32-34, producing the address 40 + 2n. The other sets of gates load accumulator and index register addresses into MA32-35 from the appropriate IR bits.

The address transfers are as follows: A signal from the key logic clears MA prior to any transfer in from the address switches. At the beginning of every instruction cycle, IT0 clears MA and IT1, loads PC into it to retrieve the instruction, unless the processor is in a PI cycle, in

which case IT1 loads the PI channel address. AT0 clears MA in preparation for the address cycle. If the instruction specifies an index register, AT2 transfers the address in from IR14-17 and AT3 clears it again. If the address is indirect, AT5 transfers in MBRT and the cycle returns to AT0 after the new address is retrieved. When the address calculation is complete, the processor continues to the fetch cycle, in which FT1 transfers IR9-12 into MA to fetch an accumulator and FT1A subsequently clears if addressed by either half of the accumulator; FT3 transfers in MBRT and the clear follows at FT4A. Finally, FT5 again transfers MBRT to fetch C(E) and the effective address is left in MA as the cycle ends.

MA transfers in the execute cycle are required only for particular instructions. UUO, POPJ, and BLT all clear MA at ET1. For UUO, ET3 sets MA30 so that the UUO subroutine will subsequently deposit the instruction code in location 40. For the other two instructions, MBRT goes to MA at ET3. This special transfer is required for POPJ because this instruction jumps to the location specified by the contents of the top location in the pushdown list (transfers to PC must be made via MA). BLT loads a new address into MA in preparation for the subroutine which subsequently moves C(E) to a new location. Three instructions, PUSH, PUSHJ, and JRST, clear MA and load it late in the execute cycle (ET9, ET10). The first two make the transfer so that the following store cycle will deposit C(E) or PC in the top location in the pushdown list; the third uses it to display the current instruction location in case the JRST is a halt.

At the beginning of the store cycle, the appropriate address for the deposit of C(E) is already in MA. If storage of an accumulator is also required, ST3 clears MA and ST5 loads it from IR9-12.

In the upper right of Figure 7-3 is a net that increments the address contained in MA. The PA is triggered by a signal from the key logic for the operations examine next and deposit next, by UUO T1 to switch from location 40 to 41, and by ST6 which is generated by the store cycle only for the deposit of MQ in a second accumulator. The two gated pulse inputs allow counting by FT1A to fetch a second AC, and by IT1 if the indexing of the pointer has overflowed in a block IOT that is using the PI system. Note that this latter event occurs only when performing the second instruction in the pair for a PI channel and that the channel address is transferred into MA at the same time. Since the channel address always ends in 0, MA+1 merely sets MA35.

c User Mode Registers

Figures 7-4 and 7-5 show the two registers that provide protection and relocation for the user mode. The bits in both registers are numbered 18-25 to correspond to the most significant eight bits in a memory address. Information is transferred into the registers by a DATAO for the processor, but the clear and set inputs have the prefix EX because the gating by the CPA selection level is included with the executive mode logic. The DATAO loads the protection register PR from bits 0 to 7 of the I/O bus, the relocation register RLR from IOB18-25.

The executive routine determines the size and location of the block assigned to a given user program by loading PR and RLR. Since both registers correspond to the most significant eight address bits, the numbers they represent are actually multiples of octal 2000 (all numbers in this discussion are octal), but identical contents do not represent the same multiple; the two registers have entirely different functions. An address in MA is compared with C(PR) and it is legal only if MA is less than or equal to PR. Since the remainder of MA may have anything from all zeros to all ones, if PR is clear, MA may contain any address from 0 to 1777, i.e., any address in which MA18-25 is clear. A clear PR represents a block of 2000 words; setting PR25 represents a block of 4000 words. The number in RLR, however, defines the first address in the assigned block as a multiple of 2000 since RLR is added to MA18-25. An address in a user program is legal if $C(MA) \leq C(PR) \times 2000 + 1777$, and each address must be relocated to the actual address to which memory access is made is $C(MA) + C(RLR) \times 2000$.

Every time a user program reloads MA, bits 18-25 of the address are compared with PR by the chain of majority gates at the top of Figure 7-4. Each gate receives a 1 signal from an MA bit, a 0 from the corresponding PR bit and a carry in from the previous gate in the chain. The carry output of the gate is asserted whenever at least two of the inputs are asserted (a block diagram of the gate is in the lower left). The carry mentioned here is really a borrow—the comparison is a subtraction of MA from PR and the outputs are labeled to indicate that a comparison is ∿ OK when the input conditions are fulfilled. At the beginning of the chain (upper right), the borrow in is disabled and there is a borrow out only if the MA bit is 1 and the PR bit is 0. At all other stages, the output is asserted whenever MA is 1 and PR is 0 or if there is a borrow from the previous stage and either MA is 1 or PR is 0. No actual

difference is produced and the time required for the comparison is the time taken for level transitions across the chain each time MA changes state. A borrow out at the left end indicates that MA is larger than PR; and the two outputs of the last gate, when asserted at the same polarity, indicate whether or not MA contains a legal address. The nets in the lower right AND these two levels with ∿EX INH REL so that the signals sent to memory control (7.3) can be asserted only when the memory is being protected.

At the same time that MA is compared against PR, the contents of RLR are added to MA to relocate the address to the assigned block. Every block starts at a multiple of 2000 because the 8-bit RLR is added to MA18-25. The addition is carried out by a chain of dc adders shown above RLR in Figure 7-5. The circuit, shown in block diagram at the lower left, has three inputs: a bit of MA, the corresponding bit of RLR, and the carry from the next less significant adder. It has two sets of outputs, one a bit of the sum, the other a carry to the next more significant stage. The sum circuit uses exclusive OR nets to generate a 1 output whenever an odd number of the inputs are ones; the carry circuit uses a voltage division network to generate a 1 output whenever two or more of the inputs are ones. The RLR input is ORed with an enable level (the negation of the inhibit from executive mode control) so that if no relocation is desired, the sum output is equivalent to MA and there are no carries. This enable level is applied to pin H of all adders. The carry inputs to the first adder in the chain are disabled (upper right). Each time MA changes state, the sum of MA and RLR appears at the RLA outputs after a settling interval required for level transitions across the chain.

7.2 MEMORY DATA LOGIC

Transmission and receipt of data at the processor end of the memory bus are controlled by 36 pulse amplifiers, each with AND gates at both input and output (Figure 7-6). For a write cycle, the input AND gates are enabled by the 1 states of MB bits. To transmit data pulses over the bus, memory control applies a transfer pulse to all input gates and triggers those PAs that correspond to ones in MB. To receive information from a read, memory control generates a level that enables all output AND gates; pulses that arrive over the bus are then gated through to set the appropriate MB bits (6.1).

Figure 7-7 shows 36 MI flip-flops that drive a set of console indicators to display the contents of a memory location. The flip-flops are included on the arithmetic register modules, and the MB connections to the input gates are internal to the modules. Whenever the address for a memory access is the same as that in the console address switches, the read restart or write restart from the memory subroutine clears MI and loads MB into it. The write restart includes the condition of a write request for a nonexistent memory so that MI displays the word that would have been written. The indicators also display the contents of any location that the operator examines or deposits information into. The equal address condition suffices for examine and deposit, but an additional gate is necessary for examine next and deposit next, which use MC RST1 to trigger the transfer.

7.3 MEMORY CONTROL

Figures 7-8 and 7-9 show the logic that governs requests for memory access by the processor (flow chart, Figure 4-3). The processor may request three types of memory action: read, write, and read/write—the last being available only for operations in normal mode, which we shall consider first.

Pulses from the main cycles and a few subroutines trigger the memory subroutine for the appropriate type of memory access through the gates, at the top in Figure 7-8. The reason for the access is written along with the triggering pulse. Each gate generates an output that triggers the specific type of access required; however, any specific request also triggers a general request pulse (A5), and either type that requires a read clears MB (B5). For any instruction that requires fetch and pause, FT7 triggers the read/write request, but there are three inputs to the net that generate the restart for the subsequent store (B6). Indexing of a pointer in character operations or a block IOT is performed by a subroutine which skips the store cycle and returns directly to an earlier point in the main sequence. In these special cases, the restart is supplied by the subroutine; whereas, it is supplied by ST2 for all fetch-and-pause instructions that include the store cycle.

The request pulses supply the request levels to the bus by setting up the flip-flops in the upper left of Figure 7-9. A read request sets MC RD and clears MC WR, a write request sets MC WR and clears MC RD, and the read/write request sets both. A request for any type of cycle

clears MC STOP and sets MC RQ, although the general request pulse is not applied directly to the latter flip-flop; instead it enters the two delay lines in the lower right of the figure. If the system is in executive mode, the output of the right delay triggers the PA in C7 to set MC RQ; however, if the system is in user mode, a longer delay is allowed to carry out the necessary protection and relocation operations (7.1c). Then if the user program has supplied a valid address, the request flip-flop is set; but if the address is greater than the PR maximum address, the illegal address pulse (C6) causes the processor to skip the remainder of the current main sequence by jumping directly to ST7 (5.2e) and triggers a priority interrupt on the channel assigned to the processor (8.3).

The bus levels that request a cycle and specify read or write are derived through bus drivers from the outputs of the three request flip-flops. (Note that the cycle request is generated from MC RQ only if a read or write is also specified. This is done to prevent a possible malfunction in processor memory control from generating a memory request which would not result in a response from memory.) MC RD also supplies the bus enable level that gates data pulses from the bus into MB (7.2). If the operator has pressed the MEMORY STOP key or if the ADDRESS STOP switch is on and MA now contains the selected address, the main request pulse (delayed) sets the stop flip-flop (D4).

Completion of the memory subroutine is controlled by the circuits in the lower half of Figure 7-8. When the addressed memory becomes available to the processor, it returns a pulse (lower left) that acknowledges receipt of an address. This pulse triggers MC ADDR ACK, which clears MC RQ. For write access, the acknowledging pulse also sends a write restart back to memory, transfers data from MB onto the bus, and triggers the restart pulse MC RST1 from the memory subroutine to the waiting sequence (C6). For read or the read portion of read/write, the processor must wait for the read restart from memory to trigger the MC restart. In this case, the return from memory triggers a preliminary pulse so that the subroutine return is delayed. MC RS T1 clears MC RD and supplies the subroutine return. If MC STOP has been set during the subroutine, neither the read nor the write restart from memory can trigger the MC restart, and the processor stops even though RUN is not clear. The operator may then trigger the return to the waiting sequence by pressing the MEMORY CONTINUE key, which triggers MC RS T0 at KT1.

For a read/write cycle, the response by the memory in the first part is the same as for a read cycle: MC ADDR ACK clears MC RQ and the read restart generates the completion pulses, of which MC RS T1 clears MC RD. For the second part, the write request signal alone remains on the memory bus. The read/write restart pulse supplied by the processor duplicates the action of the address acknowledgement in a write cycle by triggering the write restart (Figure 7-8, lower left). Thus the processor, which already has access to the memory, both restarts the memory and provides its own subroutine completion pulses at the same time that it transfers data over the bus from MB.

The other circuits shown in the two memory control drawings handle special situations. In the lower left of Figure 7-9 is an integrating delay Type 4303 which is placed in the 1 state by every request pulse. The delay remains continuously in the 1 state as long as triggering pulses keep arriving at intervals shorter than the delay period. But should there be no request for 100 μsec, the delay returns to the 0 state, and if MC RQ is still set at this time indicating that the last request for access to memory has not been granted, the 4303 state change triggers the nonexistent-memory pulse (the pulse is inhibited if there is a memory stop). This pulse sets an error flip-flop in the processor I/O interface to trigger a priority interrupt on the assumption that if the addressed memory does not respond within 100 μsec, there is no memory with that address connected to the bus. If the console MEMORY DISABLE switch is off, the error pulse generates a restart that clears MC RQ by simulating the address acknowledgement from memory. For write, the simulated MC ADDR ACK also triggers the subroutine completion pulse in the normal manner; for any other cycle (MC RD(1)), an additional pulse is generated to supply the MC restart.

The remaining logic is associated with a pair of synchronizing flip-flops (Figure 7-9, right) that compensate for memory stops between the two parts of a read/write cycle. If the operator has pressed the MEMORY STOP key or the ADDRESS STOP switch is on (which would allow a stop on the equal address condition), AT4 sets the split/cycle synchronizing flip-flop. There is no need to set it earlier in the main sequence because all instruction and address memory requests are for read only: a fetch-and-pause request can be made only in the fetch cycle. The 1 state of the sync flip-flop prevents FT7 from triggering the read/write request pulse and prevents the read/write restart pulse from triggering the write restart for memory. Instead,

separate read and write requests are triggered by FT7 and the read/write restart (Figure 7-8, left center) to prevent the processor from memory in the event it stops. There is an extra switch input to the sync flip-flop set gate that allows the operator to override the split cycle for maintenance purposes. In case the processor might stop while hanging onto memory, the read/write request pulse sets another flip-flop, the stop sync. If the operator should restart the processor with some operation other than memory continue, KT1 triggers the write restart to memory so that it may finish its cycle, writing the same word back into the addressed location.

CHAPTER 8

INPUT/OUTPUT

The PDP-6 input-output system includes the peripheral equipment and three sections of the arithmetic processor. The processor elements are in-out transfer control including the I/O bus, the priority interrupt system, and an I/O interface for the processor that allows IOT instructions to control the processor itself as a device. In addition to the above processor equipment, this chapter describes the control units for four of the more common in-out devices: photoelectric paper tape reader, paper tape punch, Teletype keyboard-printer, and card reader. Other equipment may be added to the system merely by connecting the associated control units to the I/O bus.

8.1 IN-OUT TRANSFER CONTROL

Figure 8-1 shows the logic that decodes the IOT instructions and times the transfer of information into and out of the computer via the I/O bus. The 36 cable drivers for the bus data lines are shown in Figure 8-2. The I/O device selection lines come from bits 3-9 of the instruction register and the bus drivers for them are shown on the IR drawing, Figure 5-7. Every device control unit contains a diode net which receives input from 7 of the 14 IOS lines, one from each bit in IR3-9. The configuration of these input connections determines the selection code for a particular device, which responds to IOT commands only when the appropriate number appears in the device code portion of an instruction word.

The cable drivers for the I/O bus are transceivers, i.e., they handle signal transmission both into and out of the computer. IOT control governs output from AR onto the bus by supplying transfer and reset levels to the drivers; any signal placed on the external bus by an input device is automatically made available through the drivers to the AR input gating, with no transfer signal required other than command levels sent to the device, followed by the driver reset. As an example, consider the driver for bit 0 shown in the upper left of Figure 8-2. A 1 in AR0 is represented by a negative level at input E (the input resistors are mounted on the AR flip-flop modules). The transfer and reset levels from IOT control are connected, respectively, to L and M. In an output sequence, L goes negative for 2.5 μsec, transferring the

input at E to outputs J, F, and H: the assertion level of the first and third is negative; the second is ground. J merely drives an indicator on the console in-out panel. F drives the external bus and is connected to the capacitor-diode inputs of the control registers and output buffers in the various devices. F and H together provide bus signals at both assertion polarities for use in I/O operations within the computer—in the priority interrupt system and in the I/O control of the processor. At the same time that the transfer level returns to ground, the reset signal brings input M to ground, placing a hard negative on the bus (F) for 2 μsec. This action discharges the capacitor-diode gates connected to the bus to ensure that there will be no conflict with a subsequent I/O operation. For input, since the outputs of the device control/status registers and input buffers are connected (through diode gates with no dc load) to the same bus lines, an addressed device merely places ones at ground assertion on the bus. With the control signals at both L and M off, a ground at F produces a negative level at H, which is connected to the AR input gating. The gate for incoming status or data is applied to the device control for 2.5 μsec and is followed by the 2-μsec reset to the bus drivers.

The levels and pulses that control the bus and devices are generated by the logic shown in Figure 8-1 (flow chart, Figure 4-13). When IR0-3 contains the code for the IOT class, the net shown at the lower left decodes IR10-12 to determine the specific instruction. The four OR gates at the right develop levels that control functions common to more than one instruction. The IOT time chain always begins at ET4 (upper left). For a block instruction, the chain starts at IOT T0, which restarts the second part of a read/write cycle to write the indexed pointer word back into memory (7.3). The return from the memory subroutine generates IOT T0A which clears MA, sets IR12 to convert the block instruction into its associated data instruction, and triggers the fetch cycle for the data. It also carries out the required skip and overflow operations depending upon whether the block transfer is using the PI system and whether the block is complete.

In any nonblock IOT, including a data instruction within a block, ET4 sets IOT GO. The 0-to-1 transition of this flip-flop triggers the initial setup delay provided the system is not still within the reset period following a previous IOT. If two IOTs occur too close together, the second waits until the fall in the first reset level itself triggers the delay. The terminating pulse from the initial setup delay, IOT T2, triggers a pair of delays: a restart delay whose terminating pulse is IOT T3, and a final setup delay whose terminating pulse is IOT T3A.

The level outputs of the initial and final setup delays are ORed to produce a 2.5-μsec gate that places information on the bus (the pulse IOT T2 keeps the gate well grounded during the initial to final setup transition). During this period, an output instruction connects AR to the bus through the I/O cable drivers; an input instruction connects the data buffer or status register to the bus in the device control. For the two output instructions, IOT pulses T2 and T3 trigger the appropriate clear and set pulses for the device. The initial setup delay allows the device selection from IR to enable the capacitor-diode gates in the device enough ahead of the clear; the restart delay allows enough time between clear and set. For any input instruction AR is cleared prior to the IOT chain by ET0; the transfer in from the bus is accomplished by IOT T3. This pulse also causes the computer to reenter the execute cycle at ET5. When all transfers are complete, IOT T3A triggers the reset delay whose level output prevents the initiation of IOT operations for 2 μsec and resets the bus.

In addition to the timing logic, Figure 8-1 also shows a pair of general control signals (upper right). Whenever computer power is turned on, -15 volts is applied to pin C of I/O cable 3, turning on all peripheral equipment. The reset signal at pin B clears the device control units on the power clear and when the operator triggers the reset from the console (both conditions are included in MR START, see 5.1). The program can also generate this reset by means of a 1 in bit 19 of the same CONO that controls the flags in the processor I/O interface.

8.2 PRIORITY INTERRUPT

The priority interrupt system allows the program to be interrupted by a signal on any one of seven channels arranged in a priority chain. The priorities assigned to the I/O devices and to the processor are completely under program control. Moreover, the program may turn any channel on or off and may request a break on any channel. The processor checks for requests at the beginning of every instruction and address cycle, and honors immediately the highest priority request that has been made. Any requests made after the processor has completed the effective address calculation must wait until the end of the current instruction except in a block transfer. The processor honors a request by entering a PI cycle and executing the instruction in location 40 + 2n where n is the channel number. No further interruptions can occur while the computer is in a PI cycle. If the break instruction is a block IOT and the block is not complete, the processor leaves the PI cycle and dismisses the break automatically.

If the break instruction is not a block IOT or the processor goes on to the second instruction in the break pair, the PI cycle terminates at the end of the non-IOT instruction, which must be a JSR to a break routine. This routine may then be interrupted at any time by a higher priority request, and the program must dismiss the channel with a JRST at the end of the routine.

The processor I/O interface and all device control units each contain a 3-bit PI assignment register: the program assigns a priority to a device by loading a channel number into its assignment register (0 means no selection). The outputs of the register are applied to a gated binary-to-octal decoder whose outputs 1 to 7 are connected to the seven PI request lines. When a device requires service, it requests a break by gating on the decoder, grounding the line corresponding to the number in the register.

a Priority Chain

The 21 flip-flops in the priority chain are shown in Figure 8-3. Each channel is controlled by a column of three flip-flops, PIOi, PIRi, and PIHi, where i is the channel number. The setting of each flip-flop represents a stage in the interruption process for a given channel in the sequence: channel on, request made, break held. Between the rows of PIO and PIR flip-flops are the logic gates through which breaks are requested by a device or the program; between the PIR and PIH rows is the priority-determining chain of level gates which is activated by the input PI ACTIVE at the left. All other signals at the left control state changes in the flip-flops by rows (for the generation of these pulses, see b below). For program control over the system, the individual channels 1 to 7 are selected by ones in bits 29-35 (IOB29-35) of CONO.

Since the control sequence for all channels is identical, let us consider channel 4. The program selects this channel by a 1 in CONO bit 32 and turns the channel on or off by setting or clearing PIO4. To request a break on channel 4, a device must ground the PI4 line, pin V of IOC3. At specific times during its operations, the processor strobes the request lines by generating PIR STB. This signal sets PIR4 if the channel is on and a device has requested a break on it. The program, however, may bypass the PIO flip-flops by selecting a channel and setting the associated PIR flip-flop directly (PIR ← IOB (1)).

Several PIR flip-flops may be set simultaneously, but the PI system honors only the highest priority request (i.e., for the lowest numbered PIR flip-flop that is set), by asserting one of the PI REQ outputs shown at the top of the figure. The selection of a request output is made by the chain of level gates between the PIR and PIH flip-flops. The 1 state of PI ACTIVE enables the highest priority stage of the chain at the left. If a break is neither requested nor held on channel 1, the first stage enables the second. Similarly, if stage 2 is enabled and an interrupt is neither requested nor held on channel 2, stage 3 is enabled, and so on to each successive stage. For example, a request is honored on channel 4 in the following manner: If the request and hold flip-flops for the first three channels are all 0, stage 4 is enabled at 2N11R. If there is currently no break held on channel 4, PIH4 is 0, satisfying AND gate 2N11R,S and producing a ground at 2N12S and 2N13R. The 1 state of PIR4 satisfies the upper AND gate asserting the request output PI REQ4, but negates input S of the lower AND gate disabling the rest of the chain to the right. After the break has been started, PI control generates PIH ←PI CH RQ, which sets PIH4. The 1 state of this flip-flop negates input 2N11S, disabling both PI REQ4 and the remainder of the chain. Furthermore, it also holds PIR4 in the 0 state, preventing any further request on the same channel while the break is in progress. These actions, however, have no effect to the left, so a request on a higher priority channel can interrupt the current break.

To dismiss a break when returning to the interrupted program, PI control generates PIH(0) ← PI OK(1), which clears the PIH flip-flop for the channel on which a break is currently being held. It does this by clearing all PIH flip-flops from the left, up to and including the first one that is 1. The enabling levels for the clear gates in the stages to the left require that both the PIH and PIR flip-flops be 0, but this condition is bound to be satisfied because setting any higher priority PIR flip-flop would have caused an interruption of the break in progress. Thus, the system always dismisses the current break which is of highest priority and returns to a lower priority break, or if there is none, to the main program.

b PI Control

Figure 8-4 shows the circuits through which the program controls the priority interrupt system, and which governs the sequence of operations in the priority chain when a device requests a break. When an IOT with device code 004 appears in the program, the AND gate in B6

enables the selection level for the PI system. In CONI, the status level from IOT control gates PI ACTIVE and the PIO flip-flops onto the I/O bus (the bus gates are shown in Figure 8-5, upper right). CONO bit 23 gates the clear pulse to clear the entire system by generating PI RESET (B4). This pulse clears PI ACTIVE, clears the PIO flip-flops directly, and generates other pulses which clear the PIR and PIH flip-flops (A1, 3). The following set pulse in CONO triggers PI CONO SET, which performs the various operations specified by bits 24-28. Bit 28 turns on the PI system by setting PI ACTIVE (C7); bit 27 turns it off. The other three control bits perform operations on channels selected by ones in bits 29-35: bits 25 and 26 turn the selected channels on and off (A8); bit 24 requests a break on the selected channels (A3).

The remaining logic in Figure 8-4 controls operations associated with the requesting, execution, and dismissal of a break (most events are listed in the flow charts for the instruction and execute cycles, Figures 4-4 and 4-5). When any device grounds a request line, the inverted level from the PIR input gating lights an indicator located at 2L20 (right of the figure). The initial time pulses in both the instruction and address cycles check for an interrupt request by generating PI SYNC (B3), which triggers PIR STB (B5), provided the processor is not in a PI cycle. The PIR strobe sets the PIR flip-flop in any on channel over which a device has requested a break. As soon as one or more PIR flip-flops are set, the priority chain asserts one and only one PI REQ level. The nets in the lower left of Figure 8-4 generate a level PI RQ if any request is honored, and encode the channel number into binary. PI RQ allows a delayed PI SYNC (5.2a) to generate IATO (A5) which places the processor in a PI cycle by setting PI CYC (C6). The processor then returns to the instruction cycle, and using the binary-encoded channel number, retrieves from memory the instruction contained in location 40 + 2n (see 7.1b).

Since the most common interrupt for an I/O device is one that merely executes a single block IOT that takes the place of a whole subroutine, let us consider this type first. If the indexing of the pointer word overflows, IOT T0A sets PI OV when the processor switches from the block to the data instruction. If the block is not complete, both a hold and a restore level are generated (upper left). In the execute cycle, ET0 holds the break by setting PIH, only to have ET1 immediately dismiss it; then ET10 ends the PI cycle by clearing PI CYC. If the block is complete (PI OV(1)), the break is not held, and after completing the data instruction, the

processor performs the instruction in location 41 + 2n (which must be a JSR to the break routine). The sequence of events is now the same as it would have been had the first instruction not been an IOT. The break is held at ET0; ET10 dismisses the PI cycle and clears PI OV. The routine must then terminate with the appropriate restore instruction to dismiss the break (B2).

When the program performs a block transfer, PIR STB is generated at BLT T4 during the processing of every word in the block. If a request is discovered, there is no IATO; the processor instead terminates the incomplete block as though it were finished (6.6a), and then returns to the instruction cycle to finish the remainder as a new BLT with initial addresses one greater than those last used. However, since a request is waiting, the instruction cycle is interrupted and the processor does not restart the block until all requests have been honored.

8.3 PROCESSOR I/O CONTROL

The interface between the arithmetic processor and its own IOT control section is primarily a system of flags and enable flip-flops that allows the processor to check its own status and permits processor actions to request sequence breaks through the priority interrupt system. Certain data transfers may also be triggered through the interface. An IOT instruction selects the processor in the same manner as any other device: when the device code 000 appears in IR3-9, the diode net at the left in Figure 8-5 generates the level CPA which gates the various command signals from IOT control. For DATAI, CPA gates the contents of the console data switches into AR (this transfer does not use the I/O bus and the gate is shown with the key logic, Figure 5-1); a processor DATAO loads the memory protection and relocation registers from the bus (7.1c). The condition commands, gated by CPA, control and sense the flip-flops shown in Figure 8-5 plus the user mode flag (5.5) and the PC change and overflow flags in the AR logic (6.2e). CPA CONO SET regulates the flip-flops according to information on the bus; CPA STATUS gates the flip-flops onto the bus. Extra bus lines are used for conditions out so that the set pulse may set or clear individual bits; some flip-flops are set only by external conditions and CPA CONO SET can only clear them. A complete list of the bus lines and the CONO actions and status bits associated with them is included in the IOT flowchart (Figure 4-13). In addition to the enable flip-flops and flags, CPA includes a PI

assignment register PIA which is loaded from IOB33-35. Various flag conditions gate the PIA decoder to request an interrupt on the PI line addressed by the register.

The first two flip-flops at the left indicate errors: the left one is set when a user program attempts to use a protected area of memory; the right one, whenever the processor requests access to memory but specifies an address which fails to elicit a response within 100 μsec (7.3). Either error flip-flop triggers an interrupt when set. The next pair of flip-flops provide a means of synchronizing computer operations to real time. A filament transformer in the power control drives a pulse generator that sets the clock flag 60 times per second. If the enable flip-flop is on, the flag causes a break every time it is set. The other two enable flip-flops allow the PC change or overflow flag in AR control to request an interrupt. There is also an overflow flag for the pushdown list: if there is a carry out of AR0 in a pushdown or pullout instruction, ET10 sets PDL OV causing a break. Both types of instructions are represented by the level that gates a switch of MB and AR at ET10. AR CRY0(1) represents overflow for pushdown, underflow for pullout.

8.4 I/O INTERFACE LOGIC

Every peripheral device attached to PDP-6 must have a control unit that acts as an interface between the device and the IOT control section of the arithmetic processor. A control unit is connected to the processor via the in-out bus, but an automatic control unit for a high-speed device may also be connected to the memory bus for direct transfers of data to and from memory. The basic control unit includes a data buffer and a control section. The data buffer contains a number of bits appropriate to the device (up to a maximum of 36) and is connected to both the device and the I/O bus data lines. The control section handles receipt and transmission of control signals for both the processor and the device.

The data and control connections from a control unit to its device depend upon the organization of the device and are described with the appropriate equipment. All control units are connected to the processor via the in-out bus, whose configuration is shown in Figure 7-10, and which includes the following lines:

1 Power On Line - This line is at -15 volts whenever computer power is on. It is connected to the remote terminal of the local/remote switch in the power control for each device and normally turns on all peripheral equipment when the computer is turned on.

1 Reset Line - This line supplies negative pulses to clear the control registers and data buffers of all equipment attached to the bus. The line is pulsed when computer power goes on or when the operator presses the I/O reset key on the processor console; the program may also generate a reset.

36 Data Lines, IOB0-35 - These lines transfer all data and control information, with ones asserted at ground. For output, the lines connect AR to the device control registers and data buffers through capacitor-diode gates; for input, the status registers and data buffers are connected to AR through Type 4657 Diode-Gate Bus Drivers (connection must be made by a 4657 or equivalent circuit that places no dc load on the bus).

7 PI Request Lines, PI1-7 - A device requests an interrupt on one of seven priority channels by placing a ground on the appropriate line.

14 In-Out Selection Lines, IOS3-9 - These seven pairs of lines are derived from the outputs of IR3-9, which contains the device code in an IOT instruction. Assertion is at ground for both ones and zeros. A device control is connected to only seven of the lines, one from each pair. The configuration of these connections determines the selection code for the device, which then responds to only those IOT commands that are accompanied by the appropriate code.

6 Command Lines - These lines provide two pairs of negative pulses for output, two negative levels for input. Both pulse pairs include a clear and a set: one pair loads conditions from the bus into the control register, the other loads the data buffer. The two input levels gate status and data onto the bus.

Every control unit includes control and status registers, although these registers usually overlap and may even be identical. That is, all control flip-flops can usually be sensed by the program as status, and all status flip-flops within the control unit can usually be governed by CONO. Some devices have additional status signals which can be sensed by the status checking instructions but cannot, in general, be affected by the program directly.

The control/status registers for all devices include the same basic elements. As an example, consider the control/status register for the paper tape reader, Figure 8-6. The basic elements are a 3-bit priority interrupt assignment register PIA and two control flip-flops, BUSY and FLAG. These elements have the same names in most control units but are distinguished by an appropriate equipment prefix. The tape reader has an additional control flip-flop that governs the data mode, i.e., whether data is to be read from the tape in binary or alphanumeric. The punch register contains the same bits; whereas the Teletype has two busy flip-flops and two flags because it is actually two devices—one input, the other output. Even in large scale in-out systems such as magnetic tape, the control connections to the processor are the same as in the tape reader, and the control and status registers differ only in complexity, not in kind. Every device has a PIA register, but it may have a command register associated with the busy flip-flop, several flags, and a number of status bits for error and other conditions. CONO can provide a maximum of 18 control bits; control information on a larger scale such as full addresses and word counts must be supplied by a DATAO. A maximum of 36 bits can be checked by a single status instruction; if more are necessary, additional device codes must be used.

Selection of a device and the timing of all transfers of data, initial conditions, and status between the processor and the peripheral control units are described in detail in 8.1 (which the reader is strongly advised to study before investigating any of the peripheral equipment). At the left in Figure 8-6 are the connections to the processor for PI requests, IOT commands, and device selection. Every device is connected to the six command lines that govern the transfer of conditions out, data in and out, and status in. The pulses provide output clear and transfer functions while IOT control gates AR onto the bus; the levels gate input onto the bus while IOT control pulses the information into AR at the processor. All devices require the condition and status commands; however, only a bidirectional device uses both types of data commands.

Although all devices are connected to the six command lines, to have any effect on a control unit, the commands must be gated in by a selection level that is generated by the net in the lower left of the figure. Every control contains such a diode net, wired to the seven IOS lines that define the device selection code. The control unit then responds to IOT commands only when the appropriate code appears in the instruction. Note that the reset bypasses the selection gate in triggering IC CLR and clears the control registers for all devices.

In the normal sequence of operation for any device, the program first provides the initial conditions by means of a CONO, whose pair of command pulses generates the IC pulses within the control unit. The first pulse usually clears all the control and status flip-flops; the second pulse loads the assigned channel number into PIA and sets up the other control bits as necessary. The control information is supplied over the in-out bus by the effective address calculated for the CONO: bits 33-35 always provide the priority assignment; the configuration for other bits depends upon the device and is listed in the flowchart of the input-output operations (Figure 4-14).

The 1 states of the status bits are separately ANDed with the status level, which is generated by any of the three conditions-in instructions. The AND gate outputs are applied, in general, to the same bus lines over which the equivalent initial conditions are sent to the control register. The outputs of individual bits in the right of the figure provide control levels for the various operations within the control unit. The outputs of the three PIA bits, however, are applied to a gated binary-to-octal decoder. Outputs 1 to 7 of this decoder are connected to the request lines of the priority interrupt system. A group level at pin P enables the decoder, placing a ground on the output corresponding to the number contained in PIA, i.e., requesting a break on the channel assigned to the device. The 0 decoder output is not connected, so it corresponds to no PI assignment.

In addition to governing information transfers, the data commands also perform certain control functions so that an entire block of data can be processed after giving the initial conditions only once. CONO must assign a PI number and provide the necessary data mode information, but it need not necessarily set BUSY. A common procedure for output is that each DATAO not only loads the buffer but also sets BUSY causing the device to operate. When a control unit has completed the transfer of data from buffer to device, its completion

signal clears BUSY and sets FLAG. The latter action enables the PIA decoder requesting an interrupt. The processor then responds with another DATAO which provides new data, clears FLAG, and again sets BUSY. Following the last word in the block, the program must provide an extra CONO to clear FLAG. For input, the initial conditions must include setting BUSY to make the control unit retrieve information from the device; the completion of the transfer from device to buffer then clears BUSY and sets FLAG for an interrupt. Then the DATAI that gates the buffer onto the bus also clears FLAG and sets BUSY causing the control unit to retrieve more data. The turnoff of the DATAI level clears the buffer in preparation for the next word. To end a block, the program must follow the last DATAI with an immediate CONO to clear BUSY, preventing the retrieval of an extra word.

8.5 STANDARD IN-OUT EQUIPMENT

Small scale in-out devices most commonly used with PDP-6 are paper tape reader and punch, Teletype keyboard-printer, and card reader. This section describes the control units for these four devices; control units for other peripheral equipment are described in separate publications. The remaining Chapter 8 logic drawings are in four groups of two or three each for the standard devices. In each group, the first drawing shows the control/status register and control connections to the processor via the I/O bus as described in 8.4. The other drawing or drawings in each group show the control connections to the device, and the data buffer with its connections to both the device and the I/O bus.

a Paper Tape Reader

In addition to PIA, BUSY, and FLAG, the control register for the tape reader (Figure 8-6) contains a data mode flip-flop B. There is also an extra status bit which indicates that the reader motor is on. Since the reader is an input device, for initial conditions CONO must provide a PI assignment, clear FLAG, set BUSY to cause the device to retrieve the first data from tape, and set up B according to the desired mode. Alphanumeric mode (B=0) reads a single 8-bit character from tape and makes it available over bits 28-35 of the bus with hole 1 in IOB35. Binary mode (B=1) reads holes 1-6 of only those characters in which hole 8 is punched and assembles six such characters into a 36-bit word. The first character encountered is made available over bits 0-5 of the bus with hole 1 on IOB5.

Figures 8-7 and 8-8 show the 36-bit buffer, which is made up of six Type 4221 6-Bit Shift Registers. A single shift register module contains one buffer bit from 0 to 5 and every sixth bit after it; output from a single hole position is always read into a single module. For example, hole 1 is associated with that part of the buffer containing bits 5, 11, 17, 23, 29, and 35 (upper right, Figure 8-7). The strobe generated by the feed hole reads hole 1 into bit 35 (the presence of a hole is indicated by a negative level), hole 2 into bit 34, etc. In alphanumeric mode, B(0) causes the strobe to load holes 8 and 7 directly into bits 28 and 29 at the same time that the other six holes are loaded into the least significant bits of the 4221 Registers. Thus in reading a single 8-hole character, the 4221 Registers are not used as shift registers at all. But in binary mode, the first strobe loads the six data holes into the least significant bits of the register, and each subsequent strobe shifts the previously read character one place to the left in parallel at the same time that it reads in a new one. The characters are counted in an extra 4221 Shift Register in the upper left of Figure 8-7. Every strobe sets the right bit SR6 and also shifts left those ones that have already been strobed in. The sixth strobe sets SR36 indicating that the entire 36-bit word has been assembled.

Retrieval of information from tape is controlled by the logic shown in the lower left of Figure 8-8. When the operator turns on the reader from the console, the level MOTOR ON generates a pulse that clears BUSY to ensure that the reader does not go into operation inadvertently. This start clear pulse also sets FLAG to cause an interrupt when the motor goes on, and the same action is also produced through a pulse generator in Figure 8-6D6 when the motor goes off. When a CONO for the reader appears in the program, the first command pulse clears the control register. The second command pulse must then set BUSY, whose 0-to-1 transition generates PTR CLR (Figure 8-8C4) to clear the buffer and the character counter SR. BUSY(1) also engages the clutch and releases the brake (lower left), placing the tape in motion. (The operator may duplicate this action by means of the console tape feed switch but no reading occurs.) The feed hole signal is connected to two pulse generators—one through an inverter—so that a leading edge pulse is generated on the transition-to-ground when the feed hole is encountered and the trailing edge generates a second pulse on the negative return. If BUSY is 1, the leading pulse triggers a 400-μsec delay whose termination generates the strobe that loads the buffer. This is timed to occur

when the holes are centered with respect to the photocells. In alphanumeric mode, every feed hole strobes all eight data holes into buffer bits 28-35. The trailing edge pulse then sets FLAG and clears BUSY: the former enables the PIA decoder, the latter releases the clutch and engages the brake, stopping the tape. (The trailing edge pulse is gated by BUSY(1) to prevent the tape feed from setting FLAG.) When the processor responds, DATAI gates the buffer onto the bus and clears FLAG. The termination of DATAI sets BUSY to restart the reader and the flip-flop transition again clears the buffer and SR by generating PTR CLR.

In binary mode, every feed hole triggers the delay but the terminating pulse generates a strobe only if hole 8 is punched. The strobe loads holes 1-6 into the least significant bits of the shift registers and sets SR6. Six strobes fill the buffer and shift a 1 into SR36, which allows the trailing edge pulse to set FLAG and clear BUSY. Retrieval of the word by the processor and restart of the reader are the same as for alphanumeric mode.

b Paper Tape Punch

The punch has the same control bits as the reader: a 3-bit PIA register, BUSY, FLAG, and a data mode flip-flop B (Figure 8-9). B determines whether punching is in alphanumeric or binary mode; unlike the reader, however, either mode punches just one 8-bit character per DATAO and the only difference between modes is in the character format. Alphanumeric mode (B=0) punches the character supplied by bits 28-35 of the bus, with IOB35 controlling hole 1. Binary mode (B=1) always punches hole 8, skips hole 7, and punches holes 6-1 according to the characters supplied by IOB30-35. Also unlike the reader, the program rather than the operator turns on the punch motor and there is a wait of 1 second after turnon to allow the motor to reach speed. To reduce wear, the motor goes off whenever the program does not call the punch for 5 seconds. Because the program must turn on the motor, the initial conditions may vary. However, in normal circumstances CONO supplies a PI assignment, sets up B according to the desired mode, and may set FLAG if the programmer wishes to handle an entire block including the first character through the PI system. Then the clear for every DATAO clears FLAG and sets BUSY to trigger the punch cycle for the character supplied by the instruction; when the cycle is complete, a done pulse clears BUSY and sets FLAG to request a priority interrupt by gating on the PIA decoder.

Figure 8-10 shows the data buffer with associated gates, punch solenoid drivers, and timing circuits that govern the motor and punching cycle. Whenever the program sets BUSY or the operator feeds tape from the console (either of which generates GO), the 4303 Integrating Delay in D6 is set and remains set as long as it is triggered at intervals shorter than the delay period. The 1 state of this delay turns on the punch motor by enabling the Type 823 SCR Driver (A1) and holds the motor on for 5 seconds after the final GO. The gate to the driver is inhibited, however, by the power clear enable from the key logic so that the punch cannot go on when computer power is turned on. The delay 1 state is also ANDed with the off level from a second delay to generate SPEED, which is in turn ANDed with BUSY(1) to generate READY. The second delay is triggered by the turnon of the first, so the assertion of the two signals by GO is delayed 1 second if the motor is off. Both signals indicate that the motor is up to speed and punching may proceed, but the busy condition in the ready signal prevents the tape feed from setting the flag and causing an interrupt.

The circuits in the upper left synchronize punching to the period of the motor and drive mechanism. A reluctance pickup provides a sync mark through a pulse generator whose output, gated by READY, triggers a 5-msec delay. The delay level output, ANDed with BUSY(1), gates the data buffer 1 outputs to enable the drivers for the punch solenoids. At the end of the punch interval, the solenoids release and the delay terminating pulse, DONE, clears BUSY and sets FLAG. Note that the input from the tape feed key operates the feed hole solenoid driver directly, provided the motor is up to speed. This allows the operator to generate tape leader: the tape feed key enables the speed level after the appropriate delay but does not generate the ready level, so it punches feed holes without punching the contents of the buffer.

The data mode flip-flop controls the character format through the gates in the lower left. In binary, B(1) provides a hard ground through the 4113 Diode Gates to hold PTP8 set and PTP7 clear regardless of the action of the DATAO clear and set pulses on the remainder of the buffer. In alphanumeric, the 4113 outputs float, and punching of holes 7 and 8 is determined solely by the contents of the corresponding buffer bits.

c Keyboard-Printer

The Teletype is actually two separate and distinct devices with a common PI channel and device code. Signal names in the logic drawings use three prefixes: TTY for elements common to both devices, TTI and TTO for signals unique to the keyboard and printer, respectively. In addition to PIA, the control register (Figure 8-11) includes two pairs of control flip-flops, TTO BUSY, TTO FLAG, TTI BUSY, and TTI FLAG. In order to allow the program to control the devices separately, the first pulse in the CONO pair clears only PIA; both clearing and setting of the other control bits are handled directly by CONO bits (of course the IOB reset does clear the control bits in addition to PIA). In regular operation, the control bits are handled by the data instructions and signals derived from flag state changes in the transmitter and receiver. For output, the DATAO clear sets TTO BUSY and clears TTO FLAG; the subsequent setting of TTO DONE in the transmitter clears TTO BUSY and sets TTO FLAG. For input, TTI BUSY is set by the 0-to-1 transition of TTI ACTIVE when the operator strikes a key. When the entire character has been received, TTI DONE is set, clearing TTI BUSY and setting TTI FLAG. The response by the program with a DATAI then clears TTI FLAG. Setting either flag gates on the PIA decoder to request an interrupt. During a prolonged idle interval, PIA should be left clear to prevent interrupts due to inadvertent keyboard manipulation. When setting up PIA following an idle period, the CONO should also clear TTI FLAG to prevent readin of a character that may have been typed accidently.

Data transmission between the processor and the Teletype control unit is in 8-bit characters over bus lines 28-35. IOB35 corresponds to the first character bit and the eighth bit (IOB28) is always 1. Between the control unit and the keyboard-printer, data transmission is in the form of 11-unit characters which are presented serially at 110 bits per second, so one complete character requires exactly 100 msec. Character transmission always begins with a start impulse (space), followed by the eight data bits in order—with ones represented by marks—and transmission is terminated by a stop impulse (2 marks). An idle line marks continuously.

At the right in Figure 8-12 are a crystal oscillator and a countdown chain that generate a pair of clocks to regulate the transmission and receipt of data. The higher speed clock for input provides greater time resolution: this allows sampling the keyboard data near the center of each bit to prevent ambiguity during level changes. Almost all of the remaining logic in the figure is included in two modules, a 4706 Teletype Receiver and a 4707 Teletype Transmitter. Each includes an 8-bit data buffer with associated control circuits and flags. Both buffers are shift registers (the output buffer is actually included within a 10-bit shift register).

TTO (Figure 8-12, upper half)

The transmitter module receives no initial reset, so the program or the operator should always send a rubout character after computer power turnon. After every transmission, the data buffer is left clear in readiness for the next character. When a DATAO triggers a transmission cycle by setting BUSY, it also clears DONE and then loads the character into the buffer and sets ENABLE. The stop timer is a 4-counter that holds at 4 until enabled by a 1 count applied to a terminal separate from the clock input. The 4 count ANDed with ENABLE(1) conditions the set gate for ACTIVE so the next clock sets it. ACTIVE(1) enables ÷2, and the transition clears OUT LINE, placing a start impulse (0) on the printer line. Clocks 2 and 3 then set and clear ÷2, whose clear transition generates the first shift pulse. Shift 1 clears ENABLE, shifts the 1 that was in ENABLE to TTO8, and moves the entire character left one unit, placing the first character bit in OUT LINE and out to the printer. The seven succeeding shifts, at clocks 5, 7, ..., 15, 17, move the rest of the character out to the printer one bit at a time; they also fill the register with a 1 followed by zeros immediately after the eighth data bit. When TTO1-8 contain 10 000 000, the AND gate in A6 is satisfied, enabling the ACTIVE clear gate. The shift at clock 19 then clears ACTIVE and shifts left again, clearing the register and setting OUT LINE to feed a stop impulse to the printer. ACTIVE(0) disables ÷2 to inhibit further shifts, enables the stop timer by loading in a 1 count, and sets DONE, which clears BUSY and sets FLAG. It also holds OUT LINE on to keep the line marking.

The next character cycle cannot begin until the ACTIVE set gate is enabled again. Clock 22 asserts the 4 count: if the new DATAO has already set ENABLE or does so before the next clock, ACTIVE is set at clock 23 and printing continues at maximum rate; if not, ACTIVE(1) does not occur until the clock following DATA SET. Thus the stop is held on the line for four clock periods or two character units (minimum) as required by the printer; and for maximum printing rate, DATAO should appear within this interval. The left and right sections of the output timing chart show, respectively, the flip-flop conditions for relatively late and normal DATAO arrival within the maximum rate interval. Levels shown at the left also apply to first-character or slow-speed printing except that the 4 count would be up already.

TTI (Figure 8-12, lower half)

At the beginning of an input cycle, the data buffer contains the previous character; other conditions are as shown at the left end of the input timing chart. The negative start impulse from the keyboard distributor asserts SPACE, which enables the ACTIVE set input through the AND gate in D8. ACTIVE(1), produced by clock 1, enables ÷ 8; the transition clears ÷ 8, clears LAST UNIT, sets BUSY, and generates SET, which sets all flip-flops in the buffer. LAST UNIT(0) enables the shift one-shot, so each 4 count asserted by ÷8 generates a shift pulse. The first shift, at clock 5, clears DONE and loads the start impulse into TTI8 as a 0; successive shifts read in the character bits and move the 0 left. When TTI1 contains the 0, the last shift fills the buffer and sets DONE and LAST UNIT. The DONE transition sets FLAG and clears BUSY; when the program responds, the DATAI gates the buffer onto the bus and clears both FLAG and DONE. LAST UNIT(1) inhibits the one-shot but enables the lower ACTIVE clear gate; no more shifts are generated within the cycle but the next 4 count clears ACTIVE, inhibiting ÷8 and enabling the upper input to the AND gate for the ACTIVE set input. The next character cycle begins when a start impulse satisfies the lower input to this gate.

The AND gate in D7 connected to the upper ACTIVE clear input prevents activation of the unit by momentary noise pulses at the data input. The start impulse must prevail through clock 5; otherwise ~TTI SPACE satisfies the gate (the buffer was set at clock 1) so the first shift clears ACTIVE to reset the device.

d Card Reader

In addition to PIA and FLAG, the control register for the card reader (Figure 8-13) contains START instead of the usual BUSY, a data mode flip-flop B, an error flip flop DATA MISSED, and an auxiliary flag, FLAG ALL. All of these but START are available for status checking, and the reader provides four additional status signals. Since START is cleared as soon as the reader starts a card cycle, the busy status is the OR of START(1) and CARD CYCLE, which is derived from the CCL signal from the reader. The negation of this signal, CARD DONE, is also available separately as a status bit (27) as are the other four reader status signals, FEED CHECK, CRL, CREL, and EOF (bits 23-26). CRL indicates that the card reader is not ready for operation, CREL indicates a validity check or read check error in the reader (for a full explanation of reader conditions, refer to card reader operation, 3.2d). The EOF signal indicates that the card hopper is empty and its assertion triggers the pulse generator in B8 to set FLAG. The PG is also triggered whenever a feed check error occurs.

Both modes determined by B assemble standard 36-bit computer words from six 6-bit characters, but the type of information and the number of columns varies. Binary mode (B=1) reads all 12 bits in a single column and assembles three columns into one word. Although 12 bits are taken from the column, it is read as two 6-bit characters. The data lines are designated CRXL where X has the values 1, 2, 4, 8, A, B, and CR1L is the least significant bit. Each column read requires two strobes: the first reads the lower half (holes 4-9) with hole 9 corresponding to CR1L; the second reads the upper half (holes 12, 11, 0, 1, 2, 3) with hole 3 corresponding to CR1L. In alphanumeric (Hollerith) mode, the reader converts the 12 bits in a column to a 6-bit code which is then read by a single strobe. This mode therefore assembles six columns into a single word. Either mode, however, requires six strobes for six characters.

Since the reader is an input device, for initial conditions CONO must provide a PI assignment, set START, and set up B according to the desired mode. Usually the program begins a card in binary because the first column contains information about the card format. Setting START causes the reader to read an entire card even though the program can retrieve no more than six columns at a time. To retrieve a single column at a time, CONO sets FLAG ALL, allowing every column read to set FLAG, which otherwise is set for every six

characters. FLAG(1) requests a priority interrupt in the usual manner, and the DATAI response by the program clears FLAG but does not affect START, which is cleared when the card cycle begins. If the program does not respond quickly enough after FLAG is set, the buffer is cleared and DATA MISSED is set, informing the program that there are still six characters in search of a reader. Completion of the entire card sets FLAG even though the buffer may not contain a complete word (a card has 80 columns). The program must provide a new CONO to read the next card.

Figures 8-14 and 8-15 show the 36-bit buffer, which is made up of six Type 4221 6-Bit Shift Registers. Each shift register module contains a buffer bit from 0 to 5 and every sixth bit after it; output from a single data line is always read into a single module. For example, CR1L is associated with that part of the buffer containing bits 5, 11, 17, 23, 29, and 35 (Figure 8-14, upper right). The strobe generated for a character reads CR1L into bit 35 (a 1 is asserted negative), CR2L into bit 34, etc. The first strobe loads the six data bits into the least significant bits of the register and each subsequent strobe shifts the previously read character one place to the left in parallel at the same time that it reads in a new one. The characters are counted in an extra 4221 Shift Register in the upper left of Figure 8-14. Every strobe sets the right bit SR6 and also shifts left those ones that have already been strobed in. The sixth strobe sets SR36 indicating that the entire 36-bit word has been assembled.

Retrieval of information from a card is controlled by the logic shown in the lower left of Figure 8-15. The first CONO pulse clears the entire buffer and character counter (C4) at the same time that it clears the control register. When START is set, SCCL is asserted (D2) causing the reader to process an entire card. When the reader starts the card, CCL asserts CARD CYCLE which clears START but continues to assert the busy status. For alphanumeric mode, CBIL is negated causing the reader to translate each column from the 12-bit Hollerith code to a 6-bit character; if B is 1, however, signals from the hole positions are applied directly to the data lines. The reader indicates that data is ready by sending a column strobe pulse to the control unit, which responds by asserting a level that causes a validity check in alphanumeric mode only. If the column does not contain a valid Hollerith character, the data lines are cleared so all zeros are read; and if the operator has pressed the validity check button on the reader, it stops and must be reset manually. The CSP from the

reader also triggers the PA in D1 to generate a strobe that loads the character into the buffer and counts it in SR. The PA output also triggers a delay which asserts CBHL for 20 μsec. This signal has no effect in alphanumeric, but in binary it causes the reader to put the upper half of the column on the data lines in place of the lower half, and the terminating pulse from the delay triggers a second strobe. There are thus six strobes for six characters whether the unit reads six columns or only three. The delay terminating pulse also indicates the presence of a column in the buffer and sets FLAG if FLAG ALL is 1. If full words are being assembled, the fifth strobe sets SR30 which then allows the sixth strobe to set FLAG. When the program responds, DATAI gates the buffer onto the bus, and the turnoff of the gating level triggers the clear to prepare the buffer for the next character. Each pulse from the 20-μsec delay also triggers a 2.2-msec delay. If this terminates while FLAG is still 1, indicating that the data has not been retrieved, DATA GONE clears the buffer and sets DATA MISSED. Finally when the card is finished, CCL from the reader is negated, generating CARD DONE, which sets FLAG and again triggers the DATA GONE delay.

The logic also includes a gate that allows the program to read any column twice. If the program gives a CONO during the card cycle, the set pulse for the initial conditions, which may change the mode, also triggers a short delay whose termination triggers the same PA that is usually triggered by the column strobe. The CONO thus generates a strobe or strobe pair that rereads the column in either the same or the opposite mode. This feature is usually used in conjunction with FLAG ALL to determine what mode to read in a card, based on information contained in the first column. The program usually starts the card in binary to inspect the first column. It may then clear FLAG ALL and reread the first column as part of a full word either in binary or in alphanumeric. The program may also use this feature to reread the third column of a binary card so that the final 78 columns are read in groups of three. If a CONO given during a card cycle sets START, the next card will be read as soon as the present one is finished.

CHAPTER 9

MAINTENANCE

This chapter discusses preventive and corrective maintenance for the arithmetic processor, tape reader, punch, keyboard-printer and card reader. Since neither the processor nor its memories can operate in isolation, the information herein must be used in conjunction with maintenance information in the memory manual. Except for the exhaustive preventive maintenance procedures for the basic in-out equipment, the maintenance is discussed at the system logic level. Circuit troubleshooting and repair are described in PDP-6 Circuits; specific lubrication and adjustment procedures as well as corrective maintenance for the in-out devices are included in the following manufacturer's manuals:

Digitronics Perforated Tape Reader Model 3500

Teletype Bulletin 215B: High Speed Tape Punch Set (BRPE)

Teletype Bulletin 281B: Model 35 Send-Receive Teletypewriter Set (KSR), Vols 1 and 2

Teletype Bulletin 1187D: Parts, Model 35 Send-Receive Teletypewriter Set (KSR)

Burroughs B122 Card Reader Technical Manual (B122.51)

The first section of the chapter discusses operation of the equipment for maintenance purposes and describes those controls and indicators not used in the course of normal system operation. The second section discusses maintenance programs and includes a list of those for the equipment described in this manual, plus the fast and core memories. The third section includes preventive maintenance schedules and a description of marginal check procedures. Finally, corrective maintenance includes troubleshooting procedures and a description of the construction of a diagnostic and exercise program loops. Special tools and test equipment required for the performance of the various procedures are listed below; except for DEC equipment, suggested commercial brands are given for purposes of specification only, and do not constitute exclusive endorsement.

Multimeter	Triplett Model 630-NA; Simpson Model 260
Dual-channel oscilloscope	Tektronix 580 series, preferably with delayed sweep trigger facilities
System module extender	DEC Type 1954
System module puller	DEC Type 1960
Paper tape gauge	DEC Type 18467
1-inch width gauge	DEC Type 0001
Tape reader cleaning kit	Digitronics MS-133
Feeler gauges	Any quality set, 1-25 mils
0-10 pound spring scale	Chatillon 719-10
0-20 pound spring scale	Chatillon 719-20
Tape punch maintenance kit	Teletype Bulletin 1124B lists all special punch tools; order with discretion
Punch lubricants	Teletype KS7470 oil; Mobilgrease #2
KSR-33 lubricants	Teletype KS7471 grease (installations without punch add KS7470 oil)
Card reader lubricants	Burroughs S15821-26 and S64960-2 oils; S15821-32 and -39 greases
Lint-free cloths	Cheese cloth or equivalent
Cotton swabs	Q-tips or equivalent
Cleaning fluids	Dupont Freon TF; denatured alcohol
Test cables and probes	Low-capacity probes for the oscilloscope; alligator clips; etc.
Super Filter Kote (aerosol)	Research Products Corp., Madison, Wisconsin

Also have standard hand tools including Phillips screwdrivers and a complete set of Allen wrenches. The card reader requires a drift punch, a brass drift rod, a plastic-headed hammer, and a pair of 18-inch water-pump pliers. Standard cleansers for the bay exteriors, etc. should also be available. At installations that include magnetic tape equipment, consult the tape transport manual for additional housekeeping necessities.

9.1 OPERATION FOR MAINTENANCE

Chapter 3 discusses normal operation of the processor and basic in-out equipment, and explains the use of the controls and indicators that are regularly available to the operator. There are many additional controls, primarily for maintenance. They include controls for ac line and for marginal checking, and maintenance switches for the logic.

a Power Controls

Most power controls and all power supplies are mounted on the plenum doors at the rear of the bays; all their switches and indicators face outward, directly behind the exterior double doors. The main power control, usually a Type 835, is at the bottom of bay 4 (the left bay of the console viewed from the rear). It has a red light that is on whenever the external ac line is plugged in, a ganged pair of ac circuit breakers, and a LOCAL/OFF/REMOTE toggle switch. Switching to LOCAL turns on power; when the switch is in REMOTE, power is controlled from the console. The total time that power has been on is registered by an elapsed time indicator at the lower left on the in-out panel (Figure 3-3). In some processors the main control is a Type 829 which has an additional switch for a delayed output that is not used. This type of control also supplies a power clear enable directly to the power clear clock; so in a system using the 829 the integrating delay shown in the lower right of Figure 5-1 is not present. Turning on the main power control supplies ac to the fans and the power supplies.

Located in the upper part of the bay 3 plenum door is a secondary power control that supplies ac for the motors in the reader, punch, and Teletype. This control is usually a Type 834, otherwise Type 811; both have controls and indicators identical to those on the 835. When this secondary control is in REMOTE, it goes on whenever the power supplies are turned on by the main control. Also at the top of bay 3 is a panel containing six ac convenience outlets.

As explained in 3.2, power to the reader and Teletype motors is controlled by the operator. The ac line to the punch is controlled by a Type 823 (mounted directly on the punch motor) through which the logic turns the punch on and off. For maintenance purposes the logic may be bypassed by means of an ON/OFF toggle switch located behind the right end of the chad box. The card reader can go on only if the operator has turned on its main power switch located on the left side of the stacker. Then it turns on with system power in the same way as the 834: a −15 vdc turnon signal is supplied through the I/O bus. If system power is off, the reader may be turned on and off independently by the POWER ON and POWER OFF buttons on the reader console.

b Marginal Check Controls

At the top of the bay 4 plenum door is a Type 734 Power Supply whose floating output may be varied from 0 to 20 volts. Its controls are on the front of the console at the bottom of the in-out panel (Figure 3-3). The 3-position switch controls the polarity of the supply output; the large knob controls the output voltage amplitude, and the dc voltmeter indicates the magnitude of the output (a similar meter is mounted directly on the supply).

In every logic mounting panel (except the lower panel of a pair containing double-height modules) pins A to D are connected to the power and ground lines. To allow submodular marginal checking, there are two independent +10 volt power lines, +10A and +10B; one is bussed to all A pins in a panel, the other to all B pins. All D pins are grounded. The C pin on every module receives −15 volts, but all the C pins are bussed together only if the panel contains no pulse amplifiers. The C pins of the pulse amplifiers in a mounting panel are separately bussed to permit independent application of negative marginal check voltage to them (in the processor the pulse amplifiers are module types 1607, 1609, 4606, 6603, 6609); the remaining modules in the panel always receive the fixed −15 volts.

At the left end of each logic panel are three toggle switches. When all three switches are down, all the modules in the panel receive the normal fixed voltages. Turning the polarity switch to +10 MC and pushing up the top or middle toggle switch applies the output of the variable power supply to the +10A or +10B bus respectively on the panel. Turning the polarity switch to −15 MC and pushing up the bottom toggle switch applies the marginal check

voltage to the C pins of the pulse amplifiers in the panel. Turning the polarity switch OFF applies normal voltage to all power lines regardless of the setting of the toggle switches. While marginal checking a panel, the toggle switches need not be turned off when switching from one polarity to the other because the polarity switch applies correct fixed voltages to all unselected lines (i.e., those of opposite polarity) even if the individual panel switches are left on. However, this interlock and the center-off polarity switch position are provided only as a convenience and should not be used as a substitute for turning off the toggle switches. At the completion of marginal check procedures the technician should turn off all three marginal check toggle switches on every mounting panel.

Marginal check voltages may also be applied to the photo amplifiers in the tape reader. Two switches, one for the eight code holes, the other for the feed hole, are located below the console shelf in the upper right corner of mounting panel 4K. When both switches are in the +10V position (left), all amplifiers receive the normal fixed +10 volts. Turning the polarity switch to +10 MC and pushing the FEED HOLE switch to the right (also labeled +10 MC) applies the output of the variable supply to the amplifier for the feed hole photodiode output. Similarly the CODE HOLE switch allows application of marginal voltage to the amplifiers for the information holes.

c Maintenance Switches

Mounted on brackets between the mounting panels on the wiring side are five toggle switches that allow the technician to alter the normal operation of the processor as an aid in troubleshooting the logic. All of these switches are off when down. One switch was mentioned in Chapter 3 because it must be used when depositing a readin loader in the bottom of core. The RIM MAINT switch is mounted just above 2M1,2, and it is shown in the lower left of Figure 5-2. Turning the switch on grounds the 0 output of the RIM SBR flip-flop so that it cannot be cleared no matter what location is used for instruction retrieval. The processor thus stays in the readin mode indefinitely, and a memory checkerboard may be run that includes the readin area. Also shown on Figure 5-2 is the REPEAT BYPASS switch which is mounted above 1M22, 23. While this switch is on, KT0A triggers the repeat delay, so the key cycle can be repeated even when the chain does not include KT4.

Two of the switches provide interruptions in the normal flow of events in the arithmetic and shift-count subroutines. Shown in the upper right of Figure 6-9 and mounted above 1H18, 19 is ART3 MAINT, which prevents the carry completion in an AR subroutine from generating the return pulse ART3; the return is instead simulated by KT2. A technician may single step through a shift-count by using SCT MAINT, which is mounted above 2C7,8 and shown at the left in Figure 6-16. With this switch on, neither SCT0 nor SCT1 can trigger the next step in the subroutine; instead each step is triggered by KT2.

The last switch may be used for troubleshooting the memory from the processor. This is SPLIT CYCLE OVERRIDE, shown at the right in Figure 7-9 and mounted above 1N15, 16. Ordinarily if the MEMORY STOP key or the ADDRESS STOP switch is on, AT4 sets the split cycle sync flip-flop. Then if the fetch cycle should generate a fetch-and-pause request, the memory subroutine makes only a read request to memory; so a processor that is undergoing trouble-shooting procedures does not hold the memory during a stop. But in troubleshooting the memory it may be desirable to stop it between the read and write parts of a cycle. The override switch accomplishes this objective by disabling the set gate to the flip-flop.

Other than the power and marginal check controls mentioned in a and b above, there are no special maintenance switches for the reader, punch, or Teletype. The card reader, however, has switches inside the hopper and stacker that generate NOT READY when the former is empty or the latter full, and these may be operated manually for maintenance. Under the reader cover is a local/remote toggle switch, LOCAL RUN. Raising this switch (local) causes the reader to feed cards continuously until the hopper is empty, and then stop with FEED CHECK on.

Besides the switches there are many maintenance indicators. This category includes at the top of bay 1 most of the indicators which display the states of all SBRs and many control flip-flops and levels. The PI REQUEST lights on the console indicate those channels on which requests have been synchronized, but the signals arriving at the logic over the I/O bus are displayed by a set of lights mounted in front of module connector 2L20. The top light is for the PI1 line.

d Single Step Operation

By using the STOP and CONTINUE keys for instructions, the operator may single step a program from one instruction to the next. Similarly, for maintenance the technician may single step from one memory call to the next by using the STOP and CONTINUE keys for memory. In either case the STOP key should be latched on, and each restart may be done manually by pressing the CONTINUE key; or the single stepping can be performed automatically by latching on the CONTINUE key, turning on the REPEAT switch, and setting the SPEED controls to the desired time interval. Each time the processor stops following a memory call, the MEM STOP light is on, and at the top of bay 1 a light must be on for the SBR that is awaiting the subroutine return. The meaning of the information displayed by the indicators on the console and the bay indicator panels can be determined by following the flow charts as the processor proceeds. Using REPEAT, the function associated with any initiating key can be repeated; all require that KT4 set the repeat delay, and the required switch settings are discussed at the end of Chapter 5.

By using the maintenance switches discussed in c above, the processor can be operated in even finer steps. Turning on ART3 MAINT inhibits the return pulse from the AR subroutines; so the processor stops every time an addition, subtraction, index, or negation is performed. The immediate results of the subroutine can then be seen by observing the lights for the arithmetic registers on bay 2, and the point at which the stop occurred is indicated by one of the SBR lights on bay 1. To observe the operation of the shift counter and the effect of shift pulses applied to AR and MQ, turn on SCT MAINT. The processor then stops prior to the shift-count subroutine and also following every step in it. After either an ART3 or an SCT stop, the processor should be restarted by pressing MEMORY CONTINUE. The attendant key cycle supplies the necessary pulse for simulating the subroutine pulse, but triggers no other operations because there is no memory stop.

To produce an automatic restart after every stop, turn on REPEAT BYPASS and latch on MEMORY CONTINUE; this causes every KT0A to retrigger the repeat delay, and the level from the key causes the delay termination to retrigger the key cycle. The ART3 and SCT switches can be left on together, so the processor stops after every AR subroutine and every step in a shift-count; if MEMORY STOP is left on too, the processor will also stop after every memory subroutine. MEMORY CONTINUE supplies the restart for all three types of stop.

While the processor is single stepping, the technician should follow the operations in the flow charts and compare the information given by them with the state of the processor as displayed in the lights. The point at which the processor has stopped is indicated by the lights on bay 1. The indicators display a great deal more information about the workings of the processor at stops within an instruction than they can provide when the processor is stopped only at the end of an instruction. For example, the indirect light I is always off at the completion of an instruction, but when single stepping through an address cycle with AR and memory stops, the light may sometimes be on. For particulars always refer to the flow charts.

9.2 MAINTENANCE PROGRAMS

MAINDEC (maintenance DEC) programs permit self-testing of the PDP-6 for check-out, preventive maintenance, or diagnosing equipment malfunctions. Each MAINDEC package consists of program tapes and a reference manual. All manuals have the same format: an abstract; a section containing operator information; another suggesting applications of the programs; and a third describing the programs.

The first section is all that the operator need use when running a MAINDEC. It lists the required tapes, specifies the usage of the console switches, and gives detailed instructions for loading and starting the programs. The final part specifies how errors are indicated to the operator (e.g., programmed halts or typeouts), provides information as to the cause of the error, and tells the operator how to repeat or restart the program. The second section suggests various applications including procedures for using the MAINDEC with marginal checking. It specifies the panels and margin levels that apply to each program. The third section contains detailed program descriptions including octal and symbolic listings; flow charts are included where appropriate. Each description explains what the program is testing and in what manner, and furnishes information to aid in updating or modifying the program.

The following MAINDECs are applicable to the basic hardware (consult the DEC program library for the current list of all MAINDECs for PDP-6).

Test	MAINDEC
Instruction Test	
Part 1	601-1
Part 2	601-2
Part 3	601-3
Part 4	601-4
Part 5	601-5
Micro Checkerboard	602
Memory Address Test	603
Clock Test	604
Memory Speed Test	605
Memory Retention after Power Failure	606
Memory Overlap	607
Power Failure Test	608
Reader Binary Test	610
Reader Alpha Test	611
Punch Test	612
Memory Data Test	613
Teleprinter Test	614
Memory Checkerboard	
Low 64 x 4K	662-1
Hi 4 x 4K	622-2
16 x 16K	622-3
16 x 16K Interleave	622-4
Protect and Relocate Test	623
Card Reader Test	641
Fast Memory Test	662

9.3 PREVENTIVE MAINTENANCE

This section discusses preventive maintenance schedules and use of marginal check during PM procedures, and lists recommended procedures for the arithmetic processor and the basic in-out equipment. Preventive maintenance consists of tasks performed prior to initial operation of the system and periodically during its operating life to ensure that it is in satisfactory operating

condition. Faithful performance of these tasks forestalls possible future failure by discovering progressive deterioration and correcting minor damage at an early stage. The tasks consists of mechanical checks, including cleaning and inspection; marginal checks, which aggravate borderline circuit conditions or intermittent failures to make them easy to detect; and checks of specific elements such as power supplies, in-out interface circuits, drivers, and sense elements.

a Schedules and Margins

Preventive maintenance procedures should be performed according to strict schedules. Recommended intervals for PM checks are based on elapsed operating time as well as calendar schedules. For convenience in scheduling normal operations, and to minimize malfunctions, large-scale PM procedures that occur at long intervals should be staggered. Each user should set up a schedule for his entire installation that distributes the total PM task evenly over the longest interval recommended. In every operating shift, a specific period should be assigned to performance of the scheduled PM program for that shift. Without an appropriately designed program of this type the PM function would necessitate system unavailability for two or three days every month.

Marginal checking utilizes the MAINDEC diagnostic programs to test the functional capabilities of the system with module operating voltages biased within specified margins above or below nominal levels. Failures brought about by over- or under-biasing are detected by the program, which provides a printout or other visual indication helpful in locating the source of the malfunction. Marginal components can then be replaced during scheduled preventive maintenance. After such preventive replacement, or when no marginal components exist, operating voltages are then biased beyond the specified margins, and the margins at which circuits fail are recorded in the maintenance log. By plotting bias voltages obtained during each scheduled PM, progressive deterioration is easily observable and expected failure dates are predictable. Accurate information acquired over a long period provides a basis for planning future preventive replacement. These plots are also useful in locating marginal or intermittent components such as deteriorating transistors.

Raising the operating voltage above +10 volts increases the transistor cutoff bias that must be overcome by the previous driving transistor; therefore low-gain transistors fail. Lowering the voltage below +10 volts reduces transistor cutoff bias and noise rejection, and thus provides a test for high-leakage transistors and simulates high temperature conditions (to check for thermal runaway). Raising or lowering the −15 volt supply (this is done only to pulse amplifiers) increases or decreases the output pulse amplitude. Each panel has a separate negative bus to pulse amplifiers; the line to pin C for logic level modules comes directly from the power supply, whereas the line to pulse amplifiers comes through the marginal check switch. The +10 margins are normally +5 and +15 volts; the −15 margins are −7 and −8 volts. The −15 volt supply must never be made more negative than −18 volts or damage to the logic can result.

For every system in the PDP-6 the user should keep preventive maintenance voltage charts on which are plotted the failure margins for each major section of logic while running the appropriate MAINDEC. DEC supplies a standard form for such plots; each page has three pairs of charts for +10A, +10B, and −15 marginal checks. The ordinate is the margin level as read from the meter on the supply; the abscissa is the time (nine spaces are provided below each chart for the date). At each check-out, two points are plotted: the voltage level which is too high for normal operation, and that which is too low. Thus a chart shows the change in time of the voltage region over which a particular section of the logic operates properly. When equipment malfunction is induced by abnormal margin levels (e.g., the region being too narrow), troubleshooting procedures (9.4) should be performed to diagnose the abnormality. At the top of each chart, spaces are provided for entering the page number in the maintenance log where such dysfunction is explained.

b Arithmetic Processor PM

The procedures in this section apply only to the arithmetic processor; procedures for the in-out equipment are given in c below.

Daily Operator Maintenance

1. Run all five parts of the Instruction Test (MAINDEC 601) without margins. Log all error halts, noting the cause if known.

2. Check that all cooling fans of the system are running and that cooling air flows freely through the filters.

3. Replace any noncritical components such as indicators, fuses, etc; note any replacement in the log.

The remaining procedures are to be performed by trained personnel only.

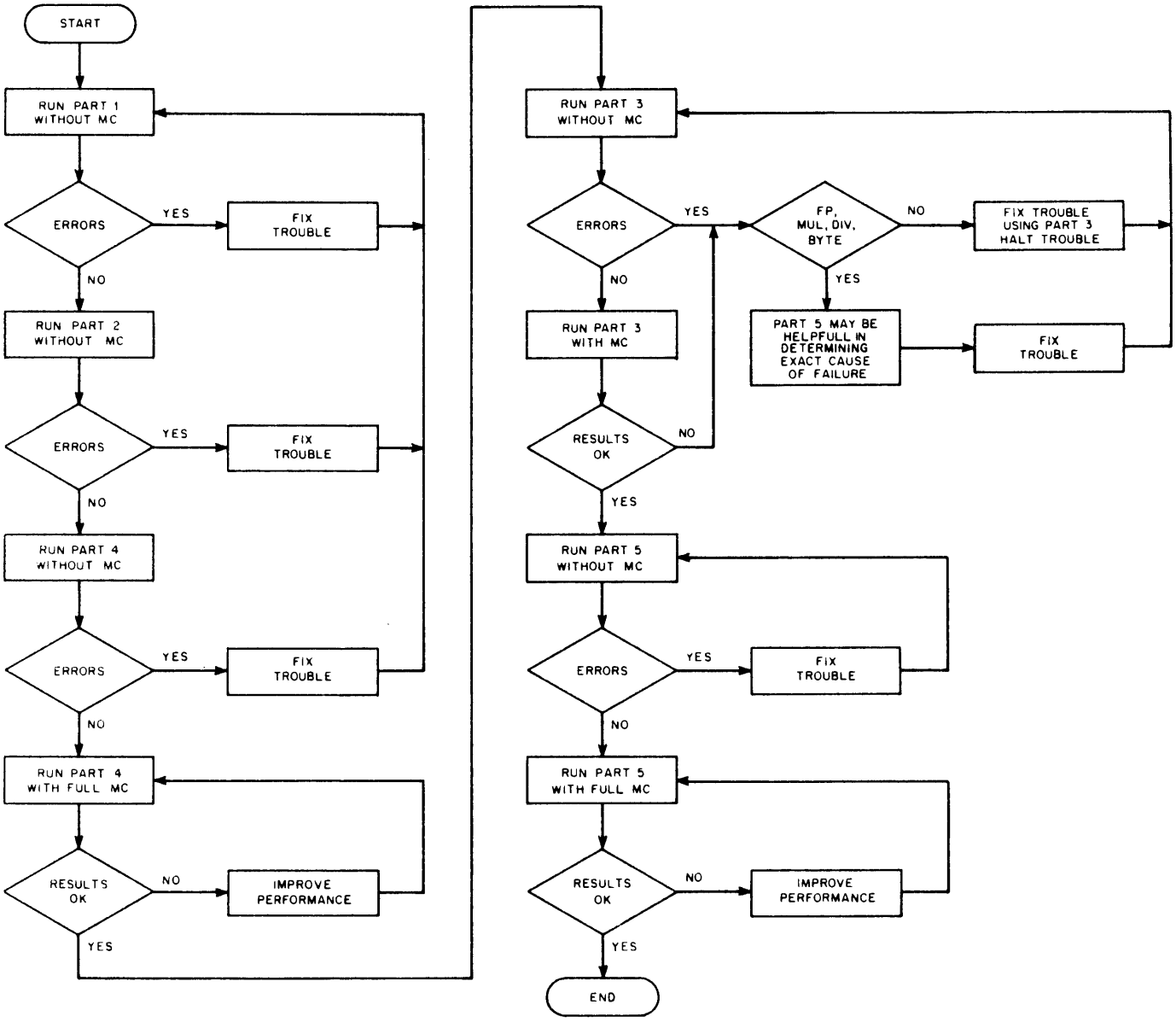


Figure 9-1 Processor Marginal Check Flow

Weekly

1. Check the operator log; note any malfunctions which have occurred in the machine, and take corrective measures if necessary.

2. If trouble noted in the log cannot be reproduced by normal trouble-shooting methods, contact the operator who made the note; using his program, try to reproduce the trouble. If the system can be kept operational by use of moderate margin voltages on certain small sections of logic, do so. Do not fail to make an appropriate entry in the log describing any provisional measures adopted, and notify the scheduling authority that diagnostic down time will be required to rectify the trouble.

Every 1000 Hours

1. Run all five parts of the Instruction Test (MAINDEC 601) with margins, using the flow diagram (Figure 9-1) and plotting the results on the preventive maintenance voltage charts. In case of failure at abnormal margin levels, note all circumstances in the maintenance log and cross reference the log to the chart by page number.

2. Change and clean the air filters at the bottom of every bay in the entire system using the following procedure:

   a. Loosen the two thumb screws holding the fan and filter housing to the floor of the bay.

   b. Remove the housing. Take the filter out of the housing and install a clean one.

   c. Replace the housing containing the clean filter and tighten the two thumb screws.

d. Clean the dirty filter by flushing it thoroughly with hot tap water in a direction opposite to that of air flow. When all dust and lint is removed shake out excess moisture.

e. Stand the filter on one end for 10 to 15 minutes to allow remaining moisture to evaporate. If the flush water is sufficiently hot, the filter should dry completely in about 15 minutes.

f. Spray the filter with aerosol Super Filter Kote or an equivalent product. The spray serves both as a dirt capturing medium and as a detergent which helps wash out trapped dust and lint during the next reverse flushing.

c In-Out Equipment PM

This section includes preventive maintenance procedures for the tape reader, punch, keyboard-printer, card reader, and DEC logic associated with them. Procedures for all four devices are grouped together under the recommended PM intervals.

The paper tape equipment requires accurately punched tape. A DEC standard paper tape gauge is supplied with the system; use it for all paper tape measurements. A paper tape with the correct specifications is shown on the next page.

The most important dimension is the lateral distance from feedhole to inner edge: it must be .392 ± .002 inch. When examining the tape on the gauge with the naked eye, this dimension must be exact—the smallest visible error will exceed the tolerance. The reader has a wide tolerance to variations in longitudinal hole spacing, but mechanical readers in general do not. To ensure that tape punched by the processor will be readable on mechanical readers such as the Teletype Model 35 or the Flexowriter, longitudinal hole spacing must be within ±5 mils accumulated error in 6 inches. The 5 mil tolerance is about one eighth the diameter of the feedhole. The output of every punch in the system should be checked once every day to ensure that the base dimensions are within tolerance. It is much easier to punch the tape accurately than to try to jockey the reader adjustments to read a wide tolerance of tape dimensions.

The punch can be adjusted to produce accurate lateral spacing on any given tape whose width is within 7 mils of the nominal 1 inch. Once it is so adjusted, any tape of 1 ± .007 inches will punch properly. If possible, make adjustments while punching the narrowest tape (.993 inch).

The following procedures are recommended for the in-out equipment.

Daily Operator Maintenance

Tape Reader –

1. Using Digitronics cleaning kit MS-133, clean the read head, tape guides, roller bearings, capstan, and brake.

2. Run the reader tests (MAINDECs 610, 611) without margins. During the tests, check that the tape is laterally positioned to ride along the inner edge of the tape guide; the photoelectric read head is designed to accept tape in this position. Log all error halts, noting the cause if known.

3. Remove the exciter lamp cover and check that the lens is clear and the photodiodes clean; replace the bulb if there is any sign of yellowing.

4. Check that the lamp cover springs are adjusted so as to just touch the top surface of the tape.

Tape Punch –

1. Run the punch test (MAINDEC 612) without margins. Log all error halts, noting the cause if known.

2. Using the DEC standard tape gauge, check that the punch output conforms to the following standards (see Figure 9-2).

   a. Feed hole to inside edge: .392 ± .002 inch.

b. ±5 mils accumulated longitudinal error in 6 inches.

c. Width: 1 ± .007 inch.

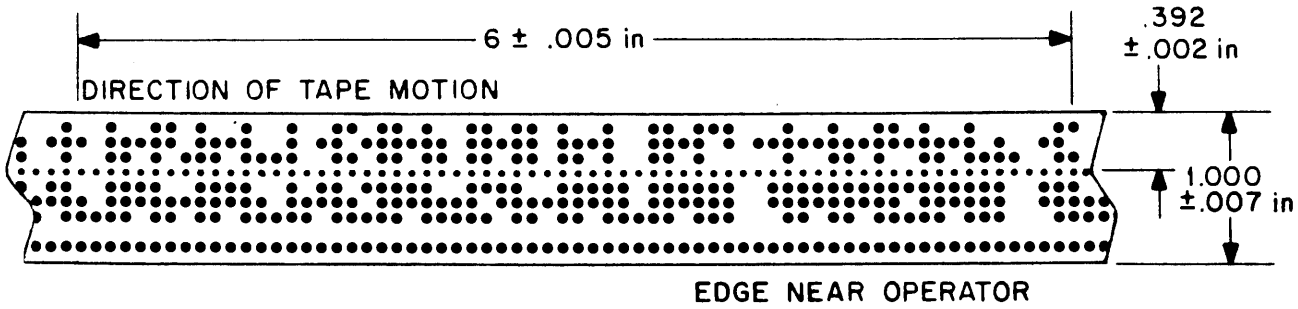


Figure 9-2 Paper Tape Dimensions

3. Check for "fuzzy" holes. This indicates a misadjusted or worn out die block.

4. Empty the chad box (the chad box should also be emptied every time a new box of tape is installed).

5. Clean the punch by blowing dust off the die block. Do not use alcohol or other solvents near the feed pawl or die block since such solvents remove the light lubricating film.

Teleprinter –

1. Inspect and clean the platen and paper guides as necessary (in general, the platen needs cleaning only if typing has run off the page or the printer has been run without paper).

2. Remove lint and other fouling material from the ribbon guides and replace the ribbon if necessary.

3. Run the Teleprinter Test (MAINDEC 614) without margins. Log all error halts, noting the cause if known.

Card Reader –

1. Remove the exciter lamp assembly. Clean the solar cells, the feed head, and the hopper area with a soft bristle brush.

CAUTION

Use only Freon as a cleaning agent for the solar cells. Other cleaning fluids may damage the potting in the assembly.

2. Run the Card Reader Test (MAINDEC 641) without margins. Watch for unusual wear on the test deck. Log all malfunctions with probable cause if known.

Every 160 Hours (Monthly)

Tape Punch –

1. Remove the punch from the console and perform the complete lubrication procedure outlined in 5 of Teletype Bulletin 215B. Be sure that no oil or grease accumulates between the armatures and magnet pole faces or between contact points.

2. Inspect the general condition of all moving parts and tightness of wiring connections. Make sure nuts and screws that lock the adjustments are tight, but do not apply sufficient torque to disturb the adjustment. See that all contact points meet squarely. Rotate the main shaft slowly in the normal direction (clockwise as viewed from the front) and activate all movable elements. Check for freedom of movement.

3. Run the Punch Test (MAINDEC 612) with margins. Plot the failure points on the PM voltage chart; log abnormal failure margins and cross reference the log to the chart by page number.

Every 330 Hours

Tape Reader –

1. Carefully perform the daily PM paying particular attention to the position of the tape as it proceeds through the reader, and making sure that it travels against the inner edge of the tape guides. Use an accurately punched paper tape, i.e., one which is well within the dimensional tolerances in Figure 9-2.

2. Carefully check the alignment of read head photodiodes with the holes in the accurate tape (with lamp cover still removed). Turn on the reader and thread the tape through the guides but not through the clutch or brake. Using a line with all holes punched, center the feedhole directly over the feedhole diode. The diodes for holes 1 and 8 must lie directly under the centers of those holes. If this alignment is incorrect or if skew is apparent, correct it by shimming either the read head or the tape guides.

3. Check the exciter bulb voltage: 8.5 ± .1 volts.

4. Check the pinch roller to capstan gap clearance with clutch de-energized. The proper clearance is 10 to 12 mils; if the clearance is insufficient, dirt may be limiting the full travel of the clutch solenoid.

5. With no tape in the reader, turn it on and lift the READER FEED key to engage the clutch as if reading tape. The correct clutch setting is that which allows the pinch roller just barely to engage the highest point on the rotating capstan, causing a slight jittering sensation in the finger when held against the roller. If the roller does not contact the capstan, or runs continuously with it, adjust the clearance according to Section 4-13 of the Digitronics manual. Severe jitter indicates excessive capstan runout, which should be tested as follows:

a. Turn the reader off, pull it from the console on its extension slides, and remove the drive belt from both capstan and motor pulley.

b. Turn on the reader and lift READER FEED to engage the clutch. Using 1- and 2-mil feeler gauges check the clearance between capstan and pinch roller for various angular positions of the capstan, turning it by hand. If runout is excessive, the capstan must be replaced.

c. Turn the reader off and replace the belt.

6. Using a spring tension scale, check that the force necessary to unseat the activated pinch roller is from 3 to 4 pounds. If adjustment is necessary use the procedure in Section 4-15 step 2 of the Digitronics manual.

7. Check for ridges on the capstan around the area where the edges of the tape ride. Such ridges can cause tape skew; they may be removed by applying a fine file to the capstan while the reader is running. Do not allow rubber dust to collect on the clutch block.

8. Check brake tension using a short piece of paper tape with a loop in one end. Insert the free end of the tape between the brake solenoid and its armature; use a spring scale hooked into the loop to measure the tension required to move the tape with the brake engaged. The force must be from 3-1/2 to 7 pounds.

Teleprinter –

1. Lift the cover and check everywhere for effects of vibration: loose nuts, screws, retaining clips, etc.

2. Check the selector magnet coils for signs of overheating. Make sure there is no lubricant or other fouling material under the armature. If necessary,

insert a piece of bond paper between the pole and armature to soak up any lubricant. Make sure no lint is left.

Card Reader – Sections and figures referenced below are in Burroughs Card Reader Technical Manual B112.51.

1. Check exciter lamp brilliancy according to 3.16; adjust if necessary. The procedure requires a deck of cards punched with a repetitive pattern of all ones in twelve columns followed by all zeros in the next five. This step and step 2 must be performed whenever photo amplifiers are changed in the reader.

2. Using the same test deck, check the leading edge timing of the strobe pulse with respect to the read pulse for all twelve rows (Section 3.17).

3. Check operation of the strobe-8 counter according to Section 3.18 (performance can be checked only by adjusting the counter).

4. Check response of card-detect solar cell CD1 according to Section 3.19 (the procedure requires a deck of cards having a 4 punch in column 80).

5. Check performance of the translator for valid Hollerith codes (Section 2.3), using a deck prepared by punching each valid code in alternate columns of a card.

6. Test performance of the validity check circuits (Section 2.4), using a 50-card deck in which column 40 of each card contains one of the invalid punches (all invalid combinations are listed at the bottom of Table A4-3).

7. Using Burroughs S64960-2 oil, lubricate the feed-slide/gib bearing surfaces (Figure 3.4-1, right), the feed-knife pivot (Figure 3.4-1, C), and the feed-slide fork/actuator-arm pivot post bearing surfaces (Figure 4.5-1, top right).

8. Run the Card Reader Test (MAINDEC 641) with margins. Plot the failure points on the PM voltage chart; log abnormal margins and cross reference the log to the chart by page number.

Every 1000 Hours

Tape Reader –

1. Perform the daily and 330-hour PM schedules. Tape must travel along the inner edge of the tape guides.

2. Check all power supply voltages including the exciter lamp voltage.

3. Using the same length of accurately punched test tape with alternate lines of ones and zeros, run the Reader Tests (MAINDECs 610, 611) with margins applied to the read amplifiers and reader control logic. Run the reader at top speed (400 characters per second). Make no adjustments unless the tape being read has lateral dimensions well within the tolerances in Figure 9-2, and travels against the inner edge of the tape guides.

4. Observe the feedhole signal at pin 18M in the panel containing the reader logic (beneath the processor console) with one vertical input of a dual-channel oscilloscope. Adjust the feedhole output for a width of 1 msec.

5. Apply a +9 volt margin to the code hole amplifiers, and adjust the read amplifier cards while observing both the code hole and feedhole signals with the scope. The trailing edge of the feedhole should lead the trailing edge of the code holes by 300 to 500 μsec; code hole width should be approximately 1.8 msec when it satisfies this trailing edge timing requirement. A read amplifier that exhibits extreme changes in gain with moderate rotation of the adjustment potentiometer should be replaced with a less sensitive card.

6. Using MAINDEC 610, run the test tape at all speeds up to 400 characters per second with code hole margins from 3 to 17 volts and feedhole margins

from 5 to 15 volts. Amplifiers should be touched up as necessary to obtain these margins. Error halts occurring with a splice on the test tape between clutch and brake can usually be disregarded since these are generally caused by poorly made splices. Plot the margin levels on the PM voltage chart; if adjustments are required, log the prior and post adjustment margin levels and cross reference the log to the chart by page number.

Tape Punch –

1. Check the Teletype Maintenance Kit for parts which may require periodic replacement.

2. Run the Punch Test (MAINDEC 612) with margins. Plot the failure points on the PM voltage chart; log abnormal failure margins and cross reference the log to the chart by page number.

Teleprinter –

1. Clean the distributor with Freon and a cotton swab.

2. Perform the complete teleprinter lubrication procedure as prescribed by Teletype Bulletin 281B, Volume 1 (there are four applicable sections; see the Bulletin table of contents).

3. Run the teleprinter test (MAINDEC 614) with margins. Plot the failure points on the PM voltage chart; log abnormal failure margins and cross reference the log to the chart by page number.

Every 2000 Hours

Tape Reader –

1. Check all electrical connections for electrical and mechanical security.

2. Inspect all moving parts for wear.

Card Reader – Sections and figures referenced below are in Burroughs Card Reader Technical Manual B122.51; part numbers specify Burroughs lubricants.

1. Check the following adjustments:

   a. Top and bottom feed roller clearance and feed-roller pivot arm spring tensions (Section 3.1).

   b. Exciter lamp bracket to read-bed plate clearance; light spot registration; side plate to insulation block clearance (Section 3.2).

   c. Top and bottom hopper throat blade clearance (Section 3.3).

   d. Feed-knife to card-column-80 clearance (Section 3.4).

   e. Clutch collar to actuated trip-arm clearance; actuated trip-arm to rear solenoid core clearance; disengaged trip-arm/clutch-collar latching ridge overlap (Section 3.5).

   f. Gap between card deck and front card guide, and slide to left gib clearance (Section 3.6).

   g. Stacker-stop-actuator arm position (Section 3.7).

   h. Reluctance-pickup strobe output pulse duration; pickup tip protrusion and orientation (Section 3.8).

   i. Clutch-reset and feed-test circuit breaker timing (Section 3.9).

2. With S64960-2 oil, lubricate the feed-roll arm pivots (Figures 4.3-1, 2), the circuit breaker contact-arm pivots (Figure 4.4-1), the clutch trip-arm pivot (Figure 3.5-1), and the stacker-switch actuator shaft (Figure 3.7-1).

3. Using S15821-39 grease, lubricate the feed roll shaft gears (Figure 4.3-3), the feed cams (Figure 4.5-1), and the circuit breaker cams (Figure 4.4-1).

4. Using S15821-32 grease, repack the clutch housing and spring (Section 4.5-1).

5. Using S15821-26 oil, lubricate the drive motor bearings.

6. Check the following bearings for wear as evidenced by excessive bearing noise under rotation; replace worn bearings using the procedures of Section 4.

a. Feed-roller shaft bearings

b. Cam-follower roll bearings

c. Pulley bearings (at the clutch assembly)

d. Clutch shaft bearings

e. Card-feed actuator-arm shaft bearings

9.4 CORRECTIVE MAINTENANCE

PDP-6 is constructed of highly reliable, solid state modules, and the possibility of failure during the first year after installation is exceedingly remote. The maintenance staff should use this period to acquire total familiarity with the system and all its documentation — approaching the system in ignorance guarantees frustration. This section contains no fault classification system with specific remedies; instead it outlines a broad plan of attack for isolating malfunctioning hardware. When diagnosis is complete, proper remedial action is manifest. Diagnosis depends on logical thinking, common sense, secure knowledge of the system, and an organized step-by-step procedure.

Do not attempt to memorize the prints; instead concentrate on associating the drawing number mnemonic with the logic section portrayed. When system logic is well understood, the logic drawings and flow charts are generally sufficient reference documentation during troubleshooting. Logic drawings are cross-referenced to the manual by section number (it appears with the figure number in the lower right) allowing recourse to the text in cases of extreme difficulty. In addition to the DEC prints and manuals, there are manufacturer's manuals for the in-out devices. Familiarity with the contents of these manuals avoids time wasted in futile searches for information.

When confronting a malfunction in the machine, the following plan of attack should be employed:

1. Initial investigation: gather all available information on the problem.

2. Preliminary check: see if the malfunction presents obvious physical symptoms.

3. Console troubleshooting: use pertinent diagnostic programs, maintenance controls, marginal check procedures, etc., to localize the problem within a particular section of the logic.

4. Logic troubleshooting: complete the diagnosis by isolating the malfunction within a particular module, supply, or control.

5. Remedial action.

6. Validation of remedy: ensure that proper system function is restored.

7. Log report: make a complete record of diagnosis, remedy, and validation.

Troubleshooting, which is discussed in a below, includes the first four steps above. The diagnosis is expected to suggest a remedy (step 5), but the remedy may require attendant procedures; such situations are treated in b. The final two steps are discussed in c.

a Troubleshooting

This section details the diagnostic procedure to explain how faults may be isolated to individual modules; troubleshooting within modules is discussed in PDP-6 Circuits.

Initial Investigation

Before commencing troubleshooting procedures, explore every possible source of information. Ascertain all possible information concerning any unusual function of the machine prior to the fault and all symptoms evident when the fault occurred, such as the type of program in progress, conditions reflected by console indicators, etc. Search the maintenance log to determine whether the present fault or a similar one has occurred before or whether any cyclic history of

related malfunction exists, to determine how similar past conditions were corrected. Examine the PM voltage charts for any steadily narrowing operating voltage regions under marginal check. If a deteriorating module produces failures seemingly related to the present trouble, diagnosis may be simplified. The more information the technician can gather, the more rapidly he can make a diagnosis. Attempting to troubleshoot the system without first exploring every available source of information usually just wastes time.

Preliminary Check

Troubleshooting begins with a check for physical symptoms of malfunction. If a very large portion of system logic is inoperative, inspect the physical and electrical security of power sources, cables, connectors, etc. Be sure that power supplies provide proper voltages. Check the condition of all filters for free flow of air; a clogged filter may allow the temperature within a bay to rise sufficiently to cause marginal semiconductors to fail. This preliminary check is useful more often for catastrophic malfunctions than for intermittent ones. Except for cable and module connections, most intermittent failures are due to cold-soldered joints or faulty circuit components. Unless the malfunction is isolated with certainty to within two or three modules by the initial investigation, it is poor strategy to start checking arbitrary modules; more sophisticated troubleshooting procedures must be used. Nevertheless, the preliminary check often discloses troubles which would otherwise require considerable diagnostic work. Few things are more annoying than the discovery of a simple power supply failure after complex, time-consuming troubleshooting procedures.

Console Troubleshooting

In many cases the initial investigation discloses an appropriate line of attack, but does not in itself pinpoint the location of a malfunction. Troubleshooting from the console localizes a malfunction within a small section of machine logic, and is usually accompanied by use of the MAINDECs (with or without marginal checking, depending on the nature of the malfunction). Intermittents caused by weakened components can almost always be aggravated and thus transformed to relatively consistent malfunctions by use of marginal checking with an appropriate MAINDEC (Section 9.2), selected on the basis of information derived from the initial investigation. Intermittent connections, however, are substantially more difficult to locate.

If the malfunction does not show up in the first run without margins, perform the operation in which the trouble was initially observed, using the same user program. In-out equipment faults frequently give indications similar to those caused by processor malfunctions. Since the entirely solid state processor is inherently more reliable, check the peripheral equipment first. Faulty ground connections between periphery and processor are common sources of trouble.

If a malfunction is caused by loading conditions unique to a user program, it can generally be detected by a MAINDEC run with margins. Malfunctions caused by differences in loading conditions show up at narrower margins than those listed in the last few plots on the PM voltage charts. If the MAINDEC discovers the error during application of marginal check, do not restart the computer; use the error halt listings to diagnose the trouble.

Console troubleshooting procedures for locating catastrophic malfunctions, or those made consistent through use of margins, should be directed toward discovering a pattern of consistency among the errors. Although some ingenuity may sometimes be necessary to discover it, all malfunctions display consistency in the errors they produce; separate error diagnoses must intersect at a common fault point.

Logic Troubleshooting

Logic troubleshooting is detail work performed on a small section of the logic after a malfunction has been detected but before the source has been identified. The two main parts of troubleshooting are to reproduce the malfunction and then to identify the source. A common technique for reproducing malfunctions is the use of diagnostic loops, though at times it may be necessary to reproduce the exact conditions under which the failure first occurred, e.g., by repeating the program that was running at the time. For an intermittent fault it may be useful to try aggravation techniques while a diagnostic loop is being run.

In general a diagnostic loop is a small string of instructions that causes the computer to perform some operation and checks that it is executed correctly; usually the error is indicated by having the computer halt. If the operation is executed correctly, the program starts over; in this manner the computer runs until a malfunction is encountered. The technician may design his own diagnostic loop and toggle it in from the console, but it is usually easier

to modify the maintenance program. MAINDEC 601 consists entirely of small diagnostic routines seldom containing more than ten instructions. Each routine halts the processor on an error by using JRST 4 with the address pointing to the first loaction of the next routine. Any routine may be converted into a diagnostic loop by replacing the instruction following the halt with a jump to the beginning of the routine.

Aggravation is intended to convert intermittent malfunctions to catastrophic ones. The technique involves either marginal check voltages or vibration (weak component detection by marginal checking is part of console troubleshooting described above). As long as reasonable care is used to avoid inflicting permanent damage, maintenance personnel should not hesitate to twist and probe at connections, cables, plugs or modules. All connections in the system are designed for excellent reliability; so plugs, cables, or sockets should be impervious to any reasonable amount of pulling or flexing. Although PDP-6 systems are tested before leaving the factory, nevertheless, a poorly soldered connection occasionally shows up; this type of fault usually appears as an intermittent and is sometimes very difficult to locate. Poorly soldered connections are more likely to appear in mounting panel or cable connections than within modules. Well hidden malfunctions of this type are occasionally located by tapping suspect modules with the plastic handle of a screwdriver or other such harmless implement.

Once the malfunction can be reproduced, the next step is to identify the source and correct it. Three common techniques are substitution, observing panel indicators, and signal tracing. Substitution is simply replacing a suspect module to see whether a malfunction is thereby cured. Whenever possible, swap modules rather than substituting a spare, e.g., swap counter or register bits. If the malfunction moves to the new location, the trouble is in the exchanged module. If the malfunction has not moved, the trouble is likely to be elsewhere, probably in the logic that feeds the module. Regardless of the outcome of the swap the good module should be returned to its original location – it is a desirable maintenance practice to keep each module in a specific place in the system.

Panel indicators may be used to detect malfunctions by following machine state changes. Using REPEAT or REPEAT BYPASS and the SPEED controls, a program loop can be run at slow speed by pausing after each instruction, memory access, AR subroutine, or even every step in a shift-count. A complete discussion of single step operation is presented in 9.1d.

Repetitive memory access can be produced without using any instructions by repeating an examine or deposit operation from the console. A single instruction may be repeated by setting it in the DATA switches and using REPEAT with the EXECUTE key. A single instruction in memory can be repeated by using START or READ IN. Complete information on the repetition of key functions is given in the final paragraph in 5.1.

Signal tracing requires a scope. With the machine running in an exercise loop, synchronize the oscilloscope sweep to one of the main sequence time pulses or to a pulse that should generate the suspect pulse or event. (The flow charts are very helpful for selecting an appropriate pulse. All pulses and levels are available at module output pins; all pin locations are shown on the logic drawings, and every signal name is prefixed by a mnemonic indicating the source logic for it.) In order that events may be viewed on the scope, the exercise loop should not halt the machine when an error occurs (to change a diagnostic loop into an exercise loop, replace the skip with a jump). It is difficult to pinpoint intermittents using signal tracing, so an effort should be made to change intermittent faults into catastrophic ones which can be located far more easily. When constructing an exercise loop keep it simple and use a minimum of instructions. For example, a simple loop can be obtained by starting the processor at a memory location that contains a jump to itself, or by repeating key functions.

Once the malfunction has been isolated and is presumably corrected, be particularly careful during validation to ensure that the module or component replaced was really at fault, and thus that the malfunction was actually cured by the replacement.

b Repair

Once a malfunctioning module has been located, repair should be made immediately by substitution, as intensive system scheduling requirements preclude use of the processor to aid in submodular troubleshooting. It is expected that defective modules will be returned to DEC for repair or replacement; those who wish to undertake module repair may refer to procedures given in PDP-6 Circuits. When replacing a module, make sure the new one is properly prepared to replace the old: in some cases adjustments must be made or internal connections must be made or broken.

Internal Connections

Most inverter, diode, capacitor-diode, and decoder modules have provision for internal jumpering to connect clamped loads at the output inverter collectors. These modules are shipped with all clamped loads connected; to prepare a replacement module for insertion in any given location the unused jumpers must be disconnected (e.g., with wire cutters). Some modules have jumpers for other purposes; for example some flip-flops, decoders, etc., have jumpers to select the connections to the module pins, and some delay lines require selection of the tap to choose the delay duration. Whether jumpers must be cut or soldered in depends upon the type of module. The utilization module lists (UML, see Appendix 1) give the jumper configurations in all locations for modules in which jumpers may be used. The jumper codes as they appear on the UML depend upon the type of module and the way in which the jumpers are used; these codes are explained in the circuit manual.

Delays

Most delay modules contain distributed-constant delay lines that cannot be adjusted. There are, however, three types of adjustable delays, 1304, 4301, and 4303. Whenever one of these modules is replaced, the new module must be adjusted to give the correct time interval. The 1304 and 4301 may be adjusted as follows:

1. Set up an exercise loop that repeatedly triggers the delay.

2. Set up an oscilloscope to observe the duration of the level output at pin J of the module (−3 volts during the delay interval, ground otherwise).

3. Set the scope sweep to the calibrated position, and select a per-centimeter sweep that displays the entire duration of the level output. The output is adjusted to the required duration by means of a screwdriver trimpot accessible through a hole in the rear of the module frame.

The 4303 has flip-flop type outputs. Use the above method but connect the scope at pin W to observe the negative output, or at pin U to observe the ground output. The delays for

power clear and punch motor turnoff are not critical, so the replacement may not need adjustment at all; the delay period for detecting a nonexistent memory should not exceed 100 μsec. The 4303 in the repeat logic is adjusted by means of the speed controls: the selector chooses one of five external capacitors or none at all (the smallest is internal); the potentiometer varies the available charge current for the selected capacitor.

Tape Reader

When replacing any reader circuit card having a trimpot adjustment, perform the entire 1000-hour PM checkout (and all checks subordinate to it). This action not only provides for the required card adjustment but also serves as a validation procedure for the repair.

Card Reader

Replacing any photoamplifier package in the card reader requires complete readjustment of the exciter lamp brilliance and of the strobe pulse coincidence, using the procedures of Sections 3.16 and 3.17 of the card reader manual. Replacement of any mechanical part that involves an adjustment requires performance of the adjustment procedure as part of the reassembly; the replacement procedures of Section 4 in the reader manual should be followed by appropriate calibration procedures from Section 3.

c Validation and Log Entry

Following replacement of any electrical component in the system, tests should be performed to ensure correction of the fault condition and to trim up final adjustments or signal levels affected by the replacement. Validation normally requires performance of the PM procedure most applicable to that portion of the system in which the fault occurred. If repair or replacement is made in an area not normally checked during preventive maintenance, the appropriate diagnostic program (MAINDEC) should be run, or an alternate operational test should be devised to ensure proper operation. It is strongly suggested that the entire preventive maintenance task be performed for the malfunctioning system component. This action ensures detection of faults that may have been masked by the just-corrected malfunction and may reveal a failure mechanism, perhaps still extant elsewhere, that is likely to cause recurrence of the fault. If the entire PM procedure can be performed while the equipment is down and available, it need not be scheduled for the whole PM interval.

Corrective maintenance activities are not complete until they are recorded in detail in the maintenance log. Include all data indicating symptoms given by the fault, the method of fault detection, the component at fault, and any comments that might be helpful in future diagnosis. Suggest improvements in PM procedures if indicated. A complete and detailed write-up of diagnostic procedures used to locate a real dog is particularly important. Although some details may seem insignificant and obvious, they should nevertheless be included since examination of the log will be a major part of the initial investigation for a future malfunction. Reduce down time by recording everything.

APPENDIX 1

ENGINEERING DRAWINGS

Reduced copies of engineering drawings are included in Volume 2 of this manual. Chapter 4 explains the drawing numbers and type codes that identify engineering drawings for all components in a PDP-6 system and details the notation and conventions used in the block schematics and flow charts, which are the basis for learning and maintaining the equipment. However, there are many other engineering drawings used primarily for reference in maintenance. For a given system component, every drawing number has the type number of that component following the size letter; drawings that apply to the entire system are identified by "6" as the type number. In the following discussion all drawings mentioned are D size unless otherwise specified; the type of drawing in each case is identified by the code, and individual drawings are specified by the number or mnemonic code that follows the drawing serial number.

The master drawing list (MDL) for the entire system is an A-6 drawing with no individual drawing number. It lists the MDLs for all system components; these are also A size and identified only by the equipment type number and drawing serial number. Each MDL lists all drawings for the system component by title, number, code, revision letter, and number of sheets. The only other drawing for the entire system is a module list ML; in this instance the system module list SML, which lists the module requirements for all system components (this drawing is discussed further in Appendix 2).

The 166 MDL lists first the SD, FDs, and BSs that accompany this manual. Detailed information about the modules that make up the processor is shown in the utilization module lists UML, of which there are ten, numbered 14 to 23. Each UML shows three mounting panels, each divided into 25 sections representing the plug-in locations. Above each location is the type number and jumper code for the module occupying it. Each location is further partitioned according to the individual circuits (flip-flops, pulse amplifiers, inverters) in the module. Circuits are identified by logical function, using the same signal names that appear on the block schematics. For example, a pulse amplifier is labeled by naming the output pulse; an inverter or diode gate, by the output of the net in which it is used.

Of the various cable diagrams, CD, the one that shows the logical signals on the in-out and memory busses is included in the manual (IOMB, Figure 7-10). Another CD that is closely associated with the logic is CONS, which shows the connections to all operating keys and switches on the console. This drawing shows the generation of logic levels from the keys and switches and the connections to the main power control including the 60-cycle signal for the clock flag. Note how the on position of the key-locked console disabling switch holds system power on but disrupts the console power that enables the keys and switches. Note also that the NO and NC labels on switch contacts bear no invariant relationship to the normal position of the switch; these are merely the labels molded onto the switch case for physical identification of connections.

A pair of drawings CBL-2 and CBL-3 show all levels from the logic to indicators and switch registers on the console and on the bay indicator panels. CBL-1 shows the locations of all plugs and jacks throughout the four bays; CH shows the configuration of the main cable harness and its many branches. There are also a large number of cable lists CL that show the connections made from cables to module connector pins and identify the signals carried on the various cable lines.

All connections made among the module connector pins are shown in a series of wiring diagrams WD. Each drawing shows three mounting panels and is identified by the panel letters. Because of the complexity of the wiring in the processor, each set of three panels is shown in two drawings, one showing for example the grounds and pulses, the other the levels and flip-flops outputs (there are also other combinations).

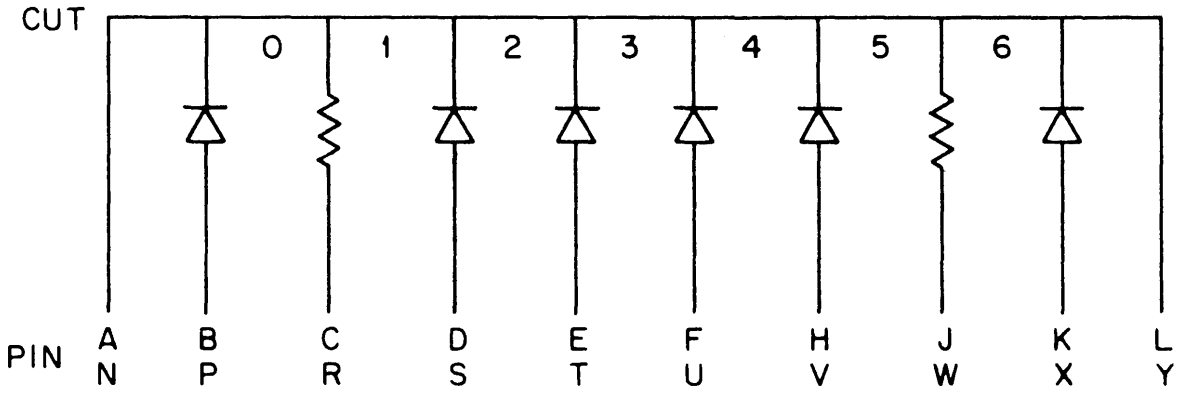
Power wiring is shown in the PW drawings, ACPW and DCPW. Each shows the location of power controls and power supplies on the inside of the rear plenum doors viewed from the front, i.e., as though one were looking right through the mounting panels from the wiring side. ACPW shows all ac connections from the power controls and fans (for machines using 240 volts, 50 cycles, drawing 50ACPW is substituted). The interlock circuit in the lower right is associated with the plenum doors in bays 3 and 4 and may be used for example in the high-voltage line to a display mounted in the console. The interlock switches have three positions including an override the technician may use during maintenance. DCPW shows all of the dc wiring from the supplies to the terminal strips. All wiring is color coded +10 red, −15 blue, ground black. As shown in

the lower left, all terminal strips and the terminals on the 728 and 734 Supplies use the same color code as the wiring; the terminals on the 778 Supply are different but the wires connected to them use the standard coding. The strip between the pair of representations for each bay is a power connector bracket, whose layout and wiring are shown at the lower left and right. These are mounted horizontally about halfway up the sides of the bays. The smaller horizontal strips in the right block for each bay are actually vertical and located at one end of every mounting panel. The green and yellow terminals and wires carry the positive and negative variable voltages for marginal check (the wires are color coded identically). In the upper right are the connections to the marginal-check control panel and the marginal-check bus. This bus carries the variable voltages for marginal checking to other equipment at the site and also carries the −15-volt turnon signal for the memories. A wiring diagram, MCLT, shows the connections from the controls on the marginal-check panel to the variable power supply at the rear of the same bay. Connections to the elapsed time meter are at the right.

Many processor drawings, such as the PWs, also present information about the in-out control units included in this manual. However, each device has its own UML, WD, and various other cable drawings as listed in the appropriate MDL. For the card reader there are a number of Burroughs logic schematics not included in the card reader manual. These include drawings H-1182627 to H-1182640, H-11900107, H-11900115, H-11877446, H-11877453, and H-11877461. The last three are lists of equations for the reader logic; the others are computer-printed block schematics. These schematics show signal flow and give pin numbers and part designations of electrical components, plug-in units, and connectors, all of which are shown by rectangles. Wires are indicated by dashed lines; a "0" at an intersection point of two wires represents a common connection. The information is laid out on a rectangular coordinate system, with all inputs to the drawing in the input column at the left, all outputs at the right. The columns between are numbered and all horizontal rows are identified by a letter and a number. All input signals are identified by at least two rows of information, the top row giving the signal name, the bottom row indicating its source by page and row. Supplementary data may be included between the top and bottom rows. Output information is identified in the same way, giving the destination of the signal by page and row.

All boxes are eight character positions wide. The first two positions are used exclusively for the pin connections of input lines, and in all but the bottom row, the final two positions are

used for pin connections of output lines. At the top center of each box is a part designation, and the bottom row gives the location. Section 2.2 of the card reader manual explains the system used to designate the locations of all electrical parts. The location of a plug-in unit is given in the form AAB5L6, which means that the unit is in rack A, panel A, row B, position 5 and that its key pin (the one larger than the others) plugs into row L, column 6 of the socket. The part designation is usually the initial letters of the words describing the part or some standard symbol such as K for relay; some of the part designations are given in Burroughs Section 2.5, and wherever the meaning of the part designation cannot be deduced, the location designation can be used to find and thus identify the part. If the box represents a pluggable unit, such as a photo amplifier, low-speed switch, or diode stick, its location also appears in Table 2.5-1. A schematic and description of the part can be found in Chapter 6. The majority of the plug-in units are diode sticks which are not described, but are all identical and are wired as follows:



A diode stick is designated in the drawings by D followed by a letter or number. It may be cut to form two or three diode gates: the top, center, or bottom part of the stick is represented by the box if a T, C, or B follows the D. A number indicates the position of the cut or cuts made (no cut is indicated by NC). Replacements are sent uncut, and the technician must cut them to conform with the configurations required for any given location. In addition to pin connections, the remaining rows may contain other information. Row 2 often indicates the part of the logic the circuit is used in. For a diode stick, rows 3 and 6 give information about the resistors at positions 1 and 6 and the gates associated with them: an R in position 3 indicates

that the resistor is tied to −12 volts; a numeral in position 4 indicates the number of diodes used in the gate; A or O in position 5 or 6 indicates the use of the gate as logical AND or OR. Information contained in other rows depends upon the element represented; for example in a box representing a relay, NO or NC in row 5 indicates normally open or normally closed.

APPENDIX 2

SPARES

For a large-scale system such as PDP-6, even moderate downtime is generally far more expensive than the cost of maintaining a fully adequate stock of spares. The system module list SML lists the quantities of all modules, power supplies and power controls used in all system components (i.e., processors, memories, in-out control units) available in PDP-6. However, individual systems may differ considerably in quantity and type of system components, so DEC Field Service supplies each user with a list geared to his particular system. This list gives the total quantity and number of spares recommended for every module, power supply, and power control. Field Service also supplies spares lists of electrical components needed for repair of DEC modules and lists of electrical and mechanical spares for the various in-out devices. It is further recommended that the user keep a spare tape reader, tape punch, and keyboard-printer, and stock the following miscellaneous items for the system.

Quantity	Item	Part Number	Vendor
1	Switch for bat-handle key	Telever Switch 16006 Cat. No. S-302 Reworked per DEC drawing MA-C-01510	Switchcraft Inc.
5	Subminiature toggle switch SPDT	6AT1	Micro Switch
1	Lockout switch DPDT	1575-L	Arrow Hart
2	Rotron fan	53E168 Type CFG	Rotron Mfg. Co.
2	Rotron filter	34-X1431	Rotron Mfg. Co.
10	Indicator lamp	MC48-639B	Transistor Electric
1	Howard fan	12-80-15	Howard Industries

APPENDIX 3

GLOSSARY

It is not intended that use of this glossary shall be substituted for reading Chapter 4 or for using the flow charts when following any sequence of events in the logic drawings. This glossary gives the meanings of all prefix codes and all terms used in the signal names for the processor and the four in-out devices. Only a few complete signal names are included because all names are composites of standard terms, and their meanings are usually quite obvious to anyone who knows the meanings of the terms and has read 4.2. No generating conditions are given for logical functions because these are listed in the flow charts. The processor flags are listed and described in 2.1c, and all inputs to the logic from the console are discussed in 3.1a. With each prefix code below are listed in parentheses the numbers of the figures to which the code applies.

A (5-3)	Address cycle.
AC	Accumulator.
AC0	Accumulator whose address is 0.
AC2	Accumulator following the one addressed by an instruction.
ACBM (5-9)	Accumulator bit modify; the logical compare instructions, which test bits of AC as specified by a mask (either E or C(E)) and may modify the masked bits.
ACCP (5-9)	Accumulator compare, i.e., those arithmetic compare instructions that compare AC with either E or C(E).

ACCP ET AL TEST ACCP ETC COND	ACCP ETC COND is the skip or jump condition as specified by bits 7 and 8 in ACBM, ACCP, or MEMAC. Since bit 6 of the instruction code specifies whether the skip or jump is to be made on the presence or absence of the condition specified by bits 7 and 8, the condition signal is exclusive ORed with IR6 to generate ACCP ET AL TEST, which is sensed by the program control logic.
ACK	Acknowledge.
ADDR	Address.
A LONG	Flip-flop set whenever any operations are performed in the calculation of an address.
AOBJP	Add one to both and jump if positive.
AOBJN	Add one to both and jump if negative.
AR (6-4, 5, 6, 7, 8, 9, 10)	Arithmetic register and associated control logic, special inputs, subroutines, and flags.
AR COM CONT	AR complement control flip-flop; set at the beginning of subtraction or decrementing in AR to indicate that result must be complemented following the addition or incrementing used.
AR CRY COMP	AR carry complete.
AR CRY0	Flip-flop set by any carry out of AR0.

AR CRY1	Flip-flop set by any carry out of AR1.
AR = FP HALF	Level asserted when the fractional part of AR as a floating-point number is of magnitude 1/2.
AR OV SET	Condition that determines overflow in AS or MEMAC.
AR SBR	Level that causes the execute cycle to pause at ET4 for an AR subroutine.
AS (5-10)	Add-subtract.
ASH	Arithmetic shift.
ASHC	Arithmetic shift combined.
B	Binary; buffered.
BLK	Block.
BLKI	Block in.
BLKO	Block out.
BLT (6-18)	Block transfer.
BLT DONE	Level that terminates a block transfer either permanently because the block is complete or temporarily to handle an interrupt.
BLT LAST	Inhibits AC storage when block transfer is complete.
BOOLE (5-10)	Boolean instructions.

BOTH	Mode in which result is deposited in both AC and memory.
CAO	Character add one (= IBP).
CBHL	Level to the card reader causing it to place the upper half of a binary column on the output lines.
C( )	Contents of.
C(E)	Contents of location specified by effective address.
CFAC (6-17)	Computer floating-arithmetic connection (subroutine interface).
CH (6-19, 20)	As a prefix, character operations (byte manipulation); also channel.
CHG	Change.
CH INC CH INC OP CH ∿INC OP	These three levels control the first part in a character operation. CH INC is asserted during the first part of any instruction that calls for incrementing the pointer. The other two levels are mutually exclusive within the first part of a character operation. CH INC OP causes incrementing of the pointer; CH ∿INC OP inhibits incrementing either because the instruction does not require it or the first part of an incrementing character operation is being repeated following a priority interrupt between the two parts.

CL	Clear.
CLK	Clock.
CLR	Clear.
CMC	Core memory control; prefix for control signals from memory (core or fast) over the bus.
COM	Complement.
COMP	Complete.
CONI	Conditions in.
CONO	Conditions out.
CONSO	Conditions in and skip if one.
CONSZ	Conditions in and skip if zero.
CONT	Control; continue.
CPA (8-5)	Processor I/O interface. This is the interface at the other end of the bus from IOT control; through it the processor, via IOT control and the bus, controls itself as a device.
CR (8-13, 14, 15)	Card reader.
CREL	Signal from the card reader that indicates a validity or read check error.
CRL	Signal from the card reader that indicates it is not ready for operation.

CRY	Carry.
CYC	Cycle.
DATAI	Data in.
DATAO	Data out.
DC (6-20)	Deposit character.
DCA	Dc adder.
DEP	Deposit.
DIR	Direct, i.e., C(E) rather than E.
DIV	Divide.
DN	Do nothing.
DPC	Deposit character (= DPB).
DPCI	Index pointer and deposit character (= IDPB).
DS (6-25, 26)	Divide subroutine.
E (5-5)	As a prefix, execute cycle; also effective address.
E LONG	Level that causes processor to use second half of execute cycle.
EOF	End of file.
EX (5-13)	As a prefix, executive mode; also examine.
EXCH	Exchange.

EXEC	Execute.
F (5-4)	As a prefix, fetch cycle; also flip-flop.
FA (6-22)	Floating add.
FAD	Floating add.
FC(C(ACLT))	Fetch the contents of the location addressed by the number in ACLT.
FC(C(ACRT))	Fetch the contents of the location addressed by the number in ACRT.
FD (6-22)	Floating divide.
FDV	Floating divide.
FE (6-14, 15)	Floating exponent.
FM (6-22)	Floating multiply.
FMC	Fast memory control.
FMP	Floating multiply.
FP (6-23)	As a prefix, floating-point exponent calculate subroutine; also floating-point instructions.
FP/CH	IR decoder output for the floating-point instructions and character operations.
FS (6-19)	Floating scale.
FSC	Floating scale.

FSB	Floating subtract.
FWT (5-9)	Full-word transfer.
HWT (5-9)	Half-word transfer.
I (5-3)	Instruction cycle.
ILL	Illegal.
ILLEG	Illegal.
INC	Increment.
INH	Inhibit.
INST	Instruction.
INT	Interrupt.
IO	In-Out.
IOB (8-1, 2)	In-out bus; IOBi (i = 0, ..., 35) is data line i on the bus.
IOS (5-7)	In-out select.
IOT (8-1)	In-out transfer.
IOT A	IR decoder output for IOT (replaced by a UUO in user mode).
IR (5-7, 8)	Instruction register and associated decoding nets.
JFCL	Jump on flag and clear.
JP (5-10)	Jump and pushdown.

JRA	Jump and restore accumulator.
JRST	Jump and restore.
JRST A	IR decoder output for JRST (replaced by a UUO in user mode if a halt or PI dismiss).
JSA	Jump and save accumulator.
JSP	Jump and save program counter.
JSR	Jump to subroutine.
K (5-2)	Key cycle.
KEY (5-1, 2)	Key.
KEY MANUAL	Level generated when any initiating key is pressed.
LC (6-20)	Load character.
LDC	Load character (= LDP).
LDCI	Increment pointer and load character (= ILDP).
LSB	Least significant bit.
LSH	Logical shift.
LSHC	Logical shift combined.
LT	Left.
MA (7-1, 2, 3)	Memory address.
MAI (7-2)	Memory address interface.

MAJ	Majority gate.
MARK	Teletype signal representing a 1.
MAS	Address switch.
MA SW	Address switch.
MB (6-1, 2, 3, 4; 7-6)	Memory buffer and associated control logic, special inputs, and data interface with memory bus.
MC (7-8, 9)	Memory control.
MD	Multiply-divide.
MEM	Memory; as a mode, result is stored only in E.
MEMAC (5-10)	Memory and accumulator modification and test instructions; these test AC against zero for a jump, and may or may not increment AC, or test C(E) against zero for a skip, and may or may not increment C(E).
MI (7-7)	Memory indicators.
MISC BITS	Miscellaneous bits.
MOVN, M	Move negative or magnitude.
MOV, S	Move or move and swap.
MP (6-21)	Multiply.
MQ (6-7, 11,12, 13)	Multiplier quotient register and associated control logic and special inputs.

MQ36	Extra MQ bit for use in multiply.
MR (5-1, 2)	Master.
MR CLR	Pulse that prepares processor for each main sequence or console function.
MR START	Pulse that clears entire computer at power turnon or when IO RESET is pressed.
MS (6-24)	Multiply subroutine.
MSB	Most significant bit.
MUL	Multiply.
MULT	Multiply.
NEGATE	Form the arithmetic 2's complement.
NR (6-27)	Normalize return.
NXT	Next.
OP	Operation.
OV	Overflow.
PC (5-11, 12)	Program counter and control.
PDL OV	Pushdown list overflow.
PI (8-3, 4)	Priority interrupt.
PIA	Priority interrupt assignment.
PICH	PI channel address.

PIH	PI hold.
PIO	PI on.
PIR	PI request.
POP	Pullout.
POPJ	Pullout and jump.
PR (7-4)	Protection.
PSE	Pause.
PTP (8-9, 10)	Paper tape punch.
PTR (8-6, 7, 8)	Paper tape reader.
PUSH	Pushdown.
PUSHJ	Pushdown and jump.
PWR	Power.
PWR CLR	Pulses that clear computer at power turnon.
RD	Read.
REL	Relocation.
REM	Remainder; floating-point mode that stores low-order half of result in AC2.
REQ	Request.
RIM	Readin mode.
RLA (7-5)	Relocation adder.

RLR (7-5)	Relocation register.
ROT	Rotate.
ROTC	Rotate Combined.
RPT	Repeat.
RQ	Request.
RS	Restart.
RST	Restore.
RT	Right.
RUN (5-1)	Run.
S (5-6)	Store cycle.
SBR	Subroutine; also a mnemonic for subroutine card 1260.
SC (6-14, 15, 16)	Shift counter.
SEL	Select.
SH (6-20)	Shift.
SH AC2	Double-length shift.
SHC (6-7)	Shift connection.
SPACE	Teletype signal for 0.
SR	Shift register.
ST	Start.

STATUS	Any IOT instruction that examines status, i.e., CONI, CONSO or CONSZ.
STB	Strobe.
SW (5-1)	Switch.
SWAP	Interchange the left and right halves of a word.
SUB	Subtract.
T	Time.
TST	Test.
TTI (8-11, 12)	Teletype input.
TTO (8-11, 12)	Teletype output.
TTY (8-11, 12)	Teletype.
UUO (5-10)	Programmed operator (unused op code); a UUO is performed when one is programmed or when it replaces an illegal user instruction.
UUO A	IR decoder output for UUO.
WR	Write.
XCT (5-10)	Execute.
2XX	Level indicating that IR contains an octal code beginning with 2.
25X	Level indicating that IR contains an octal code beginning with 25.

APPENDIX 4

INSTRUCTION CODES

Octal	Mnemonic	Octal	Mnemonic	Octal	Mnemonic
000		172	FDVM	242	LSH
:	UUO	173	FDVB	243	
:		174	FDVR	244	ASHC
077		175	FDVRL	245	ROTC
100		176	FDVRM	246	LSHC
:		177	FDVRB	247	
:		200	MOVE	250	EXCH
131		201	MOVEI	251	BLT
132	FSC	202	MOVEM	252	AOBJP
133	IBP	203	MOVES	253	AOBJN
134	ILDB	204	MOVS	254	JRST
135	LDB	205	MOVSI	255	JFCL
136	IDPB	206	MOVSM	256	XCT
137	DPB	207	MOVSS	257	
140	FAD	210	MOVN	260	PUSHJ
141	FADL	211	MOVNI	261	PUSH
142	FADM	212	MOVNM	262	POP
143	FADB	213	MOVNS	263	POPJ
144	FADR	214	MOVM	264	JSR
145	FADRL	215	MOVMI	265	JSP
146	FADRM	216	MOVMM	266	JSA
147	FADRB	217	MOVMS	267	JRA
150	FSB	220	IMUL	270	ADD
151	FSBL	221	IMULI	271	ADDI
152	FSBM	222	IMULM	272	ADDM
153	FSBB	223	IMULB	273	ADDB
154	FSBR	224	MUL	274	SUB
155	FSBRL	225	MULI	275	SUBI
156	FSBRM	226	MULM	276	SUBM
157	FSBRB	227	MULB	277	SUBB
160	FMP	230	IDIV	300	CAI
161	FMPL	231	IDIVI	301	CAIL
162	FMPM	232	IDIVM	302	CAIE
163	FMPB	233	IDIVB	303	CAILE
164	FMPR	234	DIV	304	CAIA
165	FMPRL	235	DIVI	305	CAIGE
166	FMPRM	236	DIVM	306	CAIN
167	FMPRB	237	DIVB	307	CAIG
170	FDV	240	ASH	310	CAM
171	FDVL	241	ROT	311	CAML

Octal	Mnemonic	Octal	Mnemonic	Octal	Mnemonic
312	CAME	367	SOJG	444	EQV
313	CAMLE	370	SOS	445	EQVI
314	CAMA	371	SOSL	446	EQVM
315	CAMGE	372	SOSE	447	EQVB
316	CAMN	373	SOSLE	450	SETCA
317	CAMG	374	SOSA	451	SETCAI
320	JUMP	375	SOSGE	452	SETCAM
321	JUMPL	376	SOSN	453	SETCAB
322	JUMPE	377	SOSG	454	ORCA
323	JUMPLE	400	SETZ	455	ORCAI
324	JUMPA	401	SETZI	456	ORCAM
325	JUMPGE	402	SETZM	457	ORCAB
326	JUMPN	403	SETZB	460	SETCM
327	JUMPG	404	AND	461	SETCMI
330	SKIP	405	ANDI	462	SETCMM
331	SKIPL	406	ANDM	463	SETCMB
332	SKIPE	407	ANDB	464	ORCM
333	SKIPLE	410	ANDCA	465	ORCMI
334	SKIPA	411	ANDCAI	466	ORCMM
335	SKIPGE	412	ANDCAM	467	ORCMB
336	SKIPN	413	ANDCAB	470	ORCB
337	SKIPG	414	SETM	471	ORCBI
340	AOJ	415	SETMI	472	ORCBM
341	AOJL	416	SETMM	473	ORCBB
342	AOJE	417	SETMB	474	SETO
343	AOJLE	420	ANDCM	475	SETOI
344	AOJA	421	ANDCMI	476	SETOM
345	AOJGE	422	ANDCMM	477	SETOB
346	AOJN	423	ANDCMB	500	HLL
347	AOJG	424	SETA	501	HLLI
350	AOS	425	SETAI	502	HLLM
351	AOSL	426	SETAM	503	HLLS
352	AOSE	427	SETAB	504	HRL
353	AOSLE	430	XOR	505	HRLI
354	AOSA	431	XORI	506	HRLM
355	AOSGE	432	XORM	507	HRLS
356	AOSN	433	XORB	510	HLLZ
357	AOSG	434	IOR	511	HLLZI
360	SOJ	435	IORI	512	HLLZM
361	SOJL	436	IORM	513	HLLZS
362	SOJE	437	IORB	514	HRLZ
363	SOJLE	440	ANDCB	515	HRLZI
364	SOJA	441	ANDCBI	516	HRLZM
365	SOJGE	442	ANDCBM	517	HRLZS
366	SOJN	443	ANDCBB	520	HLLO

Octal	Mnemonic	Octal	Mnemonic	Octal	Mnemonic
521	HLLOI	571	HRREI	641	TLC
522	HLLOM	572	HRREM	642	TRCE
523	HLLOS	573	HRRES	643	TLCE
524	HRLO	574	HLRE	644	TRCA
525	HRLOI	575	HLREI	645	TLCA
526	HRLOM	576	HLREM	646	TRCN
527	HRLOS	577	HLRES	647	TLCN
530	HLLE	600	TRN	650	TDC
531	HLLEI	601	TLN	651	TSC
532	HLLEM	602	TRNE	652	TDCE
533	HLLES	603	TLNE	653	TSCE
534	HRLE	604	TRNA	654	TDCA
535	HRLEI	605	TLNA	655	TSCA
536	HRLEM	606	TRNN	656	TDCN
537	HRLES	607	TLNN	657	TSCN
540	HRR	610	TDN	660	TRO
541	HRRI	611	TSN	661	TLO
542	HRRM	612	TDNE	662	TROE
543	HRRS	613	TSNE	663	TLOE
544	HLR	614	TDNA	664	TROA
545	HLRI	615	TSNA	665	TLOA
546	HLRM	616	TDNN	666	TRON
547	HLRS	617	TSNN	667	TLON
550	HRRZ	620	TRZ	670	TDO
551	HRRZI	621	TLZ	671	TSO
552	HRRZM	622	TRZE	672	TDOE
553	HRRZS	623	TLZE	673	TSOE
554	HLRZ	624	TRZA	674	TDOA
555	HLRZI	625	TLZA	675	TSOA
556	HLRZM	626	TRZN	676	TDON
557	HLRZS	627	TLZN	677	TSON
560	HRRO	630	TDZ	7-00	BLKI
561	HRROI	631	TSZ	7-04	DATAI
562	HRROM	632	TDZE	7-10	BLKO
563	HRROS	633	TSZE	7-14	DATAO
564	HLRO	634	TDZA	7-20	CONO
565	HLROI	635	TSZA	7-24	CONI
566	HLROM	636	TDZN	7-30	CONSZ
567	HLROS	637	TSZN	7-34	CONSO
570	HRRE	640	TRC		

APPENDIX 5

TELETYPE CODE

The 8-bit codes are listed below. An asterisk indicates a code that has no effect on the Model 35. Alternate characters are listed in parentheses. The characters actually contain only seven information bits. The eighth bit may be used for parity, but currently all machines are set up so that the eighth bit is a mark, and thus the codes generated from the keyboard are 200_8 greater than the corresponding ASCII codes.

Octal Code	ASCII Character	Key Combination	Remarks
200	NULL	SHIFT CTRL P	Null.
201*	SOM	CTRL A	Start of message.
202*	EOA	CTRL B	End of address.
203*	EOM	CTRL C	End of message.
204	EOT	CTRL EOT	End of transmission; shuts off TWX machines.
205	WRU	CTRL WRU	"Who are you?" Triggers "Here is...," at remote station.
206*	RU	CTRL RU	"Are you...?"
207	BELL	CTRL BELL	Rings the bell.
210*	FE	CTRL H	Format effector.
211	HT	CTRL TAB	Horizontal tab.
212	LF	LINE FEED	Line feed.
213	V TAB	CTRL VT	Vertical tab.
214	FF	CTRL FORM	Form feed.
215	CR	RETURN	Carriage return.
216*	SO	CTRL N	Shift out.
217*	SI	CTRL O	Shift in.
220*	DC0	CTRL P	Device control reserved for data linc escape.

Octal Code	ASCII Character	Key Combination	Remarks
221	DC1	CTRL Q	Turns reader on.
222*	DC2	CTRL TAPE	Turns punch on.
223	DC3	CTRL XOFF	Turns reader off.
224*	DC4	CTRL ~~TAPE~~	Turns punch off.
225*	ERR	CTRL U	Error.
226*	SYNC	CTRL V	Synchronous idle.
227*	LEM	CTRL W	Logical end of media.
230*	S0	CTRL X	Separator, information.
231*	S1	CTRL Y	Separator, data delimiter.
232*	S2	CTRL Z	Separator, words.
233*	S3	SHIFT CTRL K	Separator, groups.
234*	S4	SHIFT CTRL L	Separator, records.
235*	S5	SHIFT CTRL M	Separator, files.
236*	S6	SHIFT CTRL N	Separator, miscellaneous.
237*	S7	SHIFT CTRL O	Separator, miscellaneous.
240	Space	Space bar	
241	!	SHIFT !	
242	"	SHIFT "	
243	#	SHIFT #	
244	$	SHIFT $	
245	%	SHIFT %	
246	&	SHIFT &	
247	´(')	SHIFT ´(')	
250	(	SHIFT (	
251	)	SHIFT)	
252	*	SHIFT *	
253	+	SHIFT +	
254	,	,	
255	-	-	
256	.	.	

Octal Code	ASCII Character	Key Combination	Remarks
257	/	/	
260	ø	0	Zero, prints with a slash
261	1	1	
262	2	2	
263	3	3	
264	4	4	
265	5	5	
266	6	6	
267	7	7	
270	8	8	
271	9	9	
272	:	:	
273	;	;	
274	<	SHIFT <	
275	=	SHIFT =	
276	>	SHIFT >	
277	?	SHIFT ?	
300	↖(@)	SHIFT ↖ (@)	
301	A	A	
302	B	B	
303	C	C	
304	D	D	
305	E	E	
306	F	F	
307	G	G	
310	H	H	
311	I	I	
312	J	J	
313	K	K	
314	L	L	

Octal Code	ASCII Character	Key Combination	Remarks
315	M	M	
316	N	N	
317	O	O	
320	P	P	
321	Q	Q	
322	R	R	
323	S	S	
324	T	T	
325	U	U	
326	V	V	
327	W	W	
330	X	X	
331	Y	Y	
332	Z	Z	
333	[	SHIFT K	
334	¬ (\)	SHIFT L	
335	]	SHIFT M	
336	∧ (↑)	SHIFT ∧ (↑)	
337	_ (←)	SHIFT _ (←)	
340-373*			Lower case letters; codes cannot be generated from keyboard and should not be used in programs for reasons of compatability.
374*	ACK		Acknowledge; code cannot be generated from keyboard and should not be used in programs for reasons of compatability.
375*	①	ALT MODE	May be used for any desired control purpose.
376*	ESC		Escape; code cannot be generated from keyboard and should not be used in programs for reasons of compatability.

Octal Code	ASCII Character	Key Combination	Remarks
377*	DEL	RUB OUT	Delete
		REPT	Causes any other key that is struck to repeat continuously until REPT is released.
		LOC LF	Local line feed.
		LOC CR	Local carriage return.
		BRK RLS	Not connected.

APPENDIX 6

CARD READER CODE

6-bit Code	Character	Column Punch	6-bit Code	Character	Column Punch
00		Any invalid	40	-	11
01	1	1	41	J	11 1
02	2	2	42	K	11 2
03	3	3	43	L	11 3
04	4	4	44	M	11 4
05	5	5	45	N	11 5
06	6	6	46	O	11 6
07	7	7	47	P	11 7
10	8	8	50	Q	11 8
11	9	9	51	R	11 9
12	0	0	52		11 0
13	= [#]	8 3	53	$	11 8 3
14	' [@]	8 4	54	*	11 8 4
15		8 5	55		11 8 5
16		8 6	56		11 8 6
17		8 7	57		11 8 7
20	Space	None	60	+ [&]	12
21	/	0 1	61	A	12 1
22	S	0 2	62	B	12 2
23	T	0 3	63	C	12 3
24	U	0 4	64	D	12 4
25	V	0 5	65	E	12 5
26	W	0 6	66	F	12 6
27	X	0 7	67	G	12 7
30	Y	0 8	70	H	12 8
31	Z	0 9	71	I	12 9
32		0 8 2	72		12 0
33	,	0 8 3	73	.	12 8 3
34	([%]	0 8 4	74	) [□]	12 8 4
35		0 8 5	75		12 8 5
36		0 8 6	76		12 8 6
37		0 8 7	77		12 8 7

Invalid Punch Combinations

0 1 11	0 1 12	1 2	2 3	3 5	5 6
0 2 11	0 2 12	1 3	2 4	3 6	5 7
0 3 11	0 3 12	1 4	2 5	3 7	5 9
0 4 11	0 4 12	1 5	2 6	3 9	6 7
0 5 11	0 5 12	1 6	2 7	4 5	6 9
0 6 11	0 6 12	1 7	2 8*	4 6	7 9
0 7 11	0 7 12	1 8	2 9	4 7	8 9
0 8 11	0 8 12	1 9	3 4	4 9	11 12
0 9 11	0 9 12				

*except 2 8 0

```
$Id: pdp6-serials,v 1.7 2013/07/14 01:31:12 phil Exp $

DEC PDP-6 Serial numbers
Phil Budne <phil@ultimate.com>

26 Installed [DIGITAL Computing Timeline: 8/18/1997]
        tradition has said 23?!

"Unlucky #7 never worked" [hurley]

Invoice dates from DEC "Analysis of PDP-6 Invoiced Equipment" JPH 12/14/65
        The document seems to relates to core memory royalties.

First delivery: summer 1964 [Bell ACM article]

#1?     DEC Engineering

#2?     http://zurich.ai.mit.edu/pipermail/rrrs-authors/1986-July/000508.html
        http://www.ai.mit.edu/people/tk/pdp6/pdp6.html
        MIT Project MAC (AI)
        Later a slave device to MIT-AI KA10
        Invoice dates 12/24/64, 3/26/65
        Empty shell went to Sweden?

#3      Brookhaven National Labs (Trailer)
        Original Invoice dates 1/21/65, 2/8/65 (cancelled?)
        Invoice date 10/29/65
                second sold? [hurley]
        Bob Clements was the on-site field-service engineer
                [alt.sys.pdp10 11/13/2008]

#4?     University of Western Australia, Perth
        Invoice date 5/18/65
                (first sold) [hurley]

        NOTE: UWA APRN ==3 in 4.x sources at pdp-6.trailing-edge.com

        http://ftp.qut.edu.au/pub/info/greet.txt
                installed march 1969

        http://www.terrigal.net.au/~acms/z0104.htm
                some parts

        http://www.iucr.org/iucr-top/people/maslen.htm
                a Digital Corporation PDP-6 (serial number 0004), which was
                used to control one of the first four-circle diffractometers
                in Australia, a Hilger & Watts Y-231, in 1967

        http://www.aceware.iinet.net.au/acms/HistoryOfAcmsWA.htm
                340 display at Australian Computer Museum Society
                (in a barn, Mundaring, Western Australia)

        http://www.terrigal.net.au/~acms/z0104.htm

        http://www.alex-reid.com/Personal/Welcome-7.html

        My first Time-Shared computer - the Digital Equipment Corp PDP-6.
        This was delivered to the University of WA in May 1965, and became the first
commercially-delivered time-shared computer anywhere in the world.
        Pictured before it left the factory in Maynard, Massachussetts:
```

http://www.alex-reid.com/Personal/PDP-6-Maynard.JPG

Pictured on delivery, with Dennis Moore sweeping up:
http://www.alex-reid.com/Personal/PDP-6-Dennis-broom.JPG

Pictured towards the end of its useful life (1972), with Dennis Moore and Yow Kwan:
http://www.alex-reid.com/Personal/PDP-6.jpg

http://www.decodesystems.com/pdp6.html
The first PDP-6 time-sharing system produced for sale is scheduled for installation at the University of Western Australia in January, 1965

? Stanford AI Lab

The first computer installed at the Stanford Artificial Intelligence Lab was a DEC PDP-6, delivered in June of 1966
http://forum.stanford.edu/wiki/index.php/Early_Computers_at_Stanford
http://www.stanford.edu/~learnest/sailaway.htm

"a 6-console display system that drew text and vectors with a random-access electron beam was added in 1967"
http://www.stanford.edu/~learnest/sailaway.htm

"The original PDP-6 system had just 64k words of storage (which occupied eight large cabinets) and used microtapes for secondary storage. A fixed-head disc file built by Librascope, added in 1968, was supposed to function both as a swapping store and a permanent file store, but it turned out to be so temperature-sensitive that it was useless for file storage. The six remarkably large discs in this system, which were each 4 feet in diameter, were eventually sold as coffee tables -- I have one in my living room. Despite its large physical size, this disc system had a capacity of only about 100 megabytes. More reliable disks made by IBM, Ampex and DEC were added in later years."
http://www.stanford.edu/~learnest/sailaway.htm

"Early Digital Equipment Computing"
Photo of memory plane "moved to basement"
http://www-db.stanford.edu/pub/voy/museum/pictures/display/3-3.htm

Retired 1980.
Stanford's PDP-6 was shown at DECUS in 1984. The machine was transferred to a DEC warehouse after that event. There are no records of this machine being given to the Computer Museum, which was not part of DEC in 1984. In the late 1990s Compaq donated the contents of the DEC internal archives to The Computer Museum History Center. The Fast Memory cabinet from the Stanford PDP-6 was part of that donation. There is no evidence that the modules sold at the Boston computer museum gift shop were from the Stanford PDP-6, nor is there any evidence that the museum had ever had this machine in its possession. [wikipedia]

On display at 1984 DECUS: http://www.opost.com/dlm/tenex/
TCM has box of spare parts

http://www.computerhistory.org/collections/accession/X2244.2002B

There are 12 modules in original DEC cardboard box. The object is from Stanford's PDP-6 (from Bruce Kennard, Stanford DEC CE). The object category needs to verified by content specialist. Exhibit label for a PDP-6 Module (including contextual paragraph) is located in Object File. -- JAC 10/25/2001

? MIT Laboratory for Nuclear Science (LNS)
Invoice date 2/17/65

? United Aircraft (United Technologies)

? University of Pennsylvania
truck ran into bridge in Maynard, Ma [hurley]

? University of California Berkeley
Invoice date 10/20/65
used by Dr. Glaser, inventor of the bubble chamber
(located in basement of Cory Hall?)

? Key Data (timesharing)

? Rand Corporation (timesharing)
Invoice date 7/2/65

? Applied Logic Corporation (timesharing)
Princeton, NJ

? University of Bonn
Invoice date 6/29/65

? University of Aachen
Invoice date 6/30/65

? University of Heidelberg

? Rutgers Physics
Invoice date 3/26/65
http://www.inwap.com/pdp10/usenet/pdp6

? Lawrence Livermore Labs (dual procssor)
? Lawrence Livermore Labs (dual procssor)
Invoice date 2/11/65 x2
"Octopus" network controller; built own segmentation hardware
http://web.archive.org/web/20040810204024/http://www.nersc.gov/~deboni/Computer.history/pdp_6.html
http://www.nersc.gov/~deboni/Computer.history/pdp_6.html
http://web.archive.org/web/20040619224937/http://www.nersc.gov/~deboni/Computer.history/2.pdp.6.consoles.jpg

http://web.archive.org/web/20040621041934/http://www.nersc.gov/~deboni/Computer.history/Lafranchi.html
EL: ... Pehrson arrives in '65 and was very instrumental in a PDP-6. I think it was being prepared as the main OCTOPUS controller. Putting in a memory pagination and segmentation system.

GAM: You guys were designing the Pag-Seg (Pagination & Segmentation) unit and that was supposed to be added to the PDP-6. Wyman said no one ever used it.

replaced by KA10's

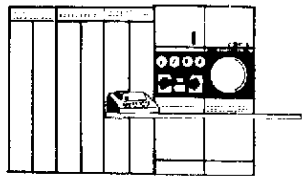
http://web.archive.org/web/20030402030119/http://www.ornl.gov/ORNL/SC/norman_hardy_1.html

> NH: They and Mel Pirtle were designing the SDS 940, yes. It was what they called the 930 from XDS, which was the same company that built the Sigma 7. The 930 was a machine much inferior to the PDP-10, but it was cheaper. And Berkeley got one, and sort of did magic on it--they did their own paging stuff. And that machine became the prototype for the 940.

http://research.microsoft.com/users/GBell/Computer_Engineering/00000532.htm

> Although we predicated the original PDP-6 hardware on multiprocessing, the Monitor was not designed explicitly for it. Lawrence Livermore Laboratory did build a two-processor system with their own operating system and special segmentation hardware

PDP–6
PRICE LIST
FEBRUARY 1, 1964



ARITHMETIC PROCESSOR TYPE 166

Fast Memory 0.5 Microsecond - 16 Words
16 Accumulators
15 Index Registers
Floating Point Arithmetic
I-O Teleprinter
7 Channel Priority Interrupt I-O System
$176,100

CORE MEMORY TYPE 163B

8,192 36-bit words $ 80,000

CORE MEMORY TYPE 163C

16,384 36-bit words $126,000

Stores either 8,192 or 16,384 36-bit words with a memory cycle time of 2.0 microseconds. Includes one memory bus interface. For each additional processor, an additional interface is required. Each interface services one memory-processor.

Each additional interface (maximum of 3) $ 2,700

MAGNETIC DRUM AND PROCESSOR TYPE 236

Controls from one to four drum units. Each drum unit consists of 768 tracks of 49,152 bits each for a total storage of 37.8 x 10^6 bits. The drum rotates at 1150 rpm (52 milliseconds per revolution), and provides a 36-bit word transfer every 6.4 microseconds. Total drum storage is 1,048,576 36-bit words. $ 95,000

PAPER TAPE AND CARD EQUIPMENT

HIGH SPEED PAPER TAPE READER AND CONTROL TYPE 760

Reads perforated paper tape photoelectrically at 400 characters per second. $ 9,000

HIGH SPEED PAPER TAPE PUNCH AND CONTROL TYPE 761

Punches 8-hole paper tape at 63.3 characters per second. $ 5,500

CARD READER AND CONTROL TYPE 461

Provides on-line reading of 80-column punched cards at 200 or 800 cards per minute in either alphanumeric or binary codes.

200 cards per minute $ 16,500
800 cards per minute $ 27,200

CARD PUNCH AND CONTROL TYPE 460

Permits on-line punching of 80-column cards at 100 or 300 cards per minute.

100 cards per minute $ 29,000
300 cards per minute $ 42,000

PRINTERS

AUTOMATIC LINE PRINTER AND CONTROL TYPE 646

Prints 300 lines per minute, 120 columns per line, 64 characters per column. $ 30,000

HIGH SPEED AUTOMATIC LINE PRINTER AND CONTROL TYPE 680

Prints 1000 lines per minute, 120 columns per line, 64 characters per column. $ 47,500

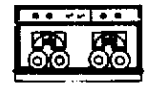
MAGNETIC TAPE EQUIPMENT

MAGNETIC TAPE TRANSPORT TYPE 570

Reads and writes IBM compatible tape at a recording density of 200 or 556 characters per inch. Tape speed is 75 or 112.5 inches per second with transfer rates from 15 to 62.5 KC. $ 30,400

MAGNETIC TAPE CONTROL TYPE 516

Automatically controls up to eight Type 570 magnetic tape transports. Permits reading, writing, forward/backward spacing, and rewind. Requires Type 136 Data Control. $ 15,000



DUAL MICROTAPE SYSTEM

Provides a fixed address magnetic tape facility for high speed loading, readout and program updating. Density is 375 ± 60 bits per track inch at a speed of 80 inches per second. Transfers 90,000 bits per second.

Type 555 Dual Transport (includes two independent tape drives) $ 7,400

Type 551 Control Unit (controls up to eight Type 555 Tape Transports. Requires Type 136 Data Control $ 14,000

OTHER EQUIPMENT

DATA CONTROL TYPE 136

Assembles and disassembles 36-bit words. May be used with up to four tape controls and two special purpose data handling devices. $ 10,000

DISPLAY MONITOR AND CONTROL TYPE 346

Plots points, lines, vectors, and characters on a 9⅜-inch square raster of 1024 points along each axis.

1½ microseconds is required per point in vector, increment and character modes; random point plotting rate of 35 microseconds. Uses fiber optic light pipe and photomultiplier system for fast detection of displayed information. $ 40,000

MULTI-USER STATION TYPE 627

Standard ASCII 8-level Teletypewriter, paper tape reader-punch, operating at 10 cps. $8,000

On Line

Prepares programs directly onto magnetic tape
Permits program execution and debugging
Permits tape editing
Provides hard copy printouts

Off Line

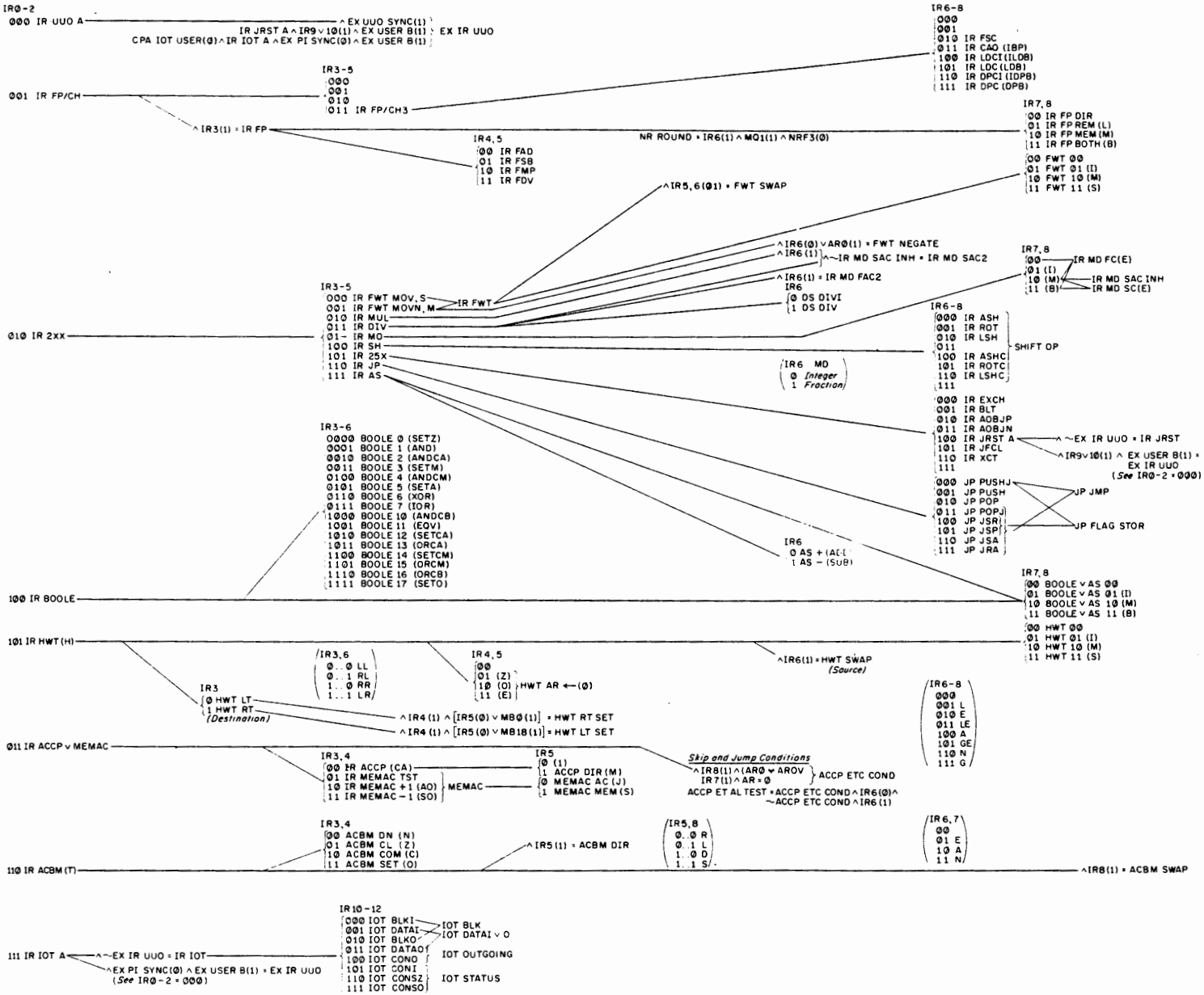
Punches paper tape

Prices quoted are effective February 1, 1964. FOB Maynard, Mass., and apply in continental United States only. Federal, state or local taxes are not included. Option prices are for factory installation; field installation prices will be quoted on request. Quantity prices are quoted on request. All prices are subject to change without notice.

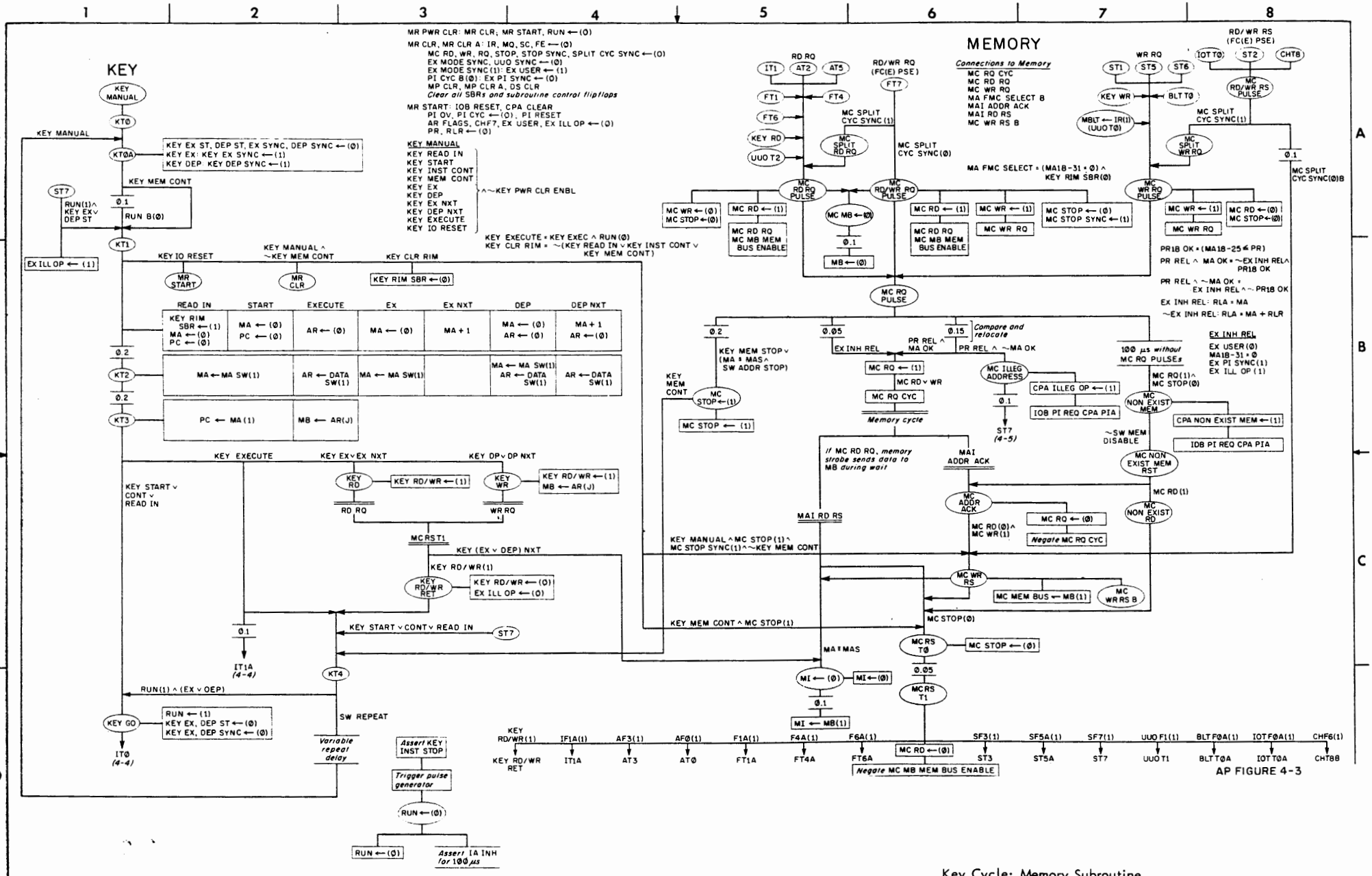
PROGRAMMED DATA PROCESSOR-6

System Block Diagram
(SD-D-166-0-BD)

LOGIC PREFIXES

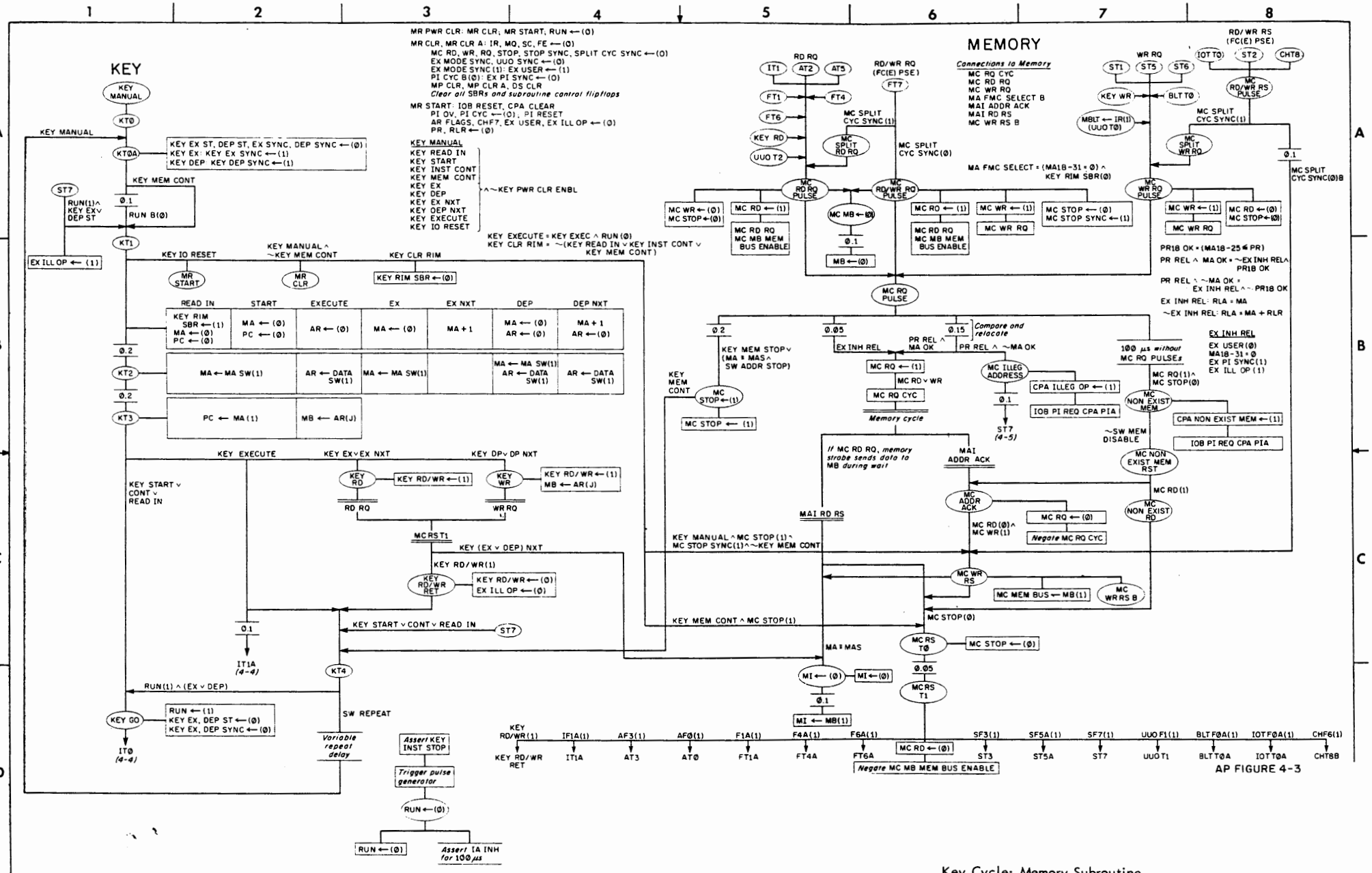


Instruction Decoding
(FD-D-166-0-IRD)

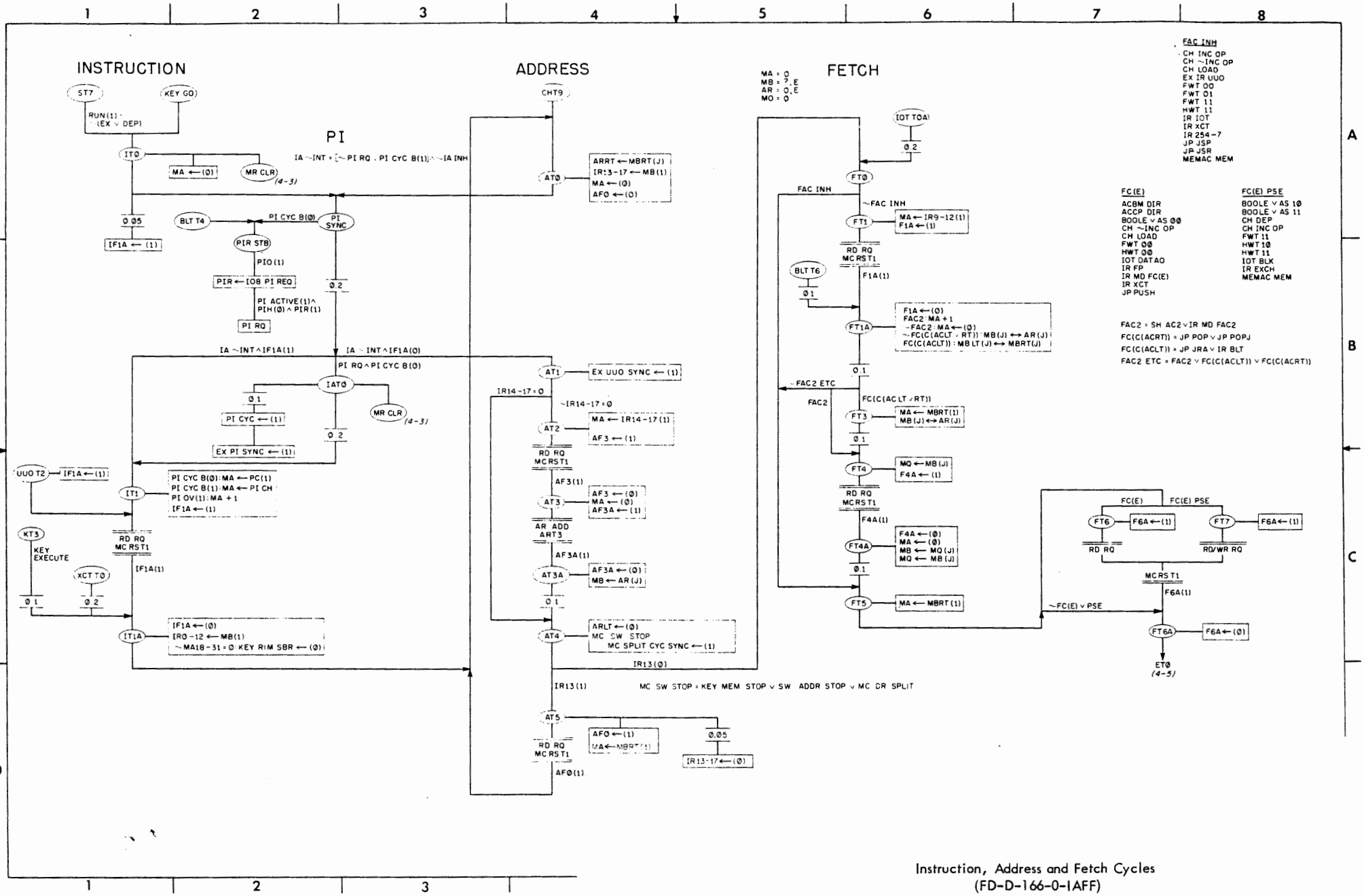


AP FIGURE 4-3

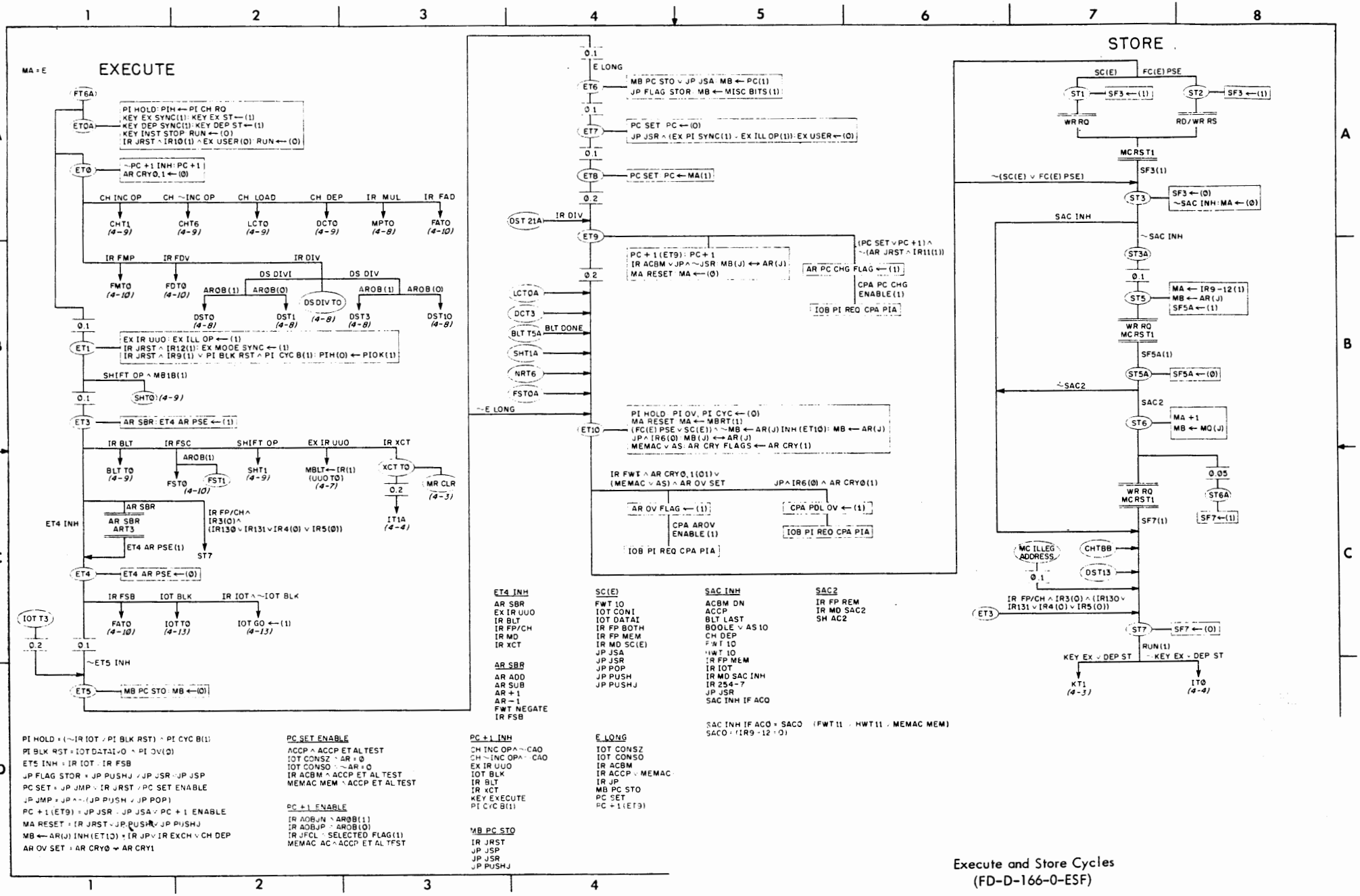
Key Cycle; Memory Subroutine
(FD-D-166-0-KMF)



Key Cycle; Memory Subroutine
(FD-D-166-0-KMF)



Instruction, Address and Fetch Cycles
(FD-D-166-0-1AFF)



Execute and Store Cycles
(FD-D-166-0-ESF)

HALF WORD TRANSFER
500-577

IR = 101 WXX YZZ
IR HWT = IRØ-2(101)
W specifies destination half
HWT LT = IR HWT : IR3(0)
HWT RT = IR HWT : IR3(1)
XX specifies action on other half – do nothing, zero, one, extend
HWT AR ← (0) : IR HWT : IR4(1) ∨ IR5(1)
HWT RT SET = HWT LT ∧ IR4(1) ∨ IR5(0) ∨ MBØ(1)
HWT LT SET = HWT RT : IR4(1) ∨ IR5(0) ∨ MB18(1)
Y specifies source half
HWT SWAP = IR HWT ∧ IR6(1)
ZZ specifies mode

IR HWT ∧ IR7,8	Action
00 = HWT 00	∕[C(E), AC] → AC
01 = HWT 01	∕[(0,E), AC] → AC
10 = HWT 10	∕[AC, C(E)] → E
11 = HWT 11	∕[C(E)] → E

Initial Registers	HWT 00: AR = AC MB = C(E) HWT 01: AR = AC MB = Ø,E HWT 10: AR = AC MB = C(E) HWT 11: AR = Ø,E MB = C(E) MQ = Ø
Initial Gates	HWT 00: FC(E) HWT 10: FC(E) PSE HWT 11: FAC INH, FC(E) PSE
ETØA	HWT 10: MB(J) ↔ AR(J) HWT 11: AR ← MB(J)
ET1	HWT SWAP: MB LT(J) ↔ MB RT(J) HWT AR ← (Ø): AR ← (Ø)
ET3	
ET4	HWT LT SET: AR LT COM HWT RT SET: AR RT COM HWT LT: AR ← MB LT(J) HWT RT: AR ← MB RT(J)
ET5	
ET10	HWT 10 - 11: MB ← AR(J)
Final Gates	HWT 10: FC(E) PSE, SAC INH HWT 11: FC(E) PSE HWT 11: SACØ: SAC INH SACØ = (IR9-12 = Ø)

FULL WORD TRANSFER
200-217

IR = 010 00Y YZZ
IR 2XX = IRØ-2(010)
YY specifies instruction – MOVE, MOVS, MOVN, MOVM
IR MOV, S = IR 2XX ∧ IR3-5(000)
IR MOVN, M = IR 2XX ∧ IR3-5(001)
IR FWT = IR MOV, S : IR MOVN, M
FWT SWAP = IR FWT : IR5,6(01)
FWT NEGATE = IR MOVN, M ∧ IR6(Ø) ∨ ARØ(1)
ZZ specifies mode

IR FWT ∧ IR7,8	Action
00 = FWT 00	∕[C(E)] → AC
01 = FWT 01	∕[Ø,E] → AC
10 = FWT 10	∕[AC] → E
11 = FWT 11	∕[C(E)] → E

Initial Registers	FWT 00: AR = Ø,E MB = C(E) FWT 01: AR = Ø,E MB = ?,E FWT 10: AR = AC MB = Ø,E FWT 11: AR = Ø,E MB = C(E) MQ = Ø
Initial Gates	FWT 00: FAC INH, FC(E) FWT 01: FAC INH FWT 11: FAC INH, FC(E) PSE FWT NEGATE: ET4 INH, AR SBR
ETØA	FWT 00 - 11: AR ← MB(J) FWT 01 - 10: MB ← AR(J)
ET1	FWT SWAP: MBLT(J) ← MBRT(J)
ET3	FWT NEGATE: ET4 AR PSE ← (1) AR NEGATE: ART3
ET4	ET4 AR PSE ← (Ø) FWT SWAP: MB(J) ← AR(J)
ET5	
ET10	AR CRYØ,1(01) AR OV FLAG ← (1) FWT 10 ∨ 11: MB ← AR(J)
Final Gates	FWT 10: SC(E), SAC INH FWT 11: FC(E) PSE FWT 11 ∧ SACØ: SAC INH SACØ = (IR9-12 = Ø)

EXCH
250

IR = 010 101 000
IR 2XX = IRØ-2(010)
IR 25X = IR 2XX ∧ IR3-5(101)
IR EXCH = IR 25X ∧ IR6-8(000)

AC ⟷ C(E)

Initial Registers	AR = AC MB = C(E) MQ = Ø
Initial Gates	FC(E) PSE MB ← AR(J) INH(ET10)
ETØA	MB(J) ← AR(J)
ET1	
ET3	
ET4	
ET5	
ET10	
Final Gates	FC(E) PSE

ARITHMETIC COMPARE
300-377

IR = 011 VVW XYZ
IR ACCP ∨ MEMAC = IRØ-2(011)
VV specifies instruction type

(IR ACCP ∨ MEMAC) ∧ IR3,4	
ØØ	= ACCP
Ø1	= MEMAC TST
1Ø	= MEMAC +1
11	= MEMAC −1

MEMAC = MEMAC TST ∨ MEMAC +1 ∨ MEMAC −1
X specifies whether or not action is on condition YZ
Y specifies condition 'equals'
Z specifies condition 'less than'
XYZ together define relation R

XYZ	R
ØØØ	*Never*
ØØ1	<
Ø1Ø	=
Ø11	≤
1ØØ	*Always*
1Ø1	≥
11Ø	≠
111	>

LOGICAL COMPARE
600-677

IR = 110 VVW XYZ
IR ACBM = IRØ-2(110)
VV specifies action on masked bits

IR ACBM ∧ IR3,4	
ØØ	= ACBM DN
Ø1	= ACBM CL
1Ø	= ACBM COM
11	= ACBM SET

W specifies source of mask – immediate or memory
ACBM DIR = IR ACBM ∧ IR5(1)
X specifies whether or not action is on condition Y, i.e 'equals zero'
XY together define relation R

XY	R
ØØ	*Never*
Ø1	=Ø
1Ø	*Always*
11	≠Ø

Z specifies swapping mask selected by W
ACBM SWAP = IR ACBM ∧ IR8(1)
WZ together specify mask as (Ø,E), (E,Ø), C(E), C(E) RT LT

AOBJP - AOBJN
252 - 253

IR = Ø10 101 01X
IR 2XX = IRØ-2(Ø10)
IR 25X = IR 2XX ∧ IR3-5(1Ø1)
X specifies instruction
IR AOBJP = IR 25X ∧ IR6-8(Ø10)
IR AOBJN = IR 25X ∧ IR6-8(Ø11)

AC + 1000001 → AC
AOBJP ∧ (AC ≥ Ø): E → PC
AOBJN ∧ (AC < Ø): E → PC

IR AOBJP ∧ ARØB(Ø): PC SET
IR AOBJN ∧ ARØB(1): PC SET

Determination of Skip or Jump Condition
ACCP ETC COND = [IR7(1) ∧ (AR = Ø)] ∨ [(IR ACCP ∨ MEMAC) ∧ IR8(1) ∧ (ARØ ⟷ AR OV)]
ACCP ET AL TEST = ACCP ETC COND ⟷ IR6
ARØ ⟷ AR OV = ARØ ⟷ (AR OV SET ∧ ~MEMAC)
AR OV SET = AR CRYØ ⟷ AR CRY1

MEMAC
320-377

W specifies whether AC or C(E) compared with zero, i.e whether action is jump or skip
MEMAC AC = MEMAC ∧ IR5(Ø)
MEMAC MEM = MEMAC ∧ IR5(1)

MEMAC AC	MEMAC MEM
AC R Ø: E → PC	C(E) R Ø: PC +1
MEMAC ±1: AC ±1 → AC	MEMAC ±1: C(E) ±1 → E
ACCP ET AL TEST: PC SET	~ACØ ∧ MEMAC ±1: C(E) ±1 → AC
	SACØ = (IR9-12 = Ø)

Initial Registers	AR = AC MB = Ø,E MQ = Ø
Initial Gates	MEMAC ±1: ET4 INH AR SBR E LONG
ETØA	
ET1	
ET3	MEMAC +1: ET4 AR PSE ← (1) ∣AR +1 ART3∣ MEMAC −1: ET4 AR PSE ← (1) ∣AR −1 ART3∣
ET4	ET4 AR PSE ← (Ø)
ET5	
ET6	
ET7	PC SET: PC ← (Ø)
ET8	PC SET: PC ← MA(1)
ET9	PC SET: AR PC CHG FLAG ← (1)
ET10	AR OV SET: AR OV FLAG ← (1) AR CRY FLAGS ← AR CRY(1)
Final Gates	

ACCP
300-317

W specifies whether AC compared with E or C(E)
ACCP DIR = ACCP ∧ IR5(1)
IR5(Ø) ∧ [AC R (Ø,E)] : PC +1
IR5(1) ∧ [AC R C(E)] : PC +1

Initial Registers	AR = AC ~ACCP DIR: MB = Ø,E ACCP DIR: MB = C(E) MQ = Ø
Initial Gates	ACCP DIR: FC(E) ET4 INH AR SBR E LONG
ETØA	
ET1	
ET3	ET4 AR PSE ← (1) AR SUB ART3
ET4	ET4 AR PSE ← (Ø)
ET5	
ET6	
ET7	
ET8	
ET9	ACCP ET AL TEST: PC +1 AR PC CHG FLAG ← (1)
ET10	AR OV SET: AR OV FLAG ← (1) AR CRY FLAGS ← AR CRY(1) MB ← AR(J)
Final Gates	SAC INH

ACBM

R [*Mask* ∧ AC] : PC + 1
∕ [*Mask*, AC] → AC

Initial Registers	AR = AC ~ACBM DIR: MB = Ø,E ACBM DIR: MB = C(E) MQ = Ø
Initial Gates	ACBM DIR: FC(E) E LONG
ETØA	ACBM SWAP: MB LT(J) ↔ MB RT(J)
ET1	MB ← AR(Ø) ACBM COM: AR ← MB(⟷) ACBM SET: AR ← MB(1)
ET3	
ET4	MB(J) ↔ AR(J)
ET5	AR COM
ET6	ACBM CL: MB ← AR(Ø)
ET7	AR COM
ET8	
ET9	MB(J) ← AR(J) ACCP ET AL TEST: PC +1 AR PC CHG FLAG ← (1)
ET10	
Final Gates	ACBM DN: SAC INH

Initial Registers	AR = AC MB = Ø,E MQ = Ø
Initial Gates	ET4 INH AR SBR E LONG
ETØA	
ET1	
ET3	ET4 AR PSE ← (1) AR + 1 LT RT ART3
ET4	ET4 AR PSE ← (Ø)
ET5	
ET6	
ET7	PC SET: PC ← (Ø)
ET8	PC SET: PC ← MA(1)
ET9	PC SET: AR PC CHG FLAG ← (1)
ET10	
Final Gates	

Data Transmission and Compare Instructions
(FD-D-166-0-DTCF)

JUMP AND PUSHDOWN

260-267

IR = 010 110 xxx
IR 2xx = IR0-2 (010)
IR JP = IR 2xx ∧ IR3-5 (110)

xxx *specifies instruction*

IR JP ∧ IR6-8 (
000 = JP PUSHJ
001 = JP PUSH
010 = JP POP
011 = JP POPJ
100 = JP JSR
101 = JP JSP
110 = JP JSA
111 = JP JRA

JP JMP = IR JP ∧ ~(JP PUSH ∨ JP POP)
JP FLAG STOR = JP PUSHJ ∨ JP JSR ∨ JP JSP
~IR IOT ∧ PI CYC B(1): PI HOLD

	PUSHJ 260	PUSH 261	POP 262	POPJ 263	JSR 264	JSP 265	JSA 266	JRA 267
	AC+1000001 → AC MISC BITS, PC → C(C(ACRT)) E → PC PI ON OVERFLOW	AC+1000001 → AC C(E) → C(C(ACRT)) PI ON OVERFLOW	C(C(ACRT)) → E AC-1000001 → AC PI ON UNDERFLOW	C(C(ACRT)) → PC AC-1000001 → AC PI ON UNDERFLOW	MISC BITS, PC → E E + 1 → PC	MISC BITS, PC → AC E → PC	AC → E E, PC → AC E + 1 → PC	C(C(ACLT)) → AC E → PC
Initial Registers	AR = AC MB = 0, E MQ = 0	AR = AC MB = C(E) MQ = 0	AR = AC MB = 0, E MQ = C(C(ACRT))	AR = AC MB = 0, E MQ = C(C(ACRT))	AR = 0, E MB = ?, E MQ = 0	AR = 0, E MB = ?, E MQ = 0	AR = AC MB = 0, E MQ = 0	AR = AC RTLT MB = 0, E MQ = C(C(ACLT))
Initial Gates	ET4 INH AR SBR E LONG MB ← AR(J) INH (ET10)	FC(E) ET4 INH AR SBR E LONG MB ← AR(J) INH (ET10)	FC(C(ACRT)) ET4 INH AR SBR E LONG MB ← AR(J) INH (ET10)	FC(C(ACRT)) ET4 INH AR SBR E LONG MB ← AR(J) INH (ET10)	FAC INH E LONG MB ← AR(J) INH (ET10)	FAC INH E LONG MB ← AR(J) INH (ET10)	E LONG MB ← AR(J) INH (ET10)	FC(C(ACLT)) E LONG MB ← AR(J) INH (ET10)
ET0A			MB ← MQ(J)	MB ← MQ(J)	PI HOLD: PIH ← PI CH RQ	MB(J) ↔ AR(J)	MBLT(J) ↔ MBRT(J)	MB ← MQ(J)
ET1				MA ← (0)				
ET3	ET4 AR PSE ← (1) AR + 1 LTRT ART3	ET4 AR PSE ← (1) AR + 1 LTRT ART3	ET4 AR PSE ← (1) AR - 1 LTRT ART3	ET4 AR PSE ← (1) AR - 1 LTRT ART3				
ET4	ET4 AR PSE ← (0)	ET4 AR PSE ← (0)	ET4 AR PSE ← (0)	ET4 AR PSE ← (0)				
ET5	MB ← (0)				MB ← (0)	MB ← (0)		
ET6	MB ← MISC BITS, PC(1)				MB ← MISC BITS, PC(1)	MB ← MISC BITS, PC(1)	MB ← PC(1)	
ET7	PC ← (0)			PC ← (0)	PC ← (0) EX PI SYNC(1) ∨ EX ILL OP(1): EX USER ← (0)	PC ← (0)	PC ← (0)	PC ← (0)
ET8	PC ← MA(1)			PC ← MA(1)	PC ← MA(1) EX ILL OP ← (0)	PC ← MA(1)	PC ← MA(1)	PC ← MA(1)
ET9	MB(J) ← AR(J) MA ← (0)	MB(J) ↔ AR(J) MA ← (0)	MB(J) ↔ AR(J)	MB(J) ↔ AR(J)	PC + 1 CHF7 ← (0)	MB(J) ← AR(J)	MB(J) ↔ AR(J) PC + 1	MB(J) ↔ AR(J)
ET10	MA ← MBRT(1) MB(J) ↔ AR(J) AR CRY0(1): CPA PDL OV ← (1) IOB PI REQ CPA PIA		MB(J) ↔ AR(J) AR CRY0(1): CPA PDL OV ← (1) IOB PI REQ CPA PIA		PI HOLD: PI OV, PI CYC ← (0)			
Final Gates	SC(E)	SC(E)	SC(E)		SC(E) SAC INH		SC(E)	

JRST
254

IR = 010 101 100
IR 2xx = IR0-2 (010)
IR 25X = IR2XX ∧ IR3-5 (101)
IR JRST A = IR 25X ∧ IR 6-8 (100)
IR JRST = IR JRST A ∧ ~EX IR UUO

E → PC
IR9(1): DISMISS PI
IR10(1): HALT
IR11(1): RESTORE MISC BITS
IR12(1): ENTER USER MODE

Initial Registers	AR = 0, E MB = ?, E MQ = 0
Initial Gates	FAC INH E LONG
ET0A	IR10(1): RUN ← (0) IR11(1): AR FLAG CLEAR (*Clears* MISC BITS *or* ET0)
ET1	IR9(1): PIH(0) ← PIOK(1) IR11(1): AR FLAG SET (*Transfers* MB(1) *into* MISC BITS) IR12(1): EX MODE SYNC ← (1)
ET3	
ET4	
ET5	MB ← (0)
ET6	MB ← PC(1)
ET7	PC ← (0)
ET8	PC ← MA(1)
ET9	MA ← (0)
ET10	MA ← MBRT(1)
Final Gates	SAC INH

JFCL
255

IR = 010 101 101
IR 2xx = IR0-2 (010)
IR 25X = IR 2xx ∧ IR3-5 (101)
IR JFCL = IR 25X ∧ IR 6-8 (010)

SELECTED FLAG(1): E → PC
CLEAR SELECTED FLAGS

Initial Registers	AR = 0, E MB = ?, E MQ = 0
Initial Gates	FAC INH E LONG
ET0A	
ET1	
ET3	
ET4	
ET5	
ET6	
ET7	SELECTED FLAG(1): PC ← (0)
ET8	SELECTED FLAG(1): PC ← MA(1)
ET9	
ET10	SELECTED FLAGS ← (0)
Final Gates	SAC INH

XCT
256

IR = 010 101 110
IR 2xx = IR0-2 (010)
IR 25X = IR 2xx ∧ IR3-5 (101)
IR XCT = IR 25X ∧ IR6-8 (110)

EXECUTE C(E)

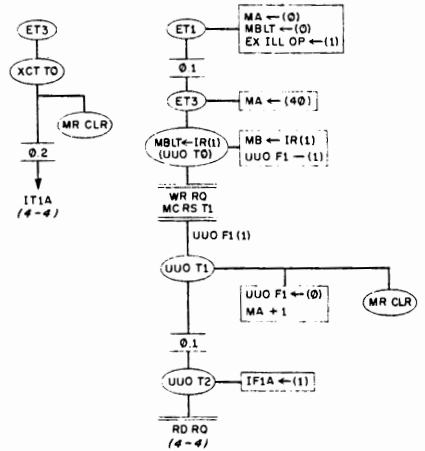
Initial Registers	AR = 0, E MB = C(E) MQ = 0
Initial Gates	FAC INH FC(E) PC + 1 INH ET4 INH

UUO
000-077

IR = 000 xxx xxx
IR UUO A = IR0-2 (000)
EX IR UUO = IR UUO A ∧ EX UUO SYNC (1) ∨ IR JRST A ∧ EX USER B(1) ∧ IR9 ∨ 10(1) ∨ IR IOT A ∧ EX USER B(1) ∧ EX PI SYNC(1) ∧ CPA IOT USEP (0)

IR, E → 40
EXECUTE 41

Initial Registers	AR = 0, E MB = ?, E MQ = 0
Initial Gates	FAC INH PC + 1 INH ET4 INH



MISC BITS
0 AR OV FLAG
1 AR CRY0 FLAG
2 AR CRY1 FLAG
3 AR PC CHG FLAG
4 CHF7
5 EX USER - STORE ONLY
EX MODE SYNC - SET ONLY

SELECTED FLAGS
IR9(1) AR OV FLAG
IR10(1) AR CRY0 FLAG
IR11(1) AR CRY1 FLAG
IR12(1) AR PC CHG FLAG

Pushdown and Jump Instructions
(FD-D-166-0-JPF)

BOOLEAN

400-477

IR = 100 XXX XYY
IR BOOLE = IR0-2(100)
XXXX *specifies Boolean function, decoded into* BOOLE 0-17 *as* BOOLE N = IR BOOLE ∧ IR3-6(N)

ADD-SUBTRACT

270-277

IR = 010 111 XYY
IR 2XX = IR0-2(010)
IR AS = IR 2XX ∧ IR3-5 (111)
X *specifies add or subtract*

IR AS ∧ IR6 → 0 = AS+, 1 = AS−

In both instruction groups YY specifies common mode

(IR BOOLE ∨ IR AS) ∧ IR7,8 → 00 = BOOLE AS 00; 01 = BOOLE AS 01; 10 = BOOLE AS 10; 11 = BOOLE AS 11

Action

BOOLE	AS
BOOLE ∨ AS 00: ƒ(AC, C(E)) → AC	AC ±C(E) → AC
BOOLE ∨ AS 01: ƒ(AC, (0,E)) → AC	AC ±(0,E) → AC
BOOLE ∧ AS 10: ƒ(AC, C(E)) → E	AC ±C(E) → E
BOOLE ∧ AS 11: ƒ(AC, C(E)) → AC, E	AC ±C(E) → AC, E

Initial Registers

AR = AC
BOOLE ∨ AS 00 ∨ 10 ∨ 11: MB = C(E)
BOOLE ∧ AS 01: MB = (0,E)
MQ = 0

Common Initial Gates

BOOLE ∧ AS 00: FC(E)
BOOLE ∧ AS 10 ∨ 11: FC(E) PSE

Initial Gates AS Only

		ET4 INH AR SBR
ET0A	BOOLE 0, 3, 14, 17: AR ← (0) BOOLE 2, 4, 12, 13, 15: AR COM	
ET1	BOOLE 6, 11, 14: AR ← MB(∨) BOOLE 1, 2, 15, 16: AR ← MB(0) BOOLE 3, 4, 7, 10, 13: AR ← MB(1)	
ET3		ET4 AR PSE ← (1) AS+: AR ADD AS−: AR SUB ART3
ET4	BOOLE 4, 10, 11, 14, 15, 16, 17: AR COM	ET4 AR PSE ← (0)
ET5		
ET10	FC(E) PSE: MB ← AR(J)	FC(E) PSE: MB ← AR(J) AR OV SET: AR OV FLAG ← (1) AR CRY FLAGS ← AR CRY(1) AR OV SET = AR CRY0 ∨ AR CRY1

Final Gates

BOOLE ∧ AS 10: FC(E) PSE, SAC INH
BOOLE ∧ AS 11: FC(E) PSE

BOOLEAN FUNCTIONS

	0 SETZ	1 AND	2 ANDCA	3 SETM
ETOA	AR ← (0)		AR COM	AR ← (0)
ET1		AR ← MB(0)	AR ← MB(0)	AR ← MB(1)
ET4				

	4 ANDCM	5 SETA	6 XOR	7 IOR
ETOA	AR COM			
ET1	AR ← MB(1)		AR ← MB(∨)	AR ← MB(1)
ET4	AR COM			

	10 ANDCB	11 EQV	12 SETCA	13 ORCA
ETOA			AR COM	AR COM
ET1	AR ← MB(1)	AR ← MB(∨)		AR ← MB(1)
ET4	AR COM	AR COM		

	14 SETCM	15 ORCM	16 ORCB	17 SETO
ETOA	AR ← (0)	AR COM		AR ← (0)
ET1	AR ← MB(∨)	AR ← MB(0)	AR ← MB(0)	
ET4	AR COM	AR COM	AR COM	AR COM

MULTIPLY-DIVIDE

220-237

IR = 010 01X YZZ
IR 2XX = IR0-2(010)
X *specifies operation*

IR 2XX ∧ IR3-5 → 010 = IR MUL; 011 = IR DIV

Y *specifies operand type:* IR6 *is sign in multiply flow; for divide*
DS DIVI = IR DIV ∧ IR6(0)
DS DIV = IR DIV ∧ IR6(1)

ZZ *specifies mode*

Levels for fetch and store according to operand type and mode

IR MD = IR 2XX ∧ IR3(0) ∧ IR4(1)
IR MD FC(E) = IR MD ∧ (IR7(0) ∧ IRB(1))
IR MD FAC2 = IR DIV · IR6(1)
IR MD SC(E) = IR MD · IR7(1)
IR MD SAC INH = IR MD · IR7(1) · IRB(0)
IR MD SAC2 = ¬(IR MD SAC INH) · (IR DIV ∨ IR MUL ∨ IR6(1))

IR MUL

Action

For IMUL (XY = 00) *store only low order half of product*
ZZ = 00: AC ← C(E) → AC
ZZ = 01: AC ← (0,E) → AC
ZZ = 10: AC ∧ C(E) → E
ZZ = 11: AC ∧ C(E) → AC; E

For MUL (XY = 01) *store entire product or high order half*
ZZ = 00: AC ← C(E) → AC, AC2
ZZ = 01: AC ∧ (0,E) → AC, AC2
ZZ = 10: AC ∧ C(E) → E
ZZ = 11: AC ∧ C(E) → AC, AC2; E

Initial Registers

AR = AC
ZZ = 00 ∨ 10 ∨ 11: MB = C(E)
ZZ = 01: MB = 0, E
MQ = 0

Initial Gates

ZZ = 00 ∨ 10 ∨ 11: FC(E)
ET4 INH

IDIV (XY = 10)

Action

ZZ = 00: AC ← C(E) → AC, AC2
ZZ = 01: AC ← (0,E) → AC, AC2
ZZ = 10: AC ← C(E) → E
ZZ = 11: AC ← C(E) → AC, AC2; E

Initial Registers

AR = AC
ZZ = 00 ∨ 10 ∨ 11: MB = C(E)
MQ = 0

Initial Gates

ZZ = 00 ∨ 10 ∨ 11: FC(E)
ET4 INH

DIV (XY = 11)

Action

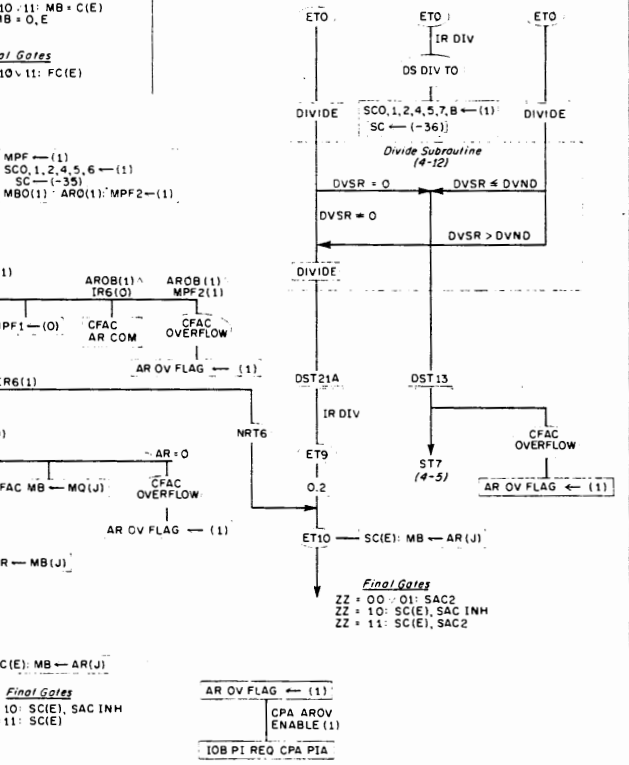
ZZ = 00: AC, AC2 ÷ C(E) → AC, AC2
ZZ = 01: AC, AC2 ÷ (0,E) → AC, AC2
ZZ = 10: AC, AC2 ÷ C(E) → E
ZZ = 11: AC, AC2 ÷ C(E) → AC, AC2; E

Initial Registers

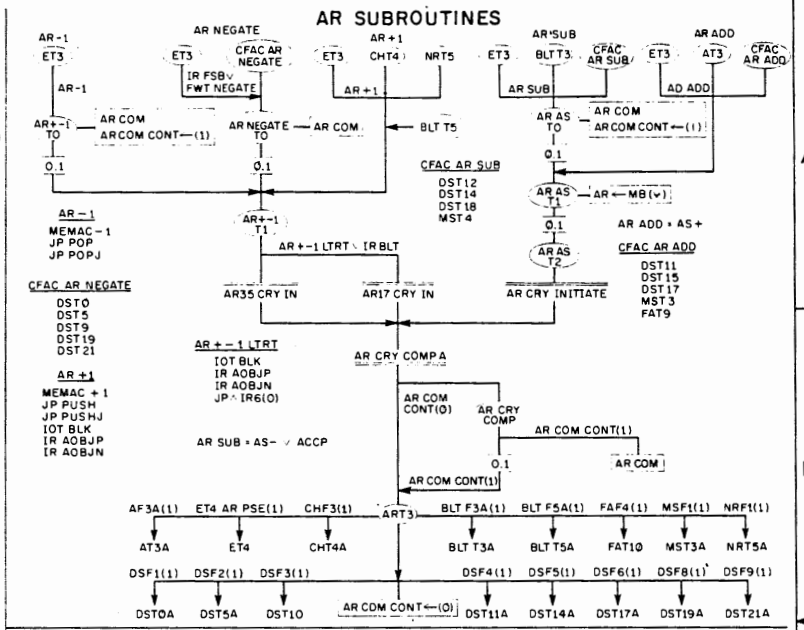
AR = AC
ZZ = 00 ∨ 10 ∨ 11: MB = C(E)
ZZ = 10: MB = 0, E
MQ = AC2

Initial Gates

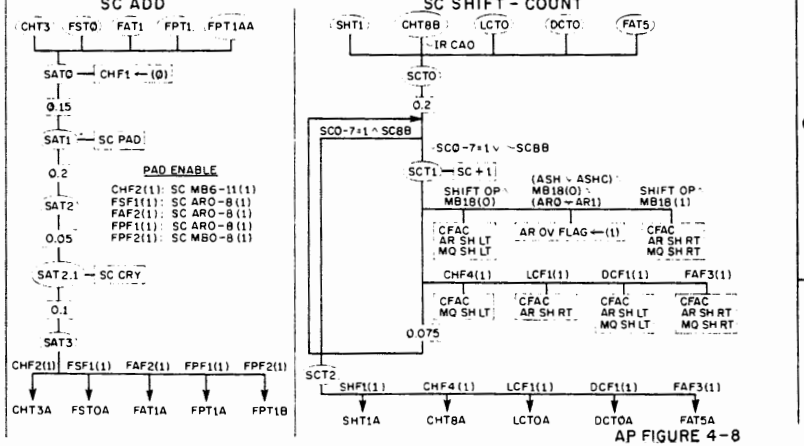
ZZ = 00 ∨ 10 ∨ 11: FC(E)
FAC2
ET4 INH



AR SUBROUTINES



SHIFT COUNTER SUBROUTINES



AP FIGURE 4-8

Fixed Point and Logical Instructions; AR and SC Subroutines
(FD-D-166-0-ALF)

CHARACTER OPERATIONS
133-137

FIRST PART-CHF5(Ø)

CH INC = (IR LDCI ∨ IR DPCI ∨ IR CAO) ∧ CHF5(Ø)
CH INC OP = CH INC ∧ CHF7(Ø)
CH ~INC OP = ~(IR LDC ∨ IR OPC) ∧ CHF5(Ø) ∨ CH INC ∧ CHF7(1)

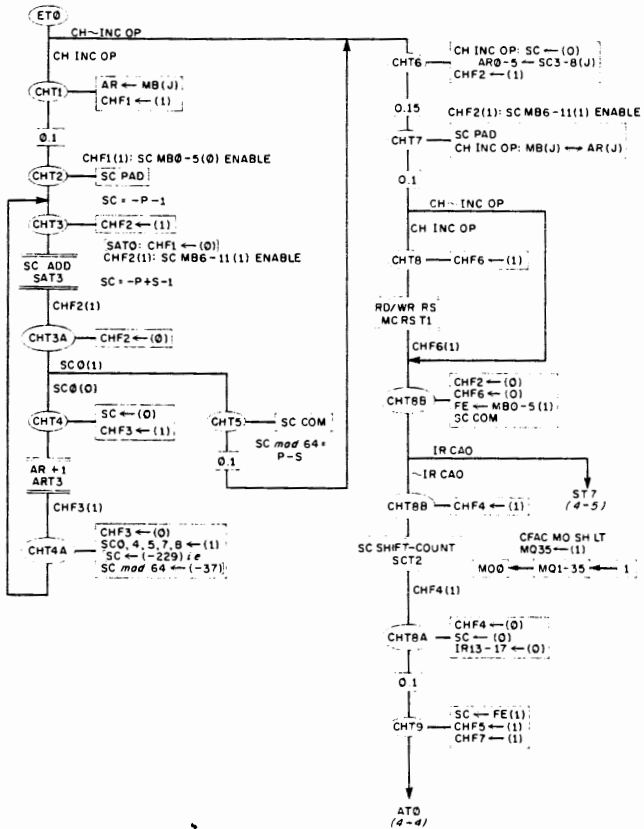
Pointer Format

P	S		I	X	Y
Ø-5	6-11	12	13	14-17	18-35

Fetch and increment pointer

Initial Registers
AR = Ø,E
MB = C(E)
MQ = Ø

Initial Gates
CH INC OP: FAC INH, FC(E) PSE
CH ~INC OP: FAC INH, FC(E)
~IR CAO: PC+1 INH
ET4 INH



CHARACTER OPERATIONS
133-137

IR = 001 011 XXX
IR FP/CH = IR0-2(001)
IR FP/CH3 = IR FP/CH ∧ IR3-5(011)
XXX *specifies instruction*

IR FP/CH3 ∧ IR6-8 → 011 = IR CAO, 100 = IR LDC1, 101 = IR LDC, 110 = IR DPC1, 111 = IR OPC

SECOND PART-CHF5(1)

CH LOAD = (IR LDC ∨ IR LDCI) ∧ CHF5(1)
CH DEP = (IR DPC ∨ IR DPCI) ∧ CHF5(1)

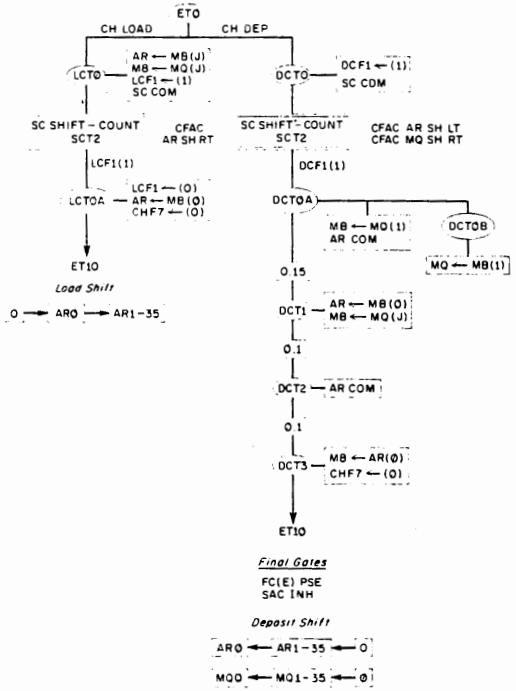
CH LOAD — *Fetch and load character*
CH DEP — *Fetch and deposit character*

Initial Registers

CH LOAD	CH DEP
AR = Ø,E	AR = Ø,E
MB = C(E)	MB = C(E)
MQ = Ø...Ø 1...1 (S)	MQ = Ø...Ø 1...1 (S)

Initial Gates

CH LOAD	CH DEP
FAC INH	FC(E) PSE
FC(E)	ET4 INH
ET4 INH	MB ← AR(J) INH(ETØ)



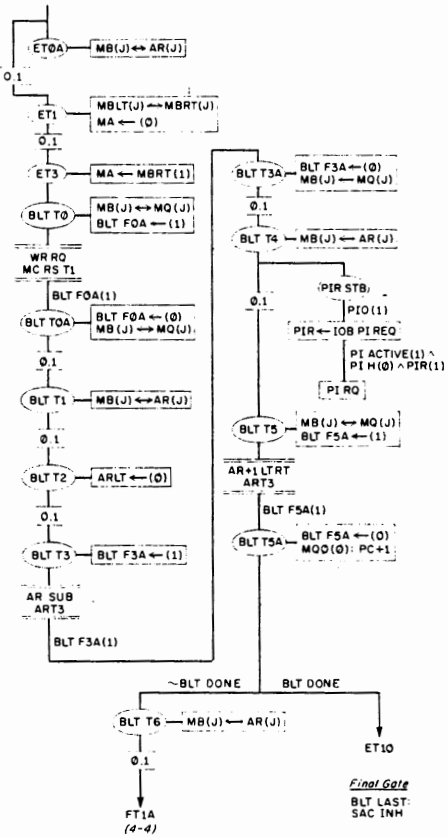
BLT
251

IR = ØIØ 1Ø1 ØØ1
IR 2XX = IR0-2(ØIØ)
IR 25X = IR 2XX ∧ IR3-5(1Ø1)
IR BLT = IR 25X ∧ IR6-8(ØØ1)
BLT DONE = MQØ(Ø) ∨ PI RQ
BLT LAST = IR BLT ∧ MQØ(Ø)

S = *Source address*
D = *Destination address*
C(S) → D
(S,D) + 1000001 → (S,D)
Repeat until D = E

Initial Registers
AR = D,S *initially* AC RT LT
MB = Ø,E
MQ = C(S) *initially* C(C(ACLT))

Initial Gates
FC(C(ACLT))
PC + 1 INH
ET4 INH

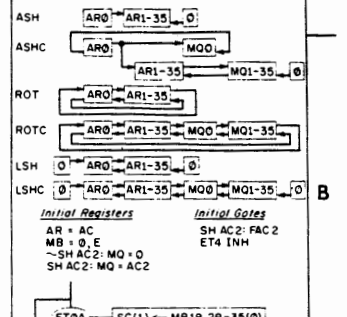


SHIFT OPERATIONS
240-242, 244-246

IR = ØIØ 1ØØ XXX
IR 2XX = IR0-2(ØIØ)
IR SH = IR 2XX ∧ IR3-5(1ØØ)
IR SH ∧ IR6-8 = ØØØ: IR ASH, ØØ1: IR ROT, ØIØ: IR LSH, 1ØØ: IR ASHC, 1Ø1: IR ROTC, 11Ø: IR LSHC

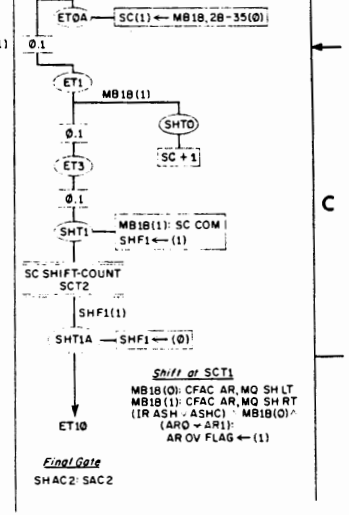
SHIFT OP = IR ASH ∨ IR ROT ∨ IR LSH ∨ IR ASHC ∨ IR ROTC ∨ IR LSHC
SH AC2 = IR ASHC ∨ IR ROTC ∨ IR LSHC
Shift E mod 256 places
~SH AC2: *f* [AC] → AC
SH AC2: *f* [AC, AC2] → AC

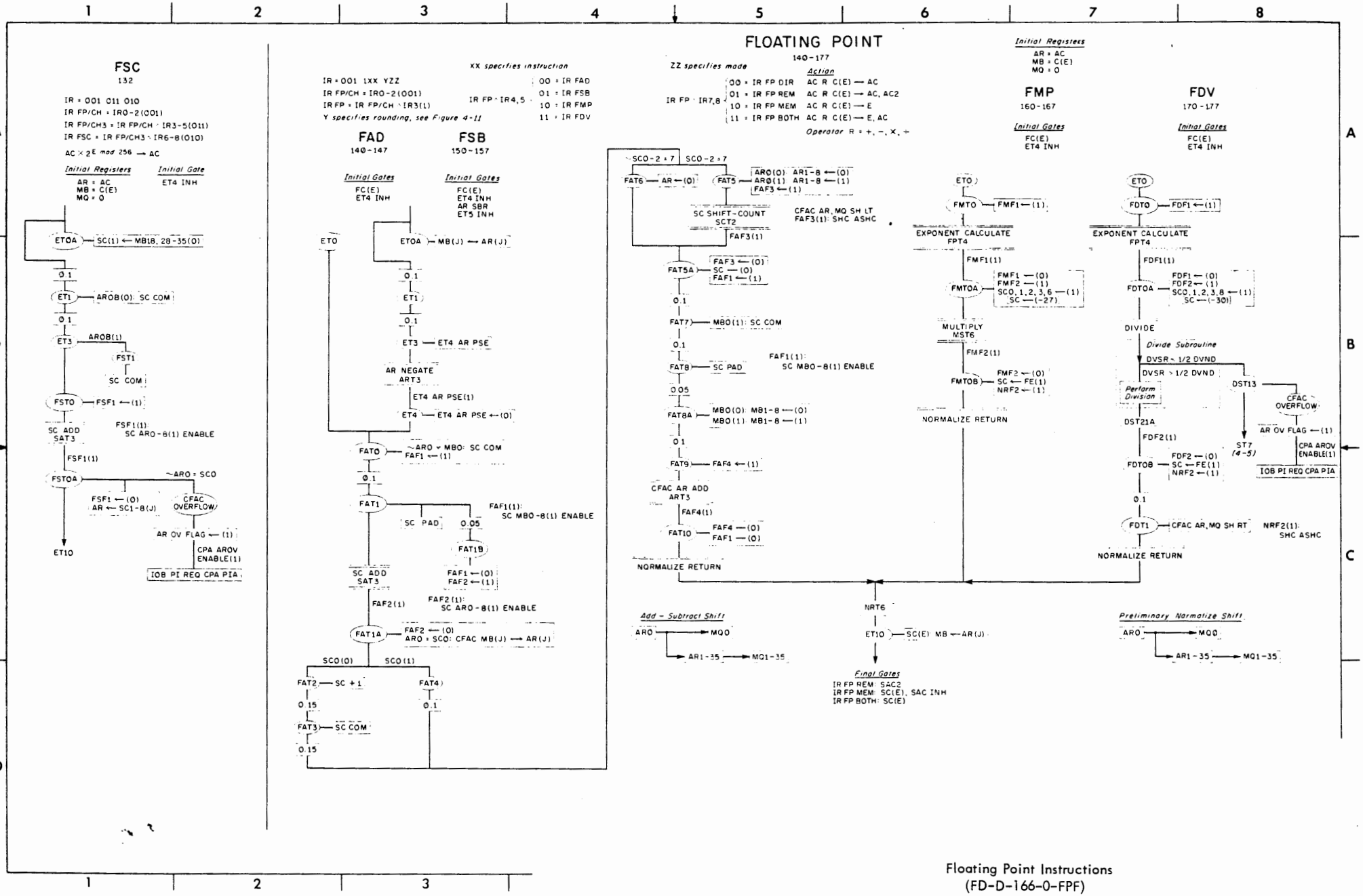
SHIFT OP AR, MQ CONNECTIONS
IR ASHC: SHC ASHC



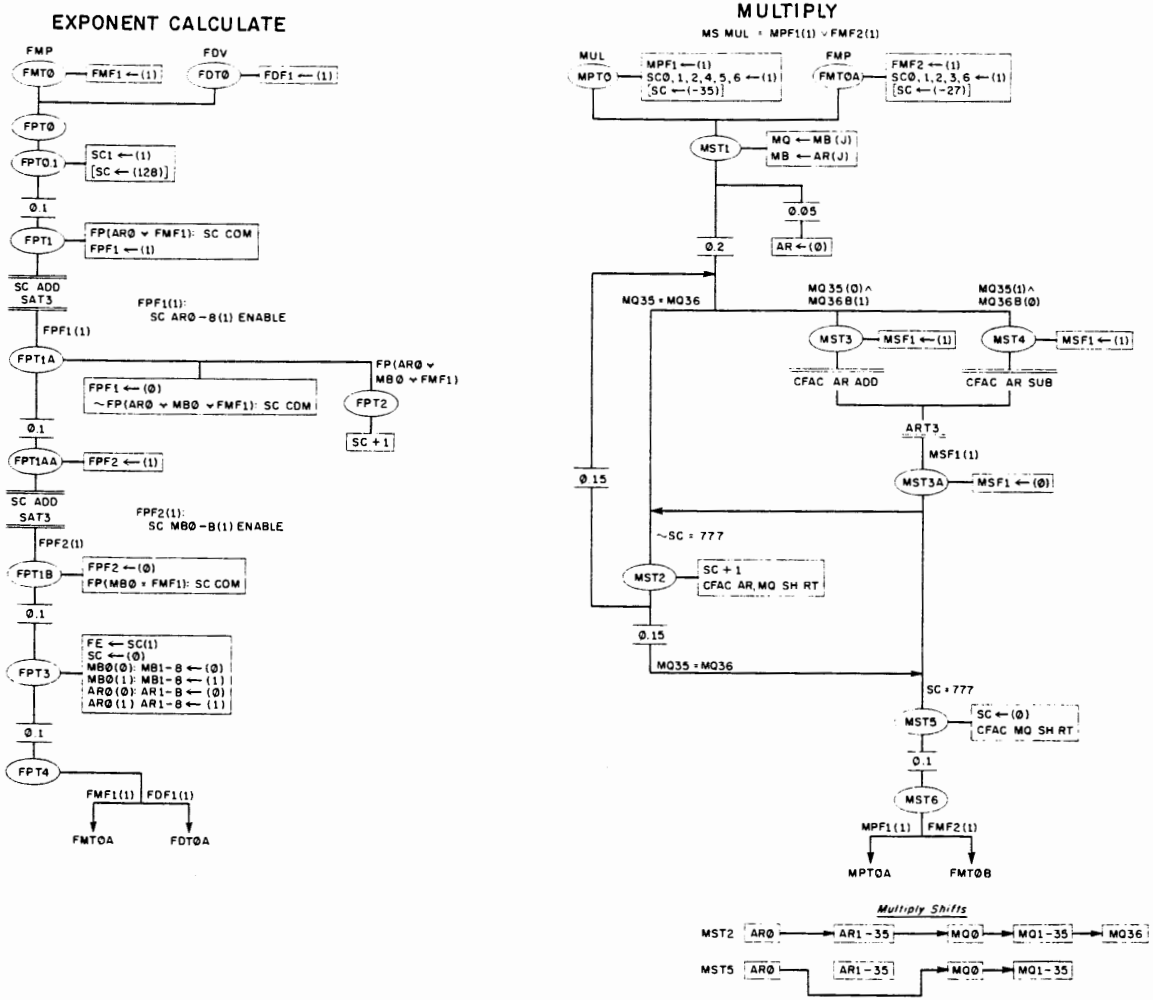
Initial Registers
AR = AC
MB = Ø,E
~SH AC2: MQ = Ø
SH AC2: MQ = AC2

Initial Gates
SH AC2: FAC 2
ET4 INH

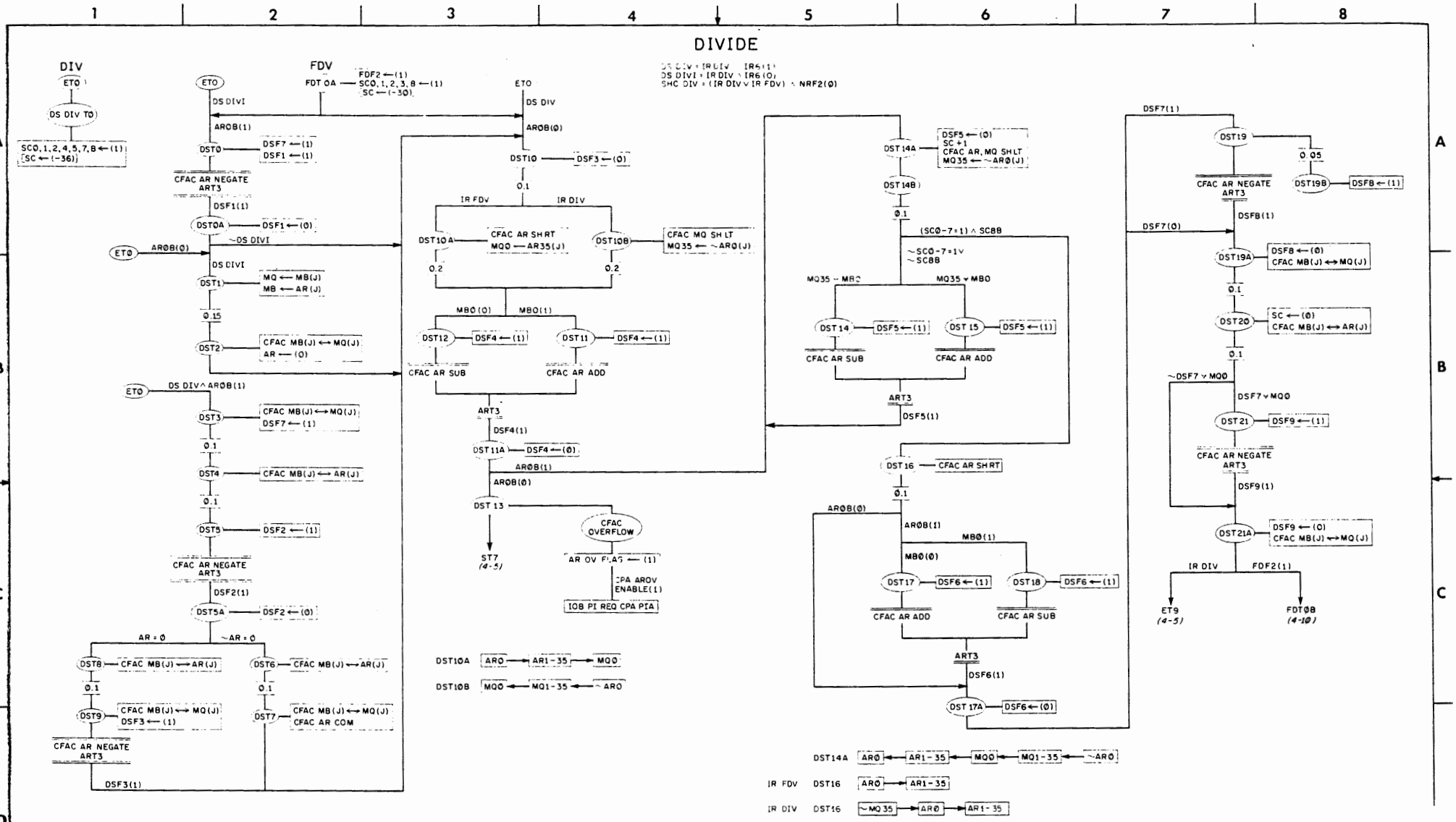




Floating Point Instructions
(FD-D-166-0-FPF)



Exponent Calculate, Multiply and Normalize Return Subroutines
(FD-D-166-0-EMNF)



Divide Subroutine
(FD-D-166-0-DSF)

IN-OUT TRANSFER

IR = 111 XXX XXX XYY Y

IR IOT A = IR0-2 (111)

IR IOT = IR IOT A ∧ ~Ex IR UUO

YYY specifies instruction

IR IOT ∧ IR10-12:
- 000 = IOT BLKI
- 001 = IOT DATAI
- 010 = IOT BLKO
- 011 = IOT DATAO
- 100 = IOT CONO
- 101 = IOT CONI
- 110 = IOT CONSZ
- 111 = IOT CONSO

XXX XXX X specifies device. IR3-9 outputs thru bus drivers to IOS3-9 lines on IO bus. Each IO control unit contains a decoding net that allows proper code to select device by goting in command signals from bus

Instruction Action

BLKI	C(E) + 1000001 → E, DEVICE BUFFER → C(C(ERT))
DATAI	DEVICE BUFFER → E
BLKO	C(E) + 1000001 → E, C(C(ERT)) → DEVICE BUFFER
DATAO	C(E) → DEVICE BUFFER
CONO	E → DEVICE CONTROL
CONI	STATUS → E
CONSZ	(STATUS ∧ E) = 0: PC + 1
CONSO	(STATUS ∧ E) ≠ 0: PC + 1

IOT BLK = IOT BLKI ∨ IOT BLKO
IOT OUTGOING = IOT DATAO ∨ IOT CONO
IOT STATUS = IOT CONI ∨ IOT CONSZ ∨ IOT CONSO
IOT DATAI ∨ O = IOT DATAI ∨ IOT DATAO
PI BLK RST = IOT DATAI ∨ O ∧ PI OV(0)
PI BLK RST ∧ PI CYC B(1): PI HOLD, PI RST
IOT CONSZ ∨ IOT CONSO: E LONG

BLK

Initial Registers
MB = C(E)
AR = 0, E
MO = 0

Initial Gates
FAC INH
FC(E) PSE
PC + 1 INH
ET4 INH
AR SBR
ET5 INH

ET0A — AR ← MB(J)
0.1
ET1
0.1
ET3 — ET4 AR PSE ← (1)
AR+1 LT RT ART3
ET4 AR PSE(1)
ET4 — ET4 AR PSE ← (0); MB(J) ↔ AR(J); EX ILL OP ← (0)
IOT T0 — IOT F0A ← (1)
RD/WR RS MC RS T1
IOT F0A(1)
IOT T0A — IOT F0A ← (0); IR12 ← (1); MA ← (0); PI CYC(1) ∧ AR CRY0(1): PI OV ← (1); PI CYC B(0) ∧ AR CRY0(0): PC + 1
0.2
FT0

Return to fetch and perform data instruction with C(MBRT) as E

	DATAI	DATAO	CONO	STATUS
Initial Registers	MB = ?, E AR = 0, E MO = 0	MB = C(E) AR = 0, E MO = 0	MB = ?, E AR = 0, E MO = 0	MB = ?, E AR = 0, E MO = 0
Initial Gates	FAC INH ET5 INH	FAC INH FC(E) ET5 INH	FAC INH ET5 INH	FAC INH ET5 INH
ET0A	PI HOLD: PIH ← PI CH RO	AR ← MB(J) PI HOLD: PIH ← PI CH RO	MBLT ← MBRT(J)	MB ← AR(J)
0.1				
ET1	AR ← (0) PI RST: PIH(0) ← PI OK(1)	PI RST: PIH(0) ← PI OK(1)		AR ← (0)
0.1				
ET3				
ET4			ARLT ← MBLT(J)	

IOT GO ← (1)
IOT RESET
~IOT RESET — *Wait for* ~IOT RESET
INITIAL SETUP 1.0
IOT T2 — IOT GO ← (0); IOB DATAO CLEAR; IOB CONO CLEAR
RESTART 1.0
IOT T3 — AR ← IOB(1); IOB DATAO SET; IOB CONO SET
0.2
ET5
FINAL SETUP 1.5
RESET 2.0

IOT OUTGOING: IOB ← AR(1)
IOT DATAI: IOB ← DATAI
IOT STATUS: IOB ← STATUS
IOT RESET
IOB BUS RESET

~E LONG
ET10 — IOT DATAO: PI HOLD: PI CYC ← (0); IOT DATAI: MB ← AR(J), PI HOLD: PI CYC ← (0); IOT CONI: MB ← AR(J)

0.1
E LONG
ET6 — AR ← MB(0)
0.1
ET7
0.1
ET8
0.2
ET9 — IOT CONSZ: AR = 0: PC + 1; IOT CONSO: ~AR = 0: PC + 1
ET10

Final Gates
IOT DATAI ∨ IOT CONI: SC(E)
SAC INH

CPA

000

CPA = IOS3-9 (000 000 0)

IOB CONO SET — CPA — CPA CONO SET — IOB19 — IOB RESET — *Clear IO equipment*
MR START — CPA CLEAR — *Clear CPA flags*
IOB DATAO CLEAR — CPA — EX CLR — PR ← (0); RLR ← (0)
IOB DATAO SET — CPA — Ex SET — PR ← IOB0-7(1); RLR ← IOB18-25(1) — *Assert* IOB ← DATAI — CPA — AR ← DATA SW(1)

IOB	CPA CONO SET	CPA STATUS
18	CPA PDL OV ← (0)	
19	IOB RESET	CPA PDL OV(1)
20	CPA IOT USER ← (1)	CPA IOT USER(1)
21	CPA IOT USER ← (0)	EX USER B(1)
22	CPA ILLEG OP ← (0)	CPA ILLEG OP(1)
23	CPA NON EXIST MEM ← (0)	CPA NON EXIST MEM(1)
24	CPA CLOCK ENABLE ← (0)	
25	CPA CLOCK ENABLE ← (1)	CPA CLOCK ENABLE(1)
26	CPA CLOCK FLAG ← (0)	CPA CLOCK FLAG(1)
27	CPA PC CHG ENABLE ← (0)	
28	CPA PC CHG ENABLE ← (1)	CPA PC CHG ENABLE(1)
29	AR PC CHG FLAG ← (0)	AR PC CHG FLAG B(1)
30	CPA AROV ENABLE ← (0)	
31	CPA AROV ENABLE ← (1)	CPA AROV ENABLE(1)
32	AR OV FLAG ← (0)	AR OV FLAG B(1)
33-35	CPA PIA ← IOB33-35(1)	CPA PIA B(1)

60 times per second from power control: CPA CLOCK FLAG ← (1)
MC ILLEG ADDRESS: CPA ILLEG OP ← (1)
MC NON EXIST MEM: CPA NON EXIST MEM ← (1)
ET10 ∧ [MB(J) ↔ AR(J) (ET10)] ∧ AR CRY0(1): CPA PDL OV ← (1)

IOB PI REQ CPA PIA
- CPA ILLEG OP(1)
- CPA NON EXIST MEM(1)
- CPA PDL OV(1)
- CPA CLOCK ENABLE(1) ∧ CPA CLOCK FLAG(1)
- CPA PC CHG ENABLE(1) ∧ AR PC CHG FLAG B(1)
- CPA AROV ENABLE(1) ∧ AR OV FLAG(1)

PI

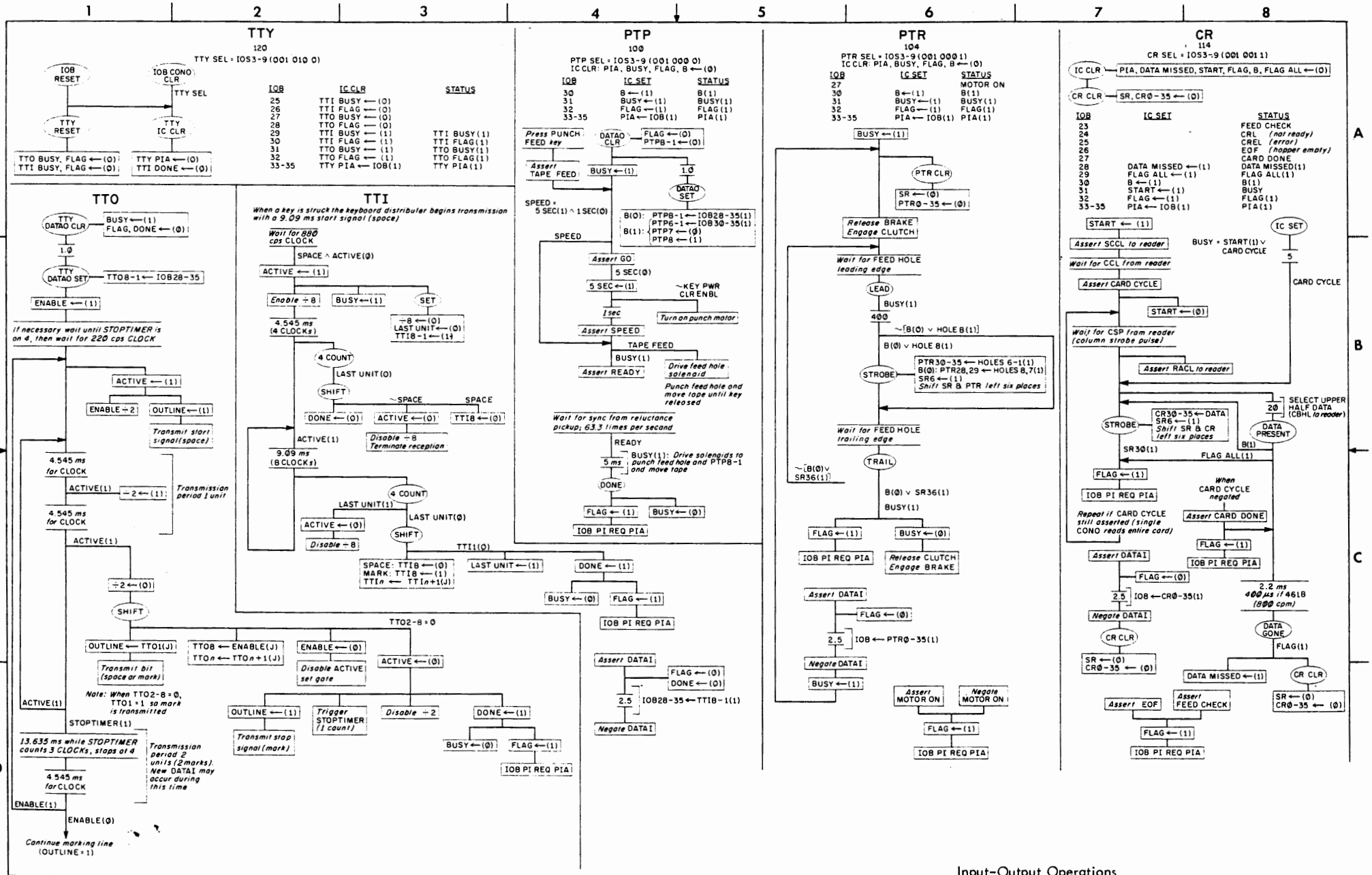
004

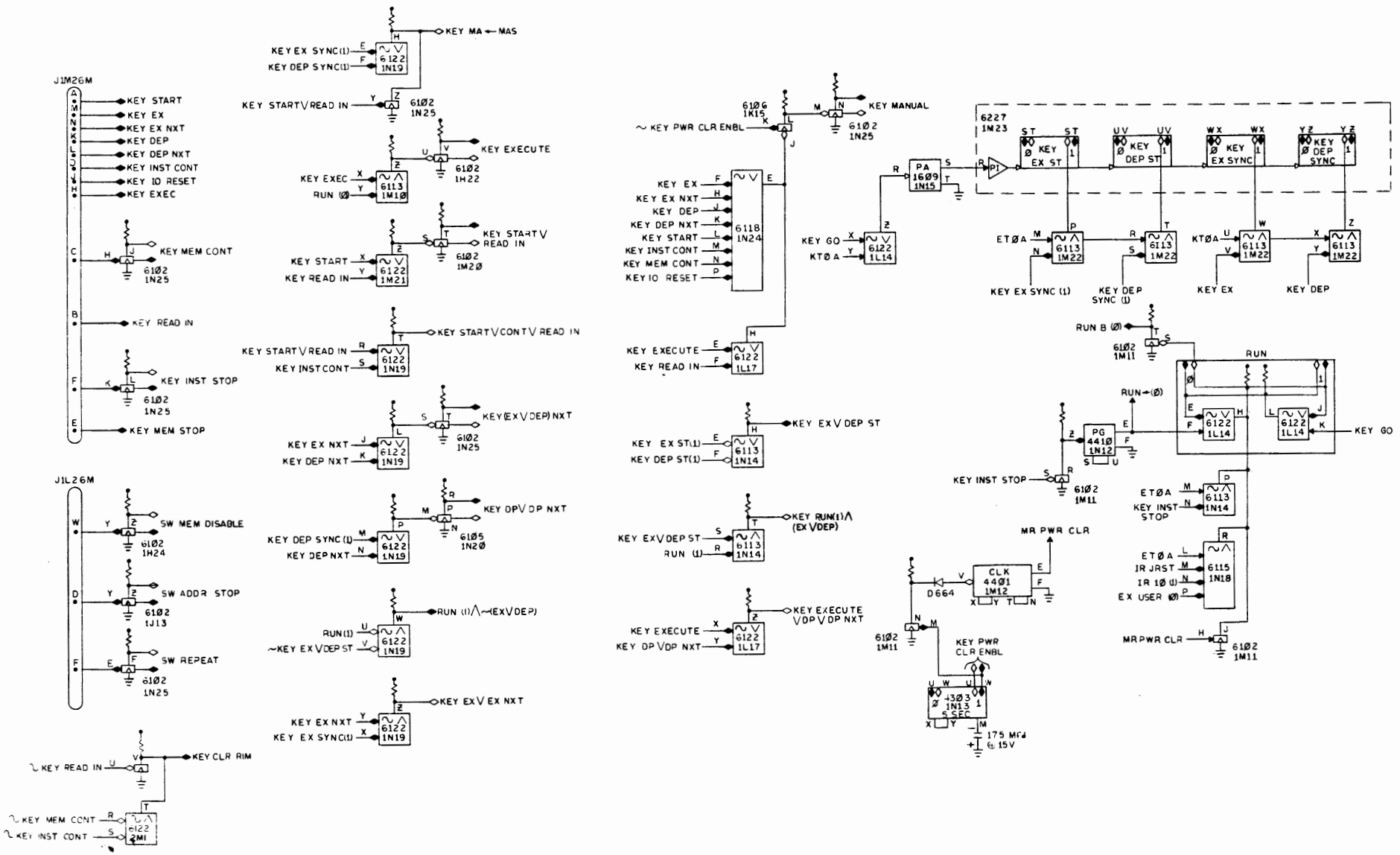
PI SELECT = IOS3-9(000 000 1)

MR START — PI OV ← (0); PI CYC ← (0)
IOB CONO CLEAR — PI SELECT ∧ IOB23 — PI RESET — PI ACTIVE ← (0); PIO ← (0); PIR ← (0); PIH ← (0)

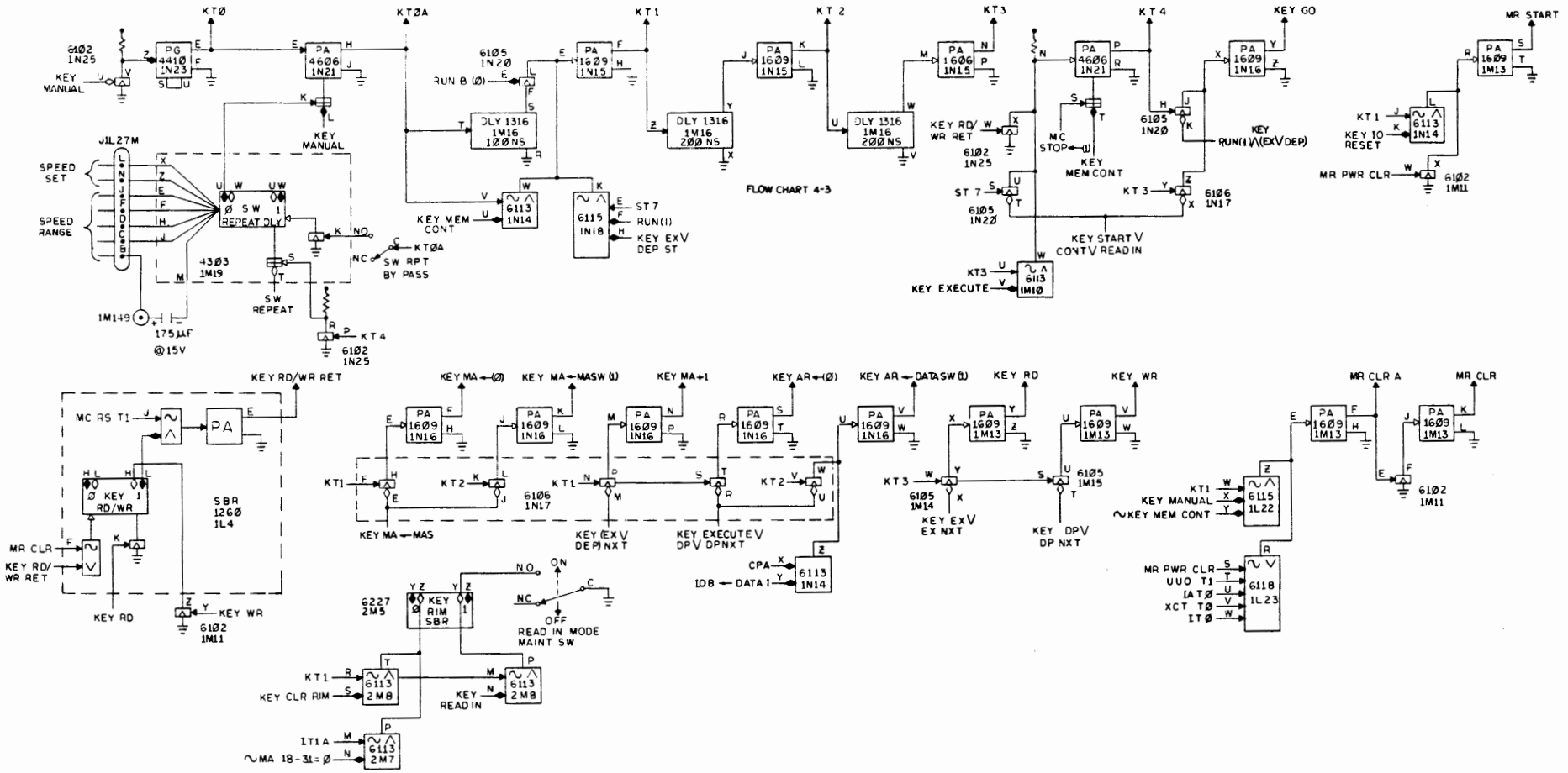
IOB	PI CONO SET	PI STATUS
24	PIR ← IOB 29-35(1)	
25	PIO ← IOB 29-35(1)	
26	PIO(0) ← IOB29-35(1)	
27	PI ACTIVE ← (0)	
28	PI ACTIVE ← (1)	PI ACTIVE(1)
29-35		PIO1-7(1)

IOT Instructions
(FD-D-166-0-IOTF)

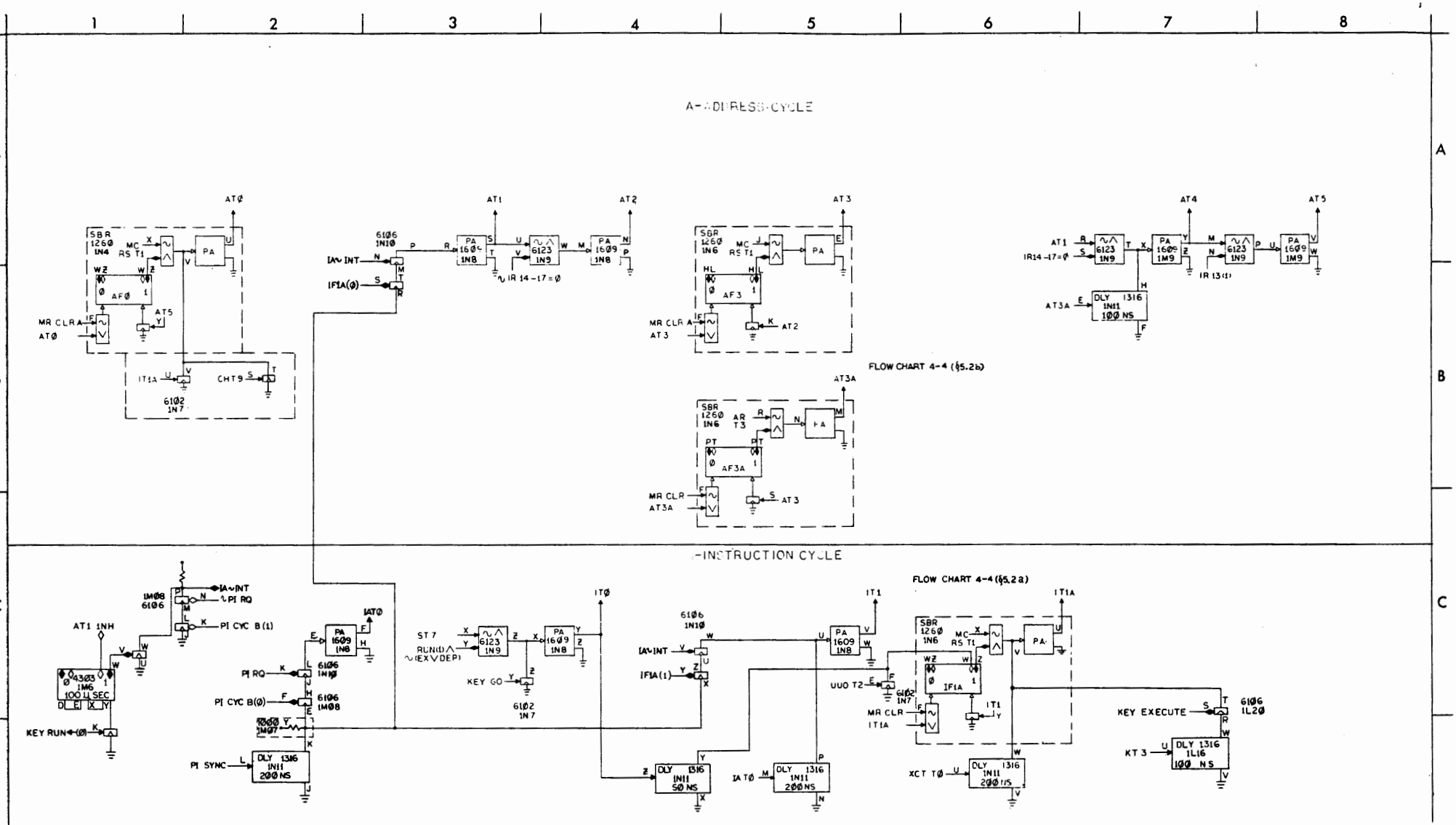

TTY
120
TTY SEL = IOS3-9 (001 010 0)
IOB RESET
IOB CONO CLR
TTY SEL
TTY RESET
TTY IC CLR
TTO BUSY, FLAG ← (0)
TTI BUSY, FLAG ← (0)
TTY PIA ← (0)
TTI DONE ← (0)
IOB | IC CLR | STATUS
25 | TTI BUSY ← (0)
26 | TTI FLAG ← (0)
27 | TTO BUSY ← (0)
28 | TTO FLAG ← (0)
29 | TTI BUSY ← (1) | TTI BUSY(1)
30 | TTI FLAG ← (1) | TTI FLAG(1)
31 | TTO BUSY ← (1) | TTO BUSY(1)
32 | TTO FLAG ← (1) | TTO FLAG(1)
33-35 | TTY PIA ← IOB(1) | TTY PIA(1)
TTO
TTY DATAO CLR
BUSY ← (1)
FLAG, DONE ← (Ø)
1.Ø
TTY DATAO SET
TTO8-1 ← IOB28-35
ENABLE ← (1)
If necessary wait until STOPTIMER is on 4, then wait for 220 cps CLOCK
ACTIVE ← (1)
ENABLE ÷ 2
OUTLINE ← (1)
Transmit start signal (space)
4.545 ms for CLOCK
ACTIVE(1)
÷2 ← (1)
Transmission period 1 unit
4.545 ms for CLOCK
ACTIVE(1)
÷2 ← (Ø)
SHIFT
OUTLINE ← TTO1(J)
Transmit bit (space or mark)
TTO8 ← ENABLE(J)
TTOn ← TTOn+1(J)
ENABLE ← (Ø)
Disable ACTIVE set gate
TTO2-8 = Ø
ACTIVE ← (Ø)
Note: When TTO2-8 = Ø, TTO1 = 1 so mark is transmitted
ACTIVE(1)
STOPTIMER(1)
13.635 ms while STOPTIMER counts 3 CLOCKs, stops at 4
Transmission period 2 units (2 marks). New DATAI may occur during this time
4.545 ms for CLOCK
ENABLE(1)
ENABLE(Ø)
Continue marking line (OUTLINE = 1)
OUTLINE ← (1)
Transmit stop signal (mark)
Trigger STOPTIMER (1 count)
Disable ÷2
DONE ← (1)
BUSY ← (Ø)
FLAG ← (1)
IOB PI REQ PIA
TTI
When a key is struck the keyboard distributer begins transmission with a 9.09 ms start signal (space)
Wait for 880 cps CLOCK
SPACE ∧ ACTIVE(Ø)
ACTIVE ← (1)
Enable ÷ 8
BUSY ← (1)
SET
÷8 ← (Ø)
LAST UNIT ← (Ø)
TTI8-1 ← (1)
4.545 ms (4 CLOCKs)
4 COUNT
LAST UNIT(Ø)
SHIFT
~SPACE
SPACE
DONE ← (Ø)
ACTIVE ← (Ø)
TTI8 ← (Ø)
ACTIVE(1)
Disable ÷8 Terminate reception
9.09 ms (8 CLOCKs)
4 COUNT
LAST UNIT(1)
LAST UNIT(Ø)
ACTIVE ← (Ø)
SHIFT
Disable ÷8
TTI1(Ø)
SPACE: TTI8 ← (Ø)
MARK: TTI8 ← (1)
TTIn ← TTIn+1(J)
LAST UNIT ← (1)
DONE ← (1)
BUSY ← (Ø)
FLAG ← (1)
IOB PI REQ PIA
PTP
100
PTP SEL = IOS3-9 (001 000 0)
IC CLR: PIA, BUSY, FLAG, B ← (Ø)
IOB | IC SET | STATUS
30 | B ← (1) | B(1)
31 | BUSY ← (1) | BUSY(1)
32 | FLAG ← (1) | FLAG(1)
33-35 | PIA ← IOB(1) | PIA(1)
Press PUNCH FEED key
DATAO CLR
FLAG ← (Ø)
PTP8-1 ← (Ø)
Assert TAPE FEED
BUSY ← (1)
1.Ø
DATAO SET
SPEED = 5 SEC(1) ∧ 1 SEC(Ø)
B(Ø): PTP8-1 ← IOB28-35(1)
PTP6-1 ← IOB30-35(1)
B(1): PTP7 ← (Ø)
PTP8 ← (1)
SPEED
Assert GO
5 SEC(Ø)
5 SEC ← (1)
~KEY PWR CLR ENBL
1 sec
Turn on punch motor
Assert SPEED
TAPE FEED
BUSY(1)
Assert READY
Drive feed hole solenoid
Punch feed hole and move tape until key released
Wait for sync from reluctance pickup; 63.3 times per second
READY
5 ms
BUSY(1): Drive solenoids to punch feed hole and PTP8-1 and move tape
DONE
FLAG ← (1)
BUSY ← (Ø)
IOB PI REQ PIA
Assert DATAI
FLAG ← (Ø)
DONE ← (Ø)
2.5
IOB28-35 ← TTI8-1(1)
Negate DATAI
PTR
104
PTR SEL = IOS3-9 (001 000 1)
IC CLR: PIA, BUSY, FLAG, B ← (Ø)
IOB | IC SET | STATUS
27 | | MOTOR ON
30 | B ← (1) | B(1)
31 | BUSY ← (1) | BUSY(1)
32 | FLAG ← (1) | FLAG(1)
33-35 | PIA ← IOB(1) | PIA(1)
BUSY ← (1)
PTR CLR
SR ← (Ø)
PTRØ-35 ← (Ø)
Release BRAKE
Engage CLUTCH
Wait for FEED HOLE leading edge
LEAD
BUSY(1)
400
~[B(Ø) ∨ HOLE B(1)]
B(Ø) ∨ HOLE B(1)
STROBE
PTR30-35 ← HOLES 6-1(1)
B(Ø): PTR28,29 ← HOLES 8,7(1)
SR6 ← (1)
Shift SR & PTR left six places
Wait for FEED HOLE trailing edge
TRAIL
~[B(Ø) ∨ SR36(1)]
B(Ø) ∨ SR36(1)
BUSY(1)
FLAG ← (1)
BUSY ← (Ø)
IOB PI REQ PIA
Release CLUTCH
Engage BRAKE
Assert DATAI
FLAG ← (Ø)
2.5
IOB ← PTRØ-35(1)
Negate DATAI
BUSY ← (1)
Assert MOTOR ON
Negate MOTOR ON
FLAG ← (1)
IOB PI REQ PIA
CR
114
CR SEL = IOS3-9 (001 001 1)
IC CLR
PIA, DATA MISSED, START, FLAG, B, FLAG ALL ← (Ø)
CR CLR
SR, CRØ-35 ← (Ø)
IOB | IC SET | STATUS
23 | | FEED CHECK
24 | | CRL (not ready)
25 | | CREL (error)
26 | | EOF (hopper empty)
27 | | CARD DONE
28 | DATA MISSED ← (1) | DATA MISSED(1)
29 | FLAG ALL ← (1) | FLAG ALL(1)
30 | B ← (1) | B(1)
31 | START ← (1) | BUSY
32 | FLAG ← (1) | FLAG(1)
33-35 | PIA ← IOB(1) | PIA(1)
START ← (1)
Assert SCCL to reader
Wait for CCL from reader
Assert CARD CYCLE
IC SET
BUSY = START(1) ∨ CARD CYCLE
5
CARD CYCLE
START ← (Ø)
Wait for CSP from reader (column strobe pulse)
Assert RACL to reader
20
SELECT UPPER HALF DATA (CBHL to reader)
STROBE
CR30-35 ← DATA
SR6 ← (1)
Shift SR & CR left six places
DATA PRESENT
B(1)
SR30(1)
FLAG ALL(1)
FLAG ← (1)
IOB PI REQ PIA
When CARD CYCLE negated
Repeat if CARD CYCLE still asserted (single CONO reads entire card)
Assert CARD DONE
FLAG ← (1)
IOB PI REQ PIA
Assert DATAI
FLAG ← (Ø)
2.5
IOB ← CRØ-35(1)
Negate DATAI
2.2 ms 400 µs if 461B (800 cpm)
DATA GONE
FLAG(1)
CR CLR
SR ← (Ø)
CRØ-35 ← (Ø)
DATA MISSED ← (1)
CR CLR
SR ← (Ø)
CRØ-35 ← (Ø)
Assert EOF
Assert FEED CHECK
FLAG ← (1)
IOB PI REQ PIA



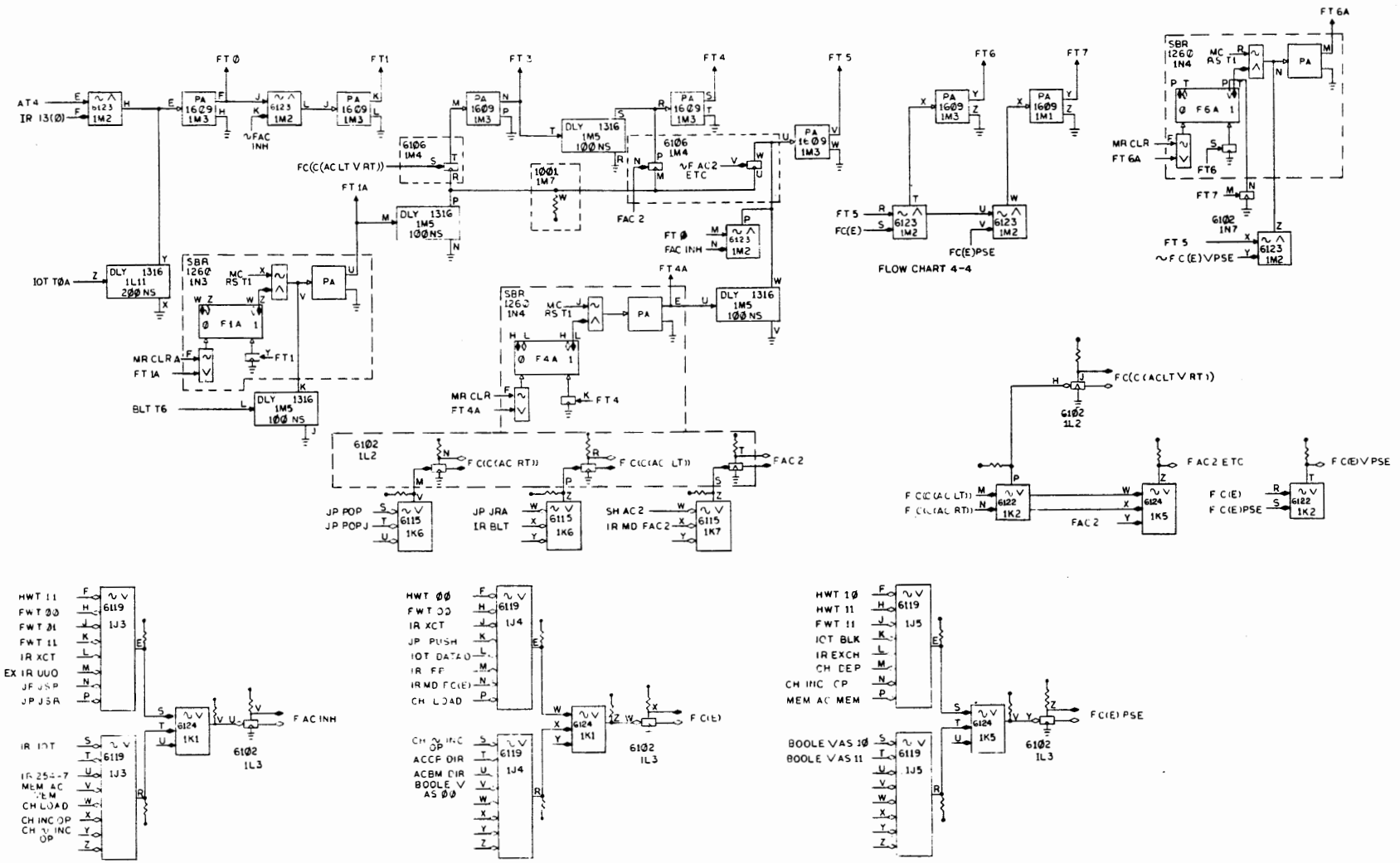
Key 1 (BS-D-166-0-KEY-1)



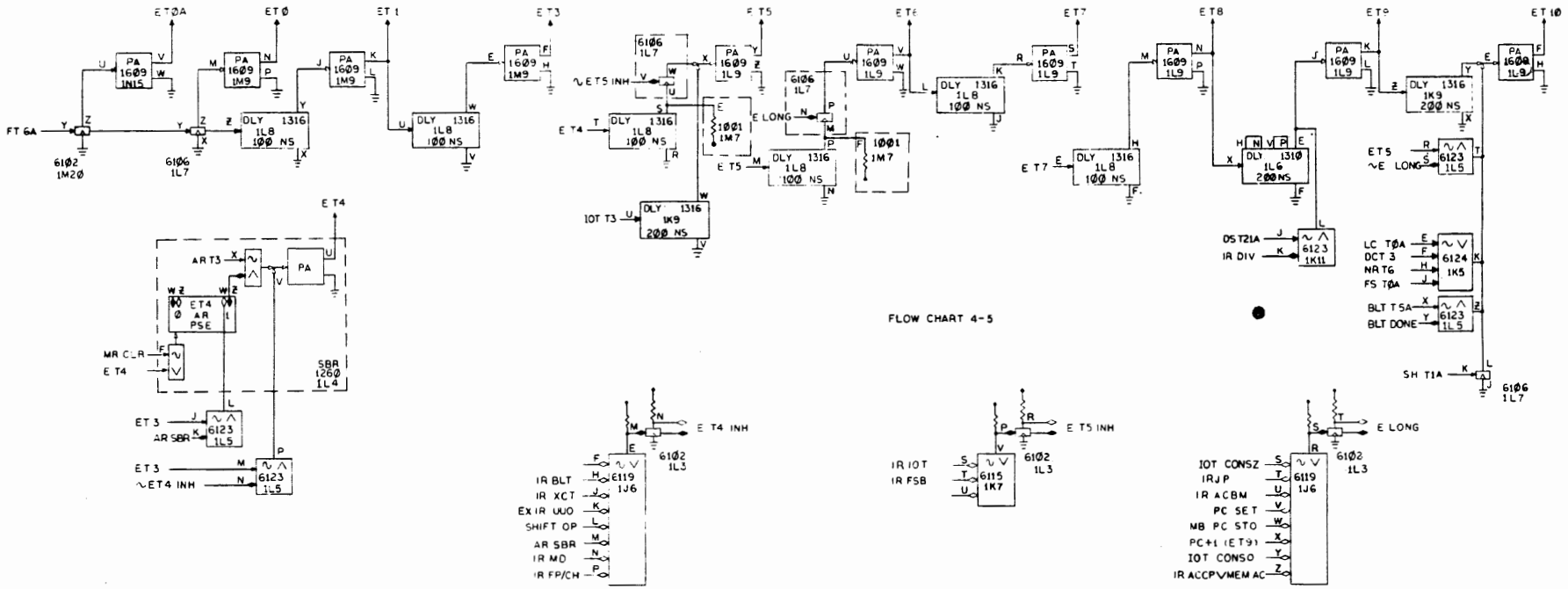
Key 2 (BS-D-166-0-KEY-2)



I, A - Instruction and Address Cycles
(BS-D-166-0-1A)

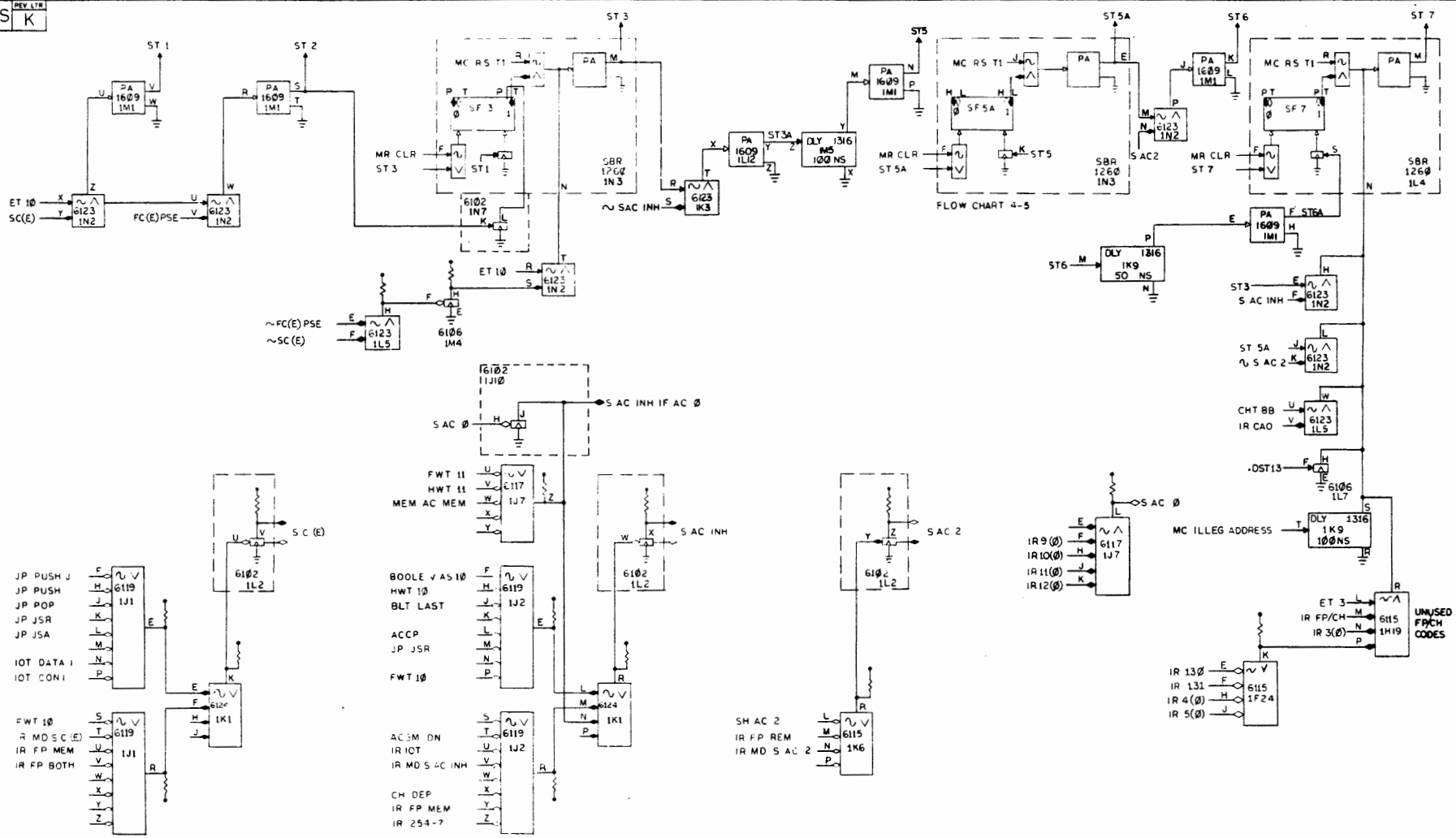


F - Fetch Cycle (BS-D-166-0-F)

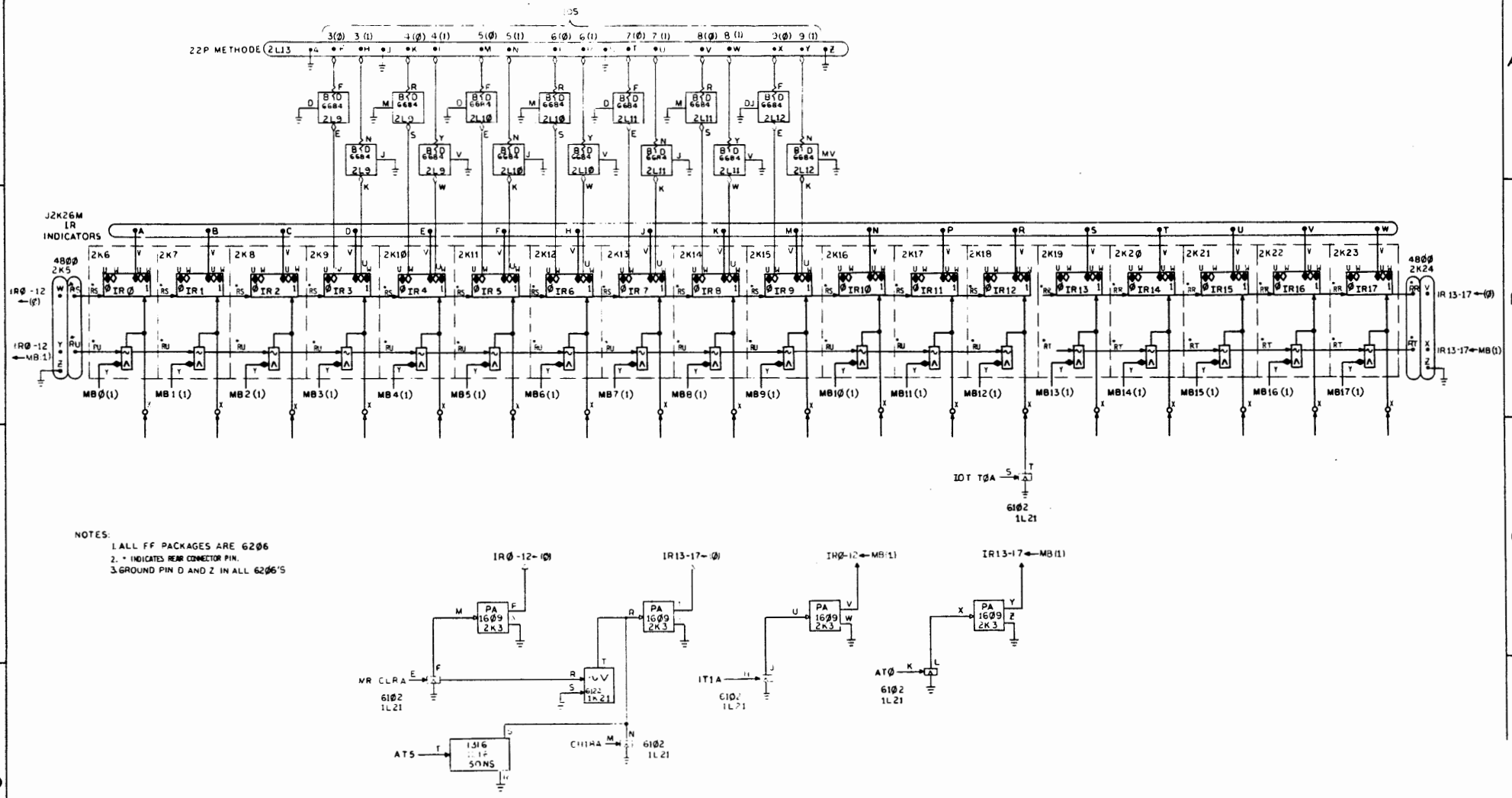


FLOW CHART 4-5

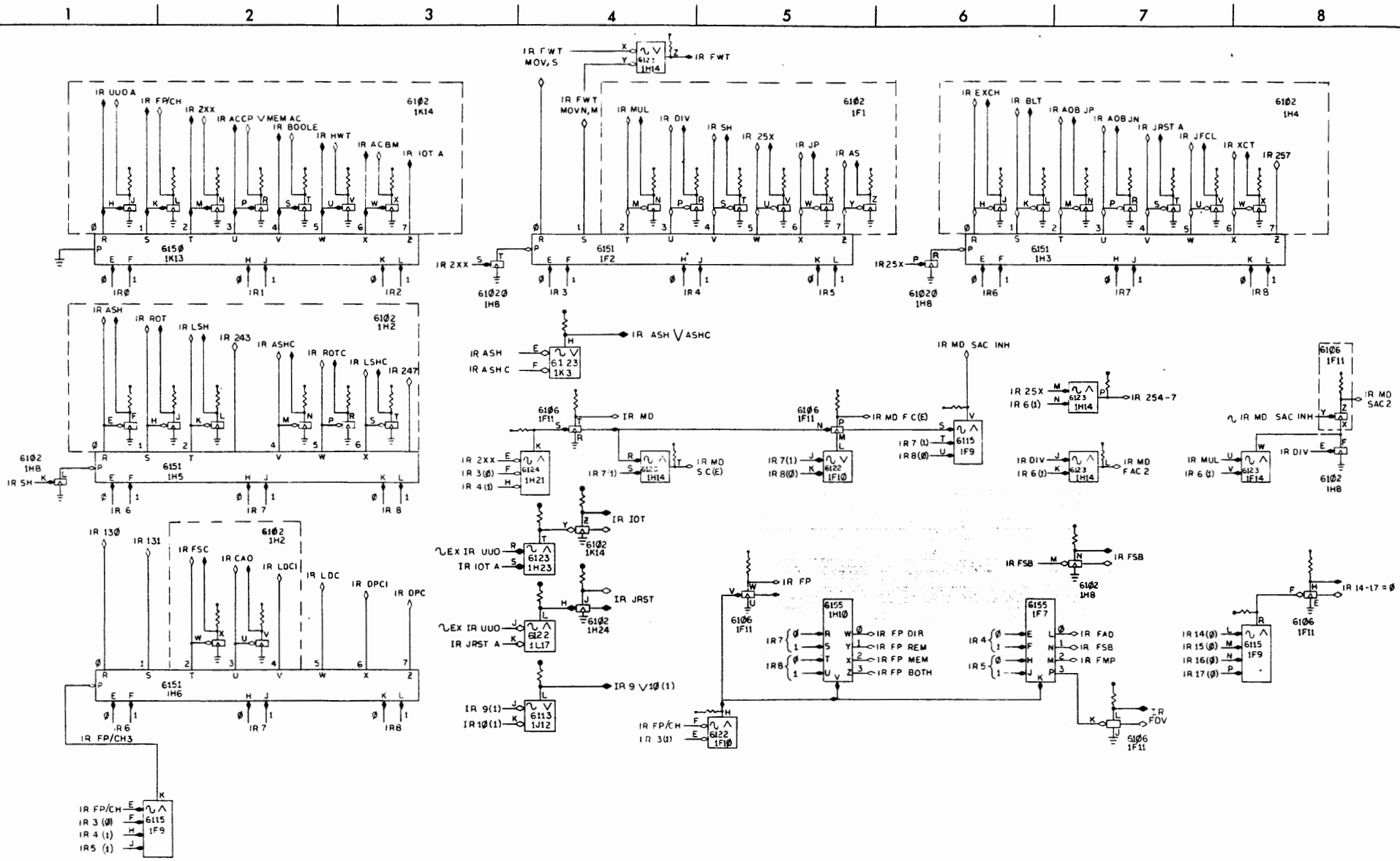
D-166-Ø-S REV LTR K



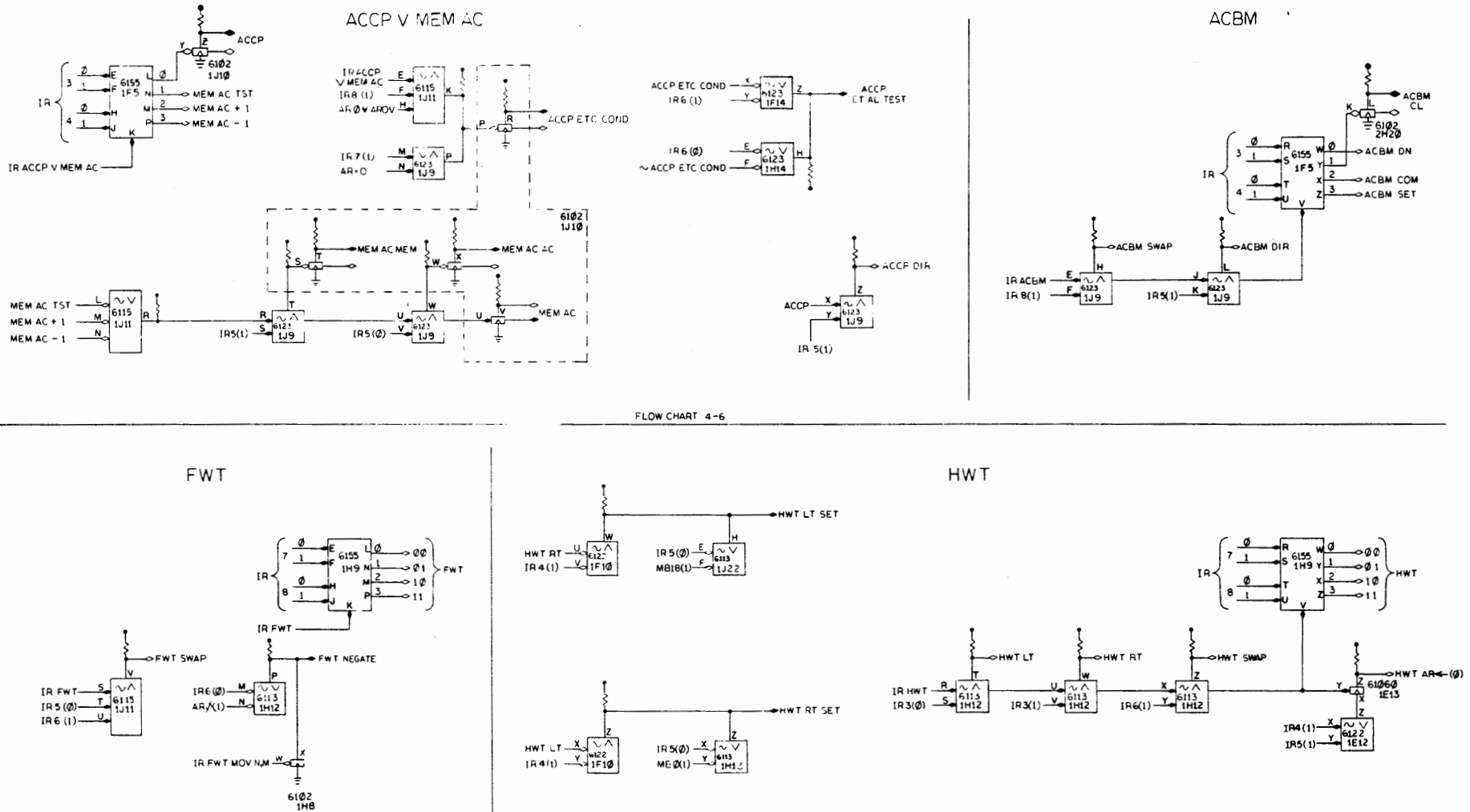
S - Store Cycle (BS-D-166-0-S)



IR – Instruction Register
(BS-D-166-0-IR-1)



IR Decoding (BS-D-166-0-IR-2)

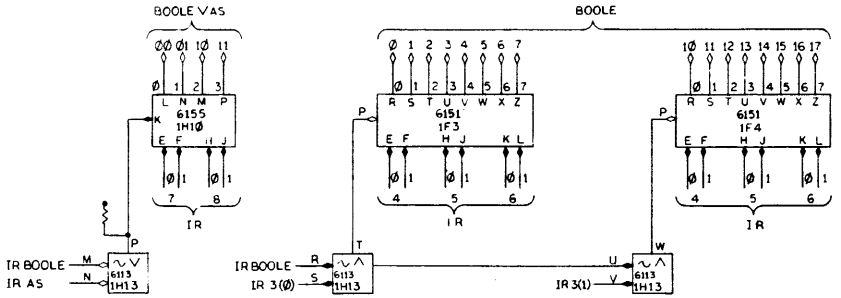


FLOW CHART 4-6

ACCP, MEMAC, ACBM, FWT, HWT
(BS-D-166-0-INS-1)

BOOLE

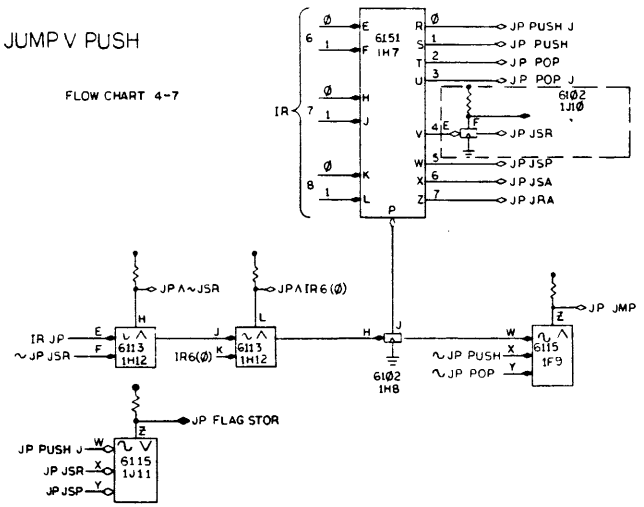
		RESULT BIT FOR OPERAND BIT CONFIGURATION				ET0 T1	ET1 T2		ET4 T3
		MEM AC	MEM AC	MEM AC	MEM AC	AR CLR	AR COM	AR←MB(Ø)	AR←MB(1)	AR←MB ▼	AR COM
		Ø Ø	Ø 1	1 Ø	1 1						
SETZ	Ø	Ø Ø	Ø	Ø	Ø	✓					
AND	MEM ∧ AC	Ø1	Ø	Ø	Ø	1			✓	✓		
ANDCA	MEM ∧ ∼AC	Ø2	Ø	Ø	1	Ø		✓		✓		
SETM	MEM	Ø3	Ø	Ø	1	1	✓			✓		
ANDCM	∼MEM ∧ AC	Ø4	Ø	1	Ø	Ø		✓		✓		✓
SETA	AC	Ø5	Ø	1	Ø	1						
XOR	MEM ∨ AC	Ø6	Ø	1	1	Ø				✓		
IOR	MEM ∨ AC	Ø7	Ø	1	1	1				✓		
ANDCB	∼MEM ∧ ∼AC	1Ø	1	Ø	Ø	Ø				✓		✓
EQV	MEM ≡ AC	11	1	Ø	Ø	1		✓			✓	✓
SETCA	∼AC	12	1	Ø	1	Ø		✓				
ORCA	MEM ∨ ∼AC	13	1	Ø	1	1		✓		✓		
SETCM	∼MEM	14	1	1	Ø	Ø	✓				✓	✓
ORCM	∼MEM ∨ AC	15	1	1	Ø	1		✓	✓			✓
ORCB	∼MEM ∨ ∼AC	16	1	1	1	Ø		✓	✓			✓
SETO	1	17	1	1	1	1	✓					✓



FLOW CHART 4-8

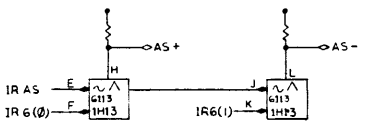
JUMP V PUSH

FLOW CHART 4-7



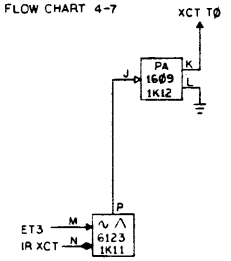
AS

FLOW CHART 4-8

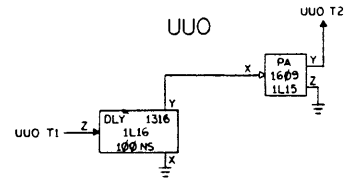


XCT

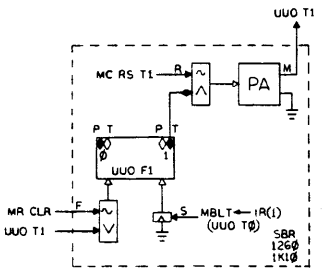
FLOW CHART 4-7



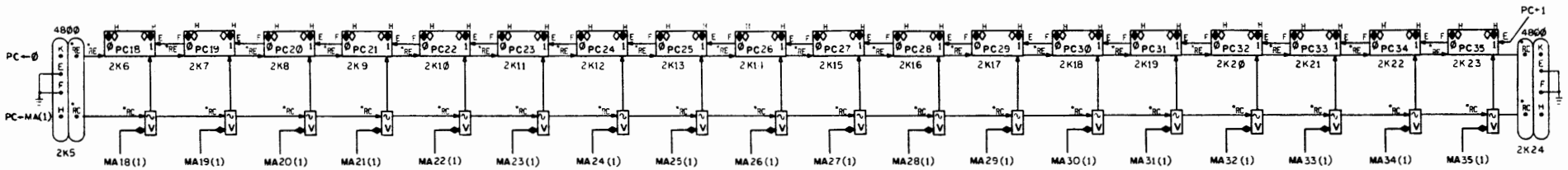
UUO



FLOW CHART 4-7



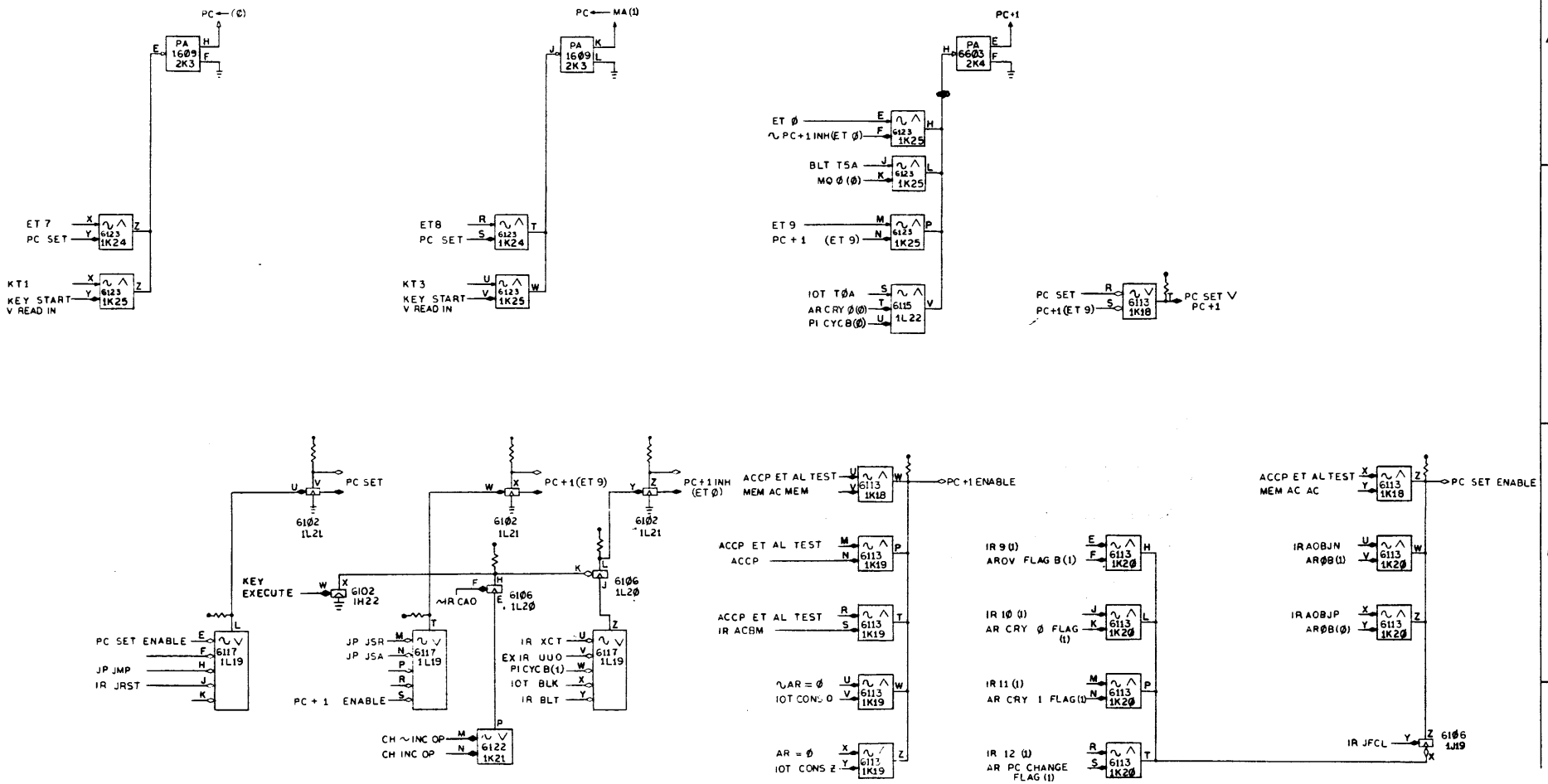
BOOLE, JP, AS, XCT, UUO
(BS-D-166-0-INS-2)



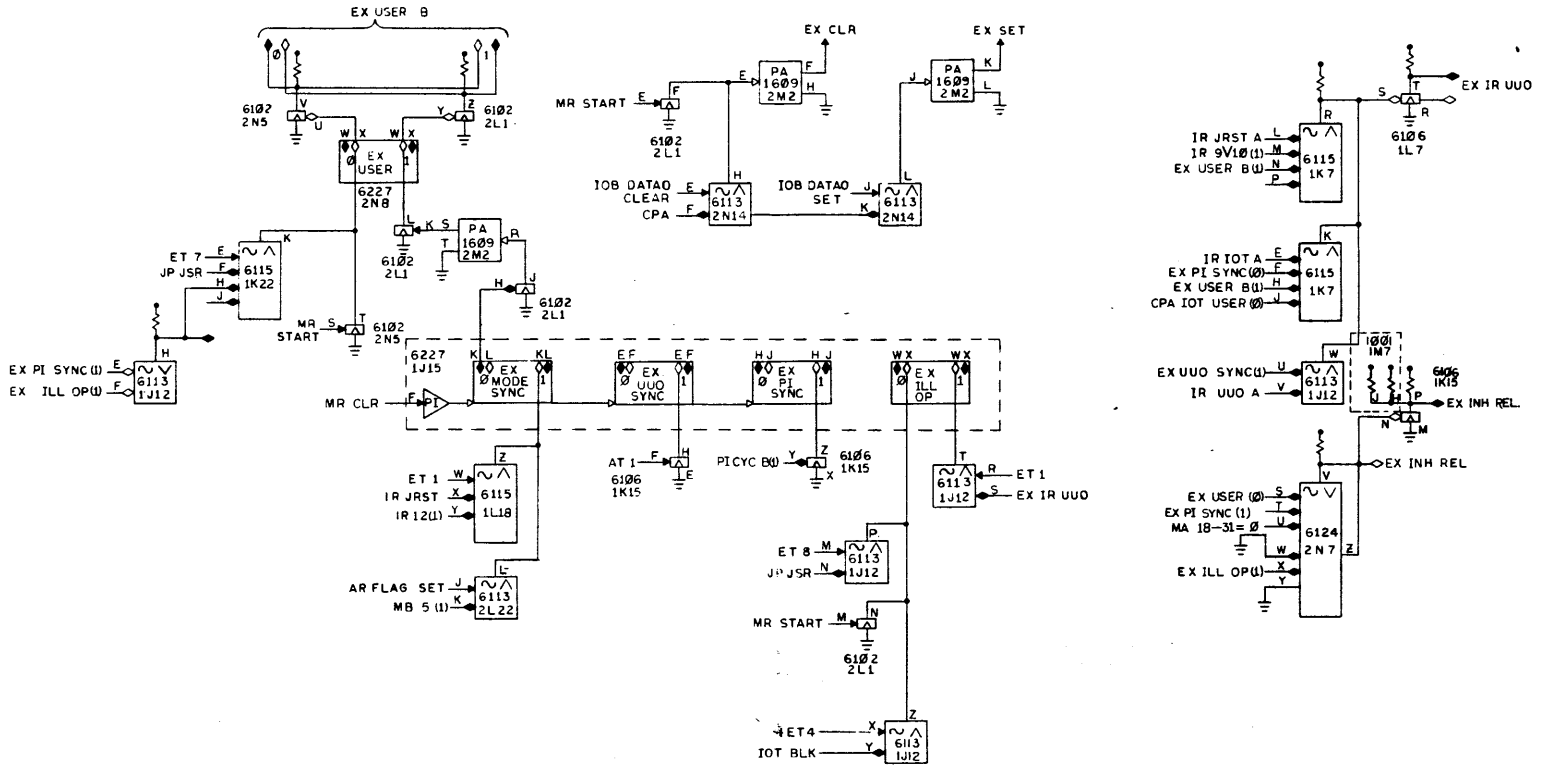
NOTES:

1. ALL FF PACKAGES ARE 6284
2. * INDICATES REAR CONNECTOR PIN
3. GROUND PIN D AND Z IN ALL 6206'S

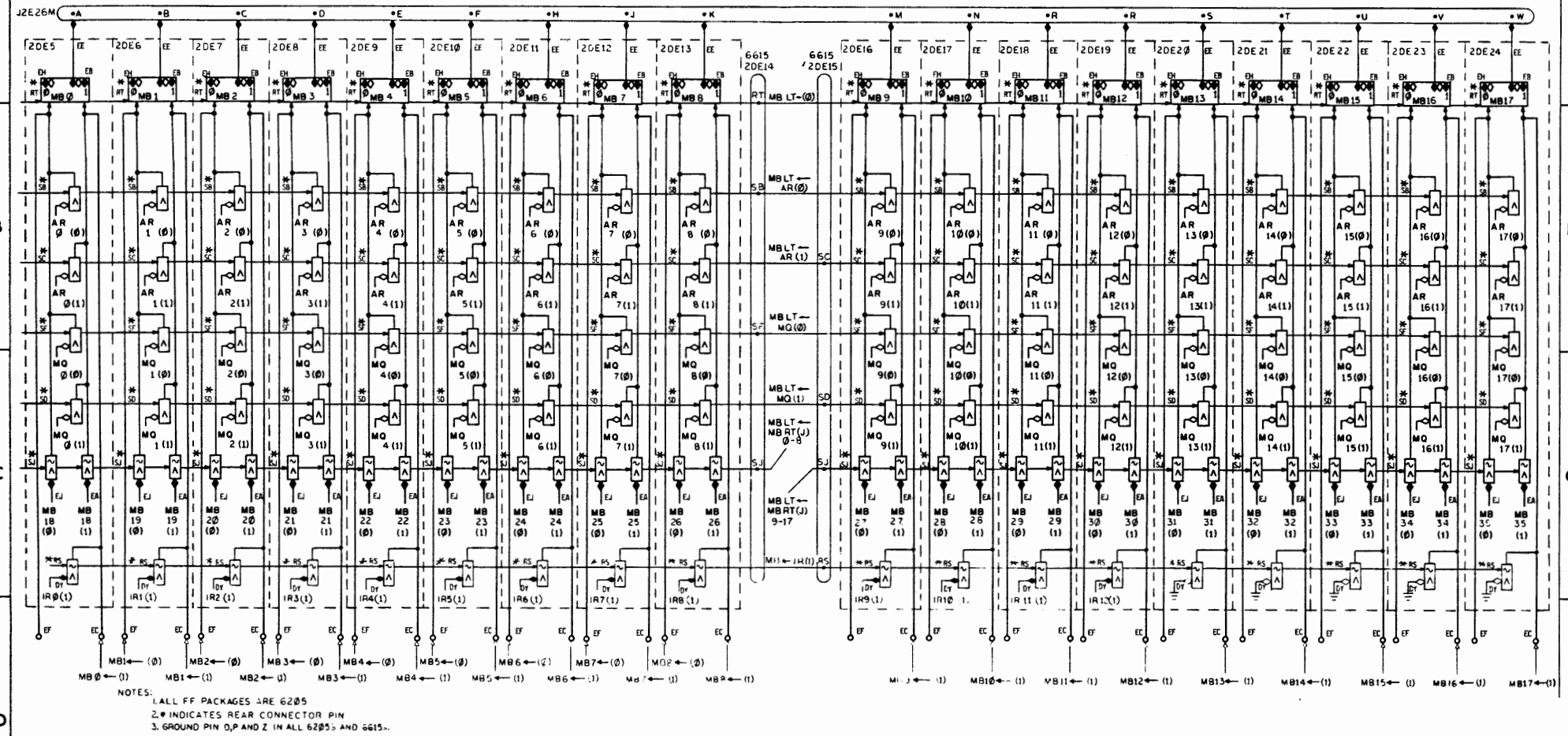
PC - Program Counter (BS-D-166-0-PC-2)



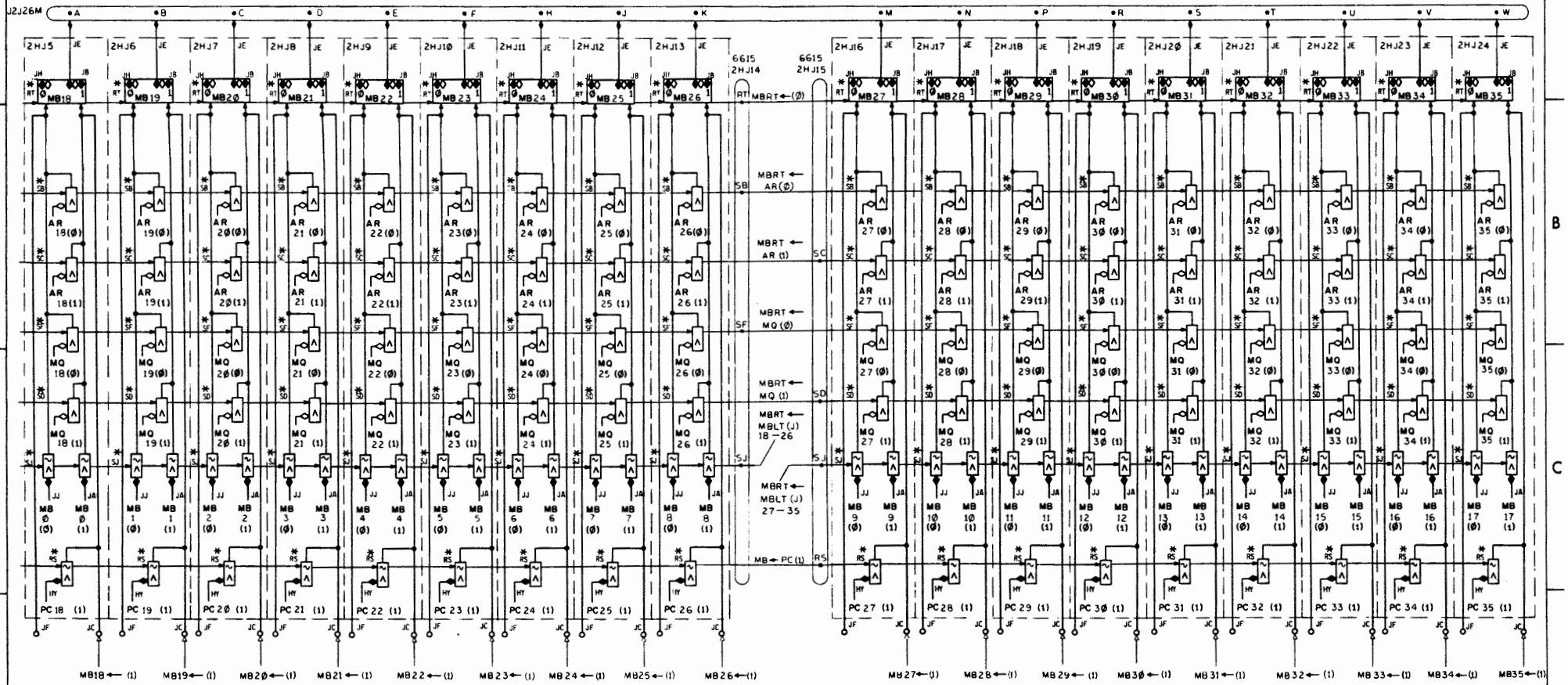
PC Control (BS-D-166-0-PC-1)



EX - Executive Mode (BS-D-166-0-EX)



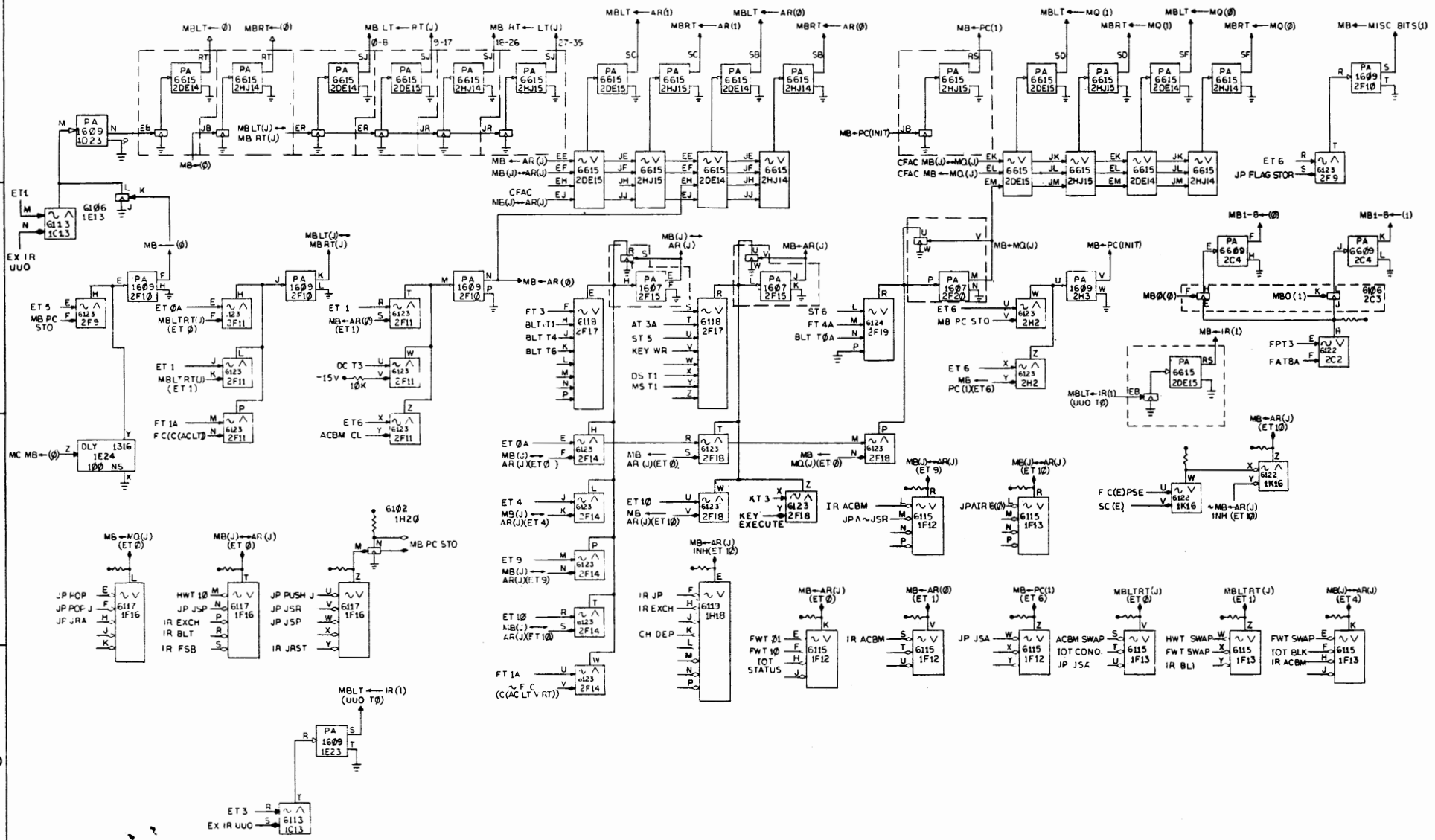
MBLT Register 0-17 (BS-D-166-0-MB-2)



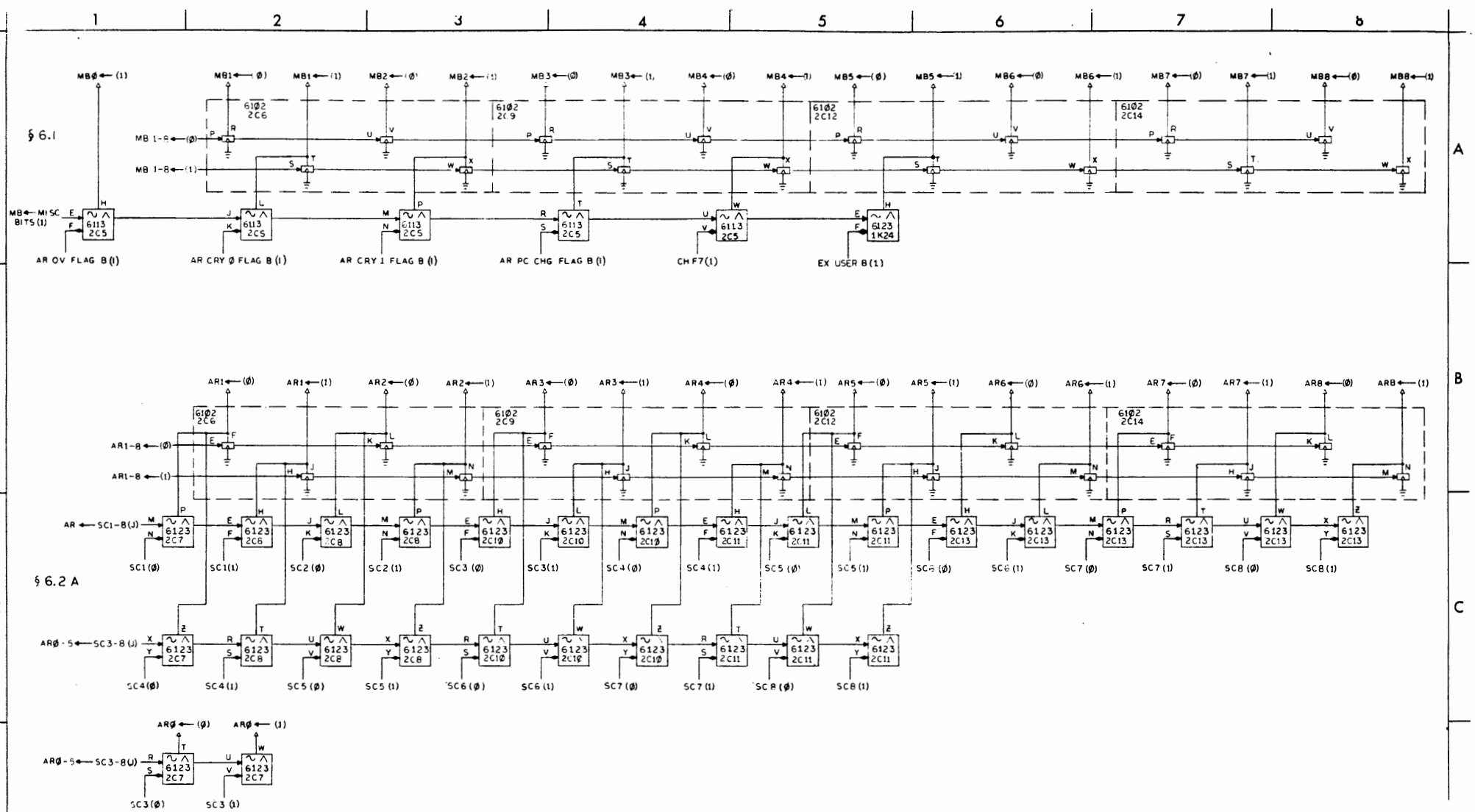
NOTES
1. ALL FF PACKAGES ARE 62Ø5
2. * INDICATES REAR CONNECTOR PIN
3. GROUND PIN D,P AND Z IN ALL 62Ø5's AND 6615's.

MBRT Register 18-35 (BS-D-166-0-MB-3)

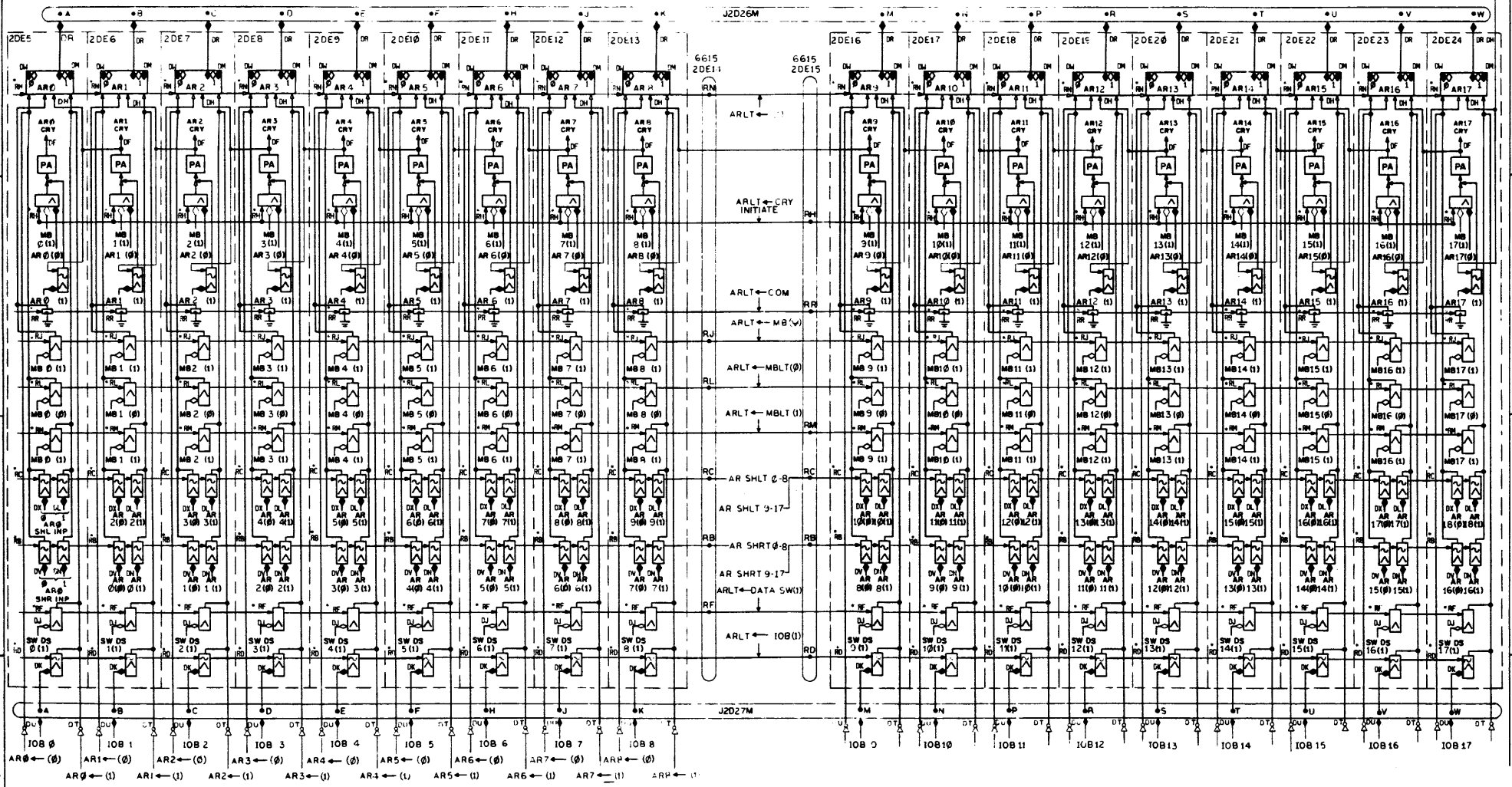
D-166-Ø-MB-1 REV LTR J



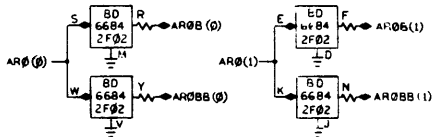
MB - Memory Buffer Control
(BS-D-166-0-MB-1)



MB, AR Special Input (BS-D-166-0-MBAR)

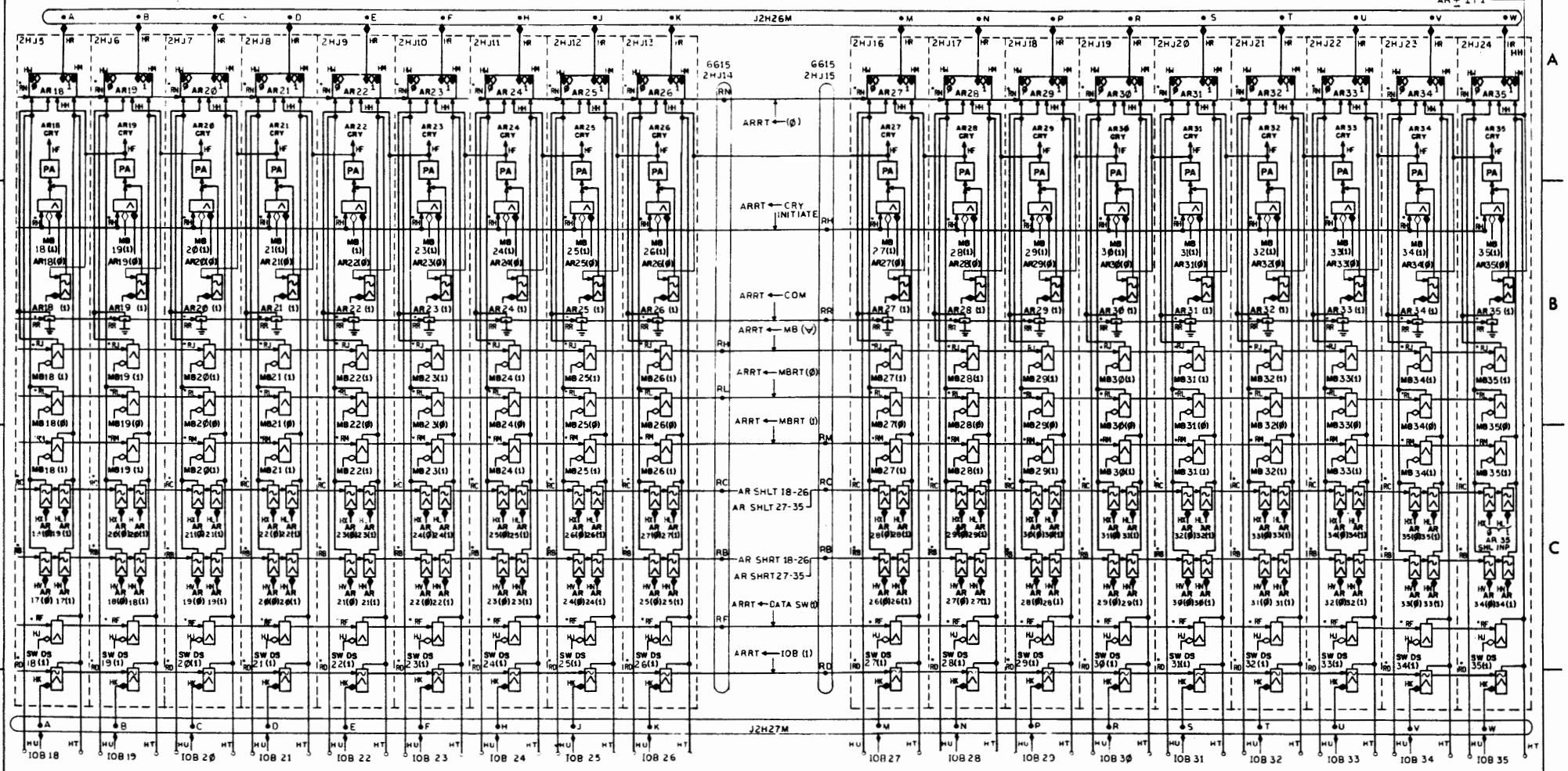


NOTE
1. * INDICATES FRONT CONNECTOR PIN
2. ALL FF PACKAGES ARE 62Ø5
3. GROUND PINS D AND Z IN ALL 62Ø5'S AND 6615S.



ARLT Register 0-17 (BS-D-166-0-AR-4)

D-166-Ø-AR-5 REV LTR B

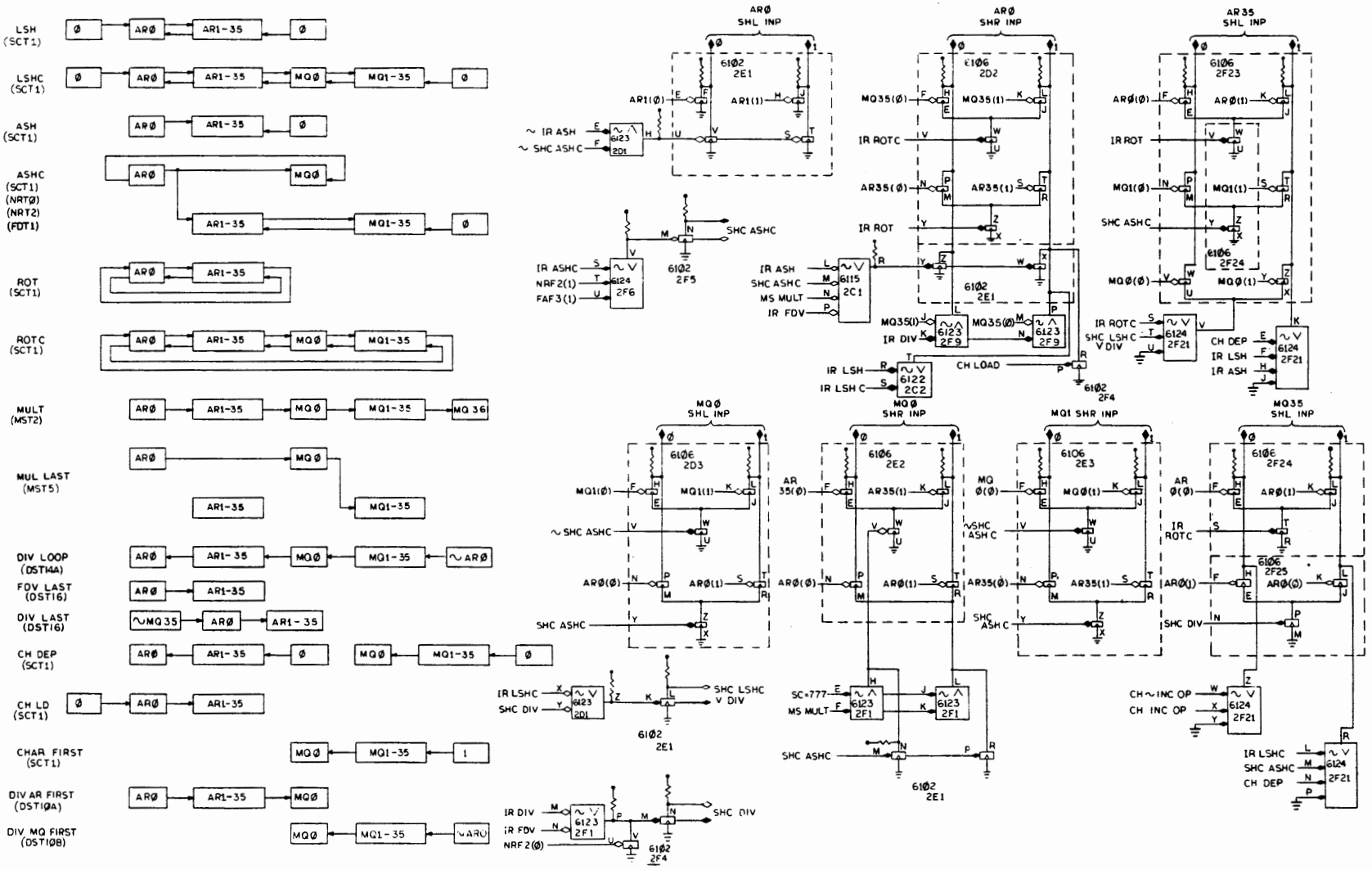


NOTES:

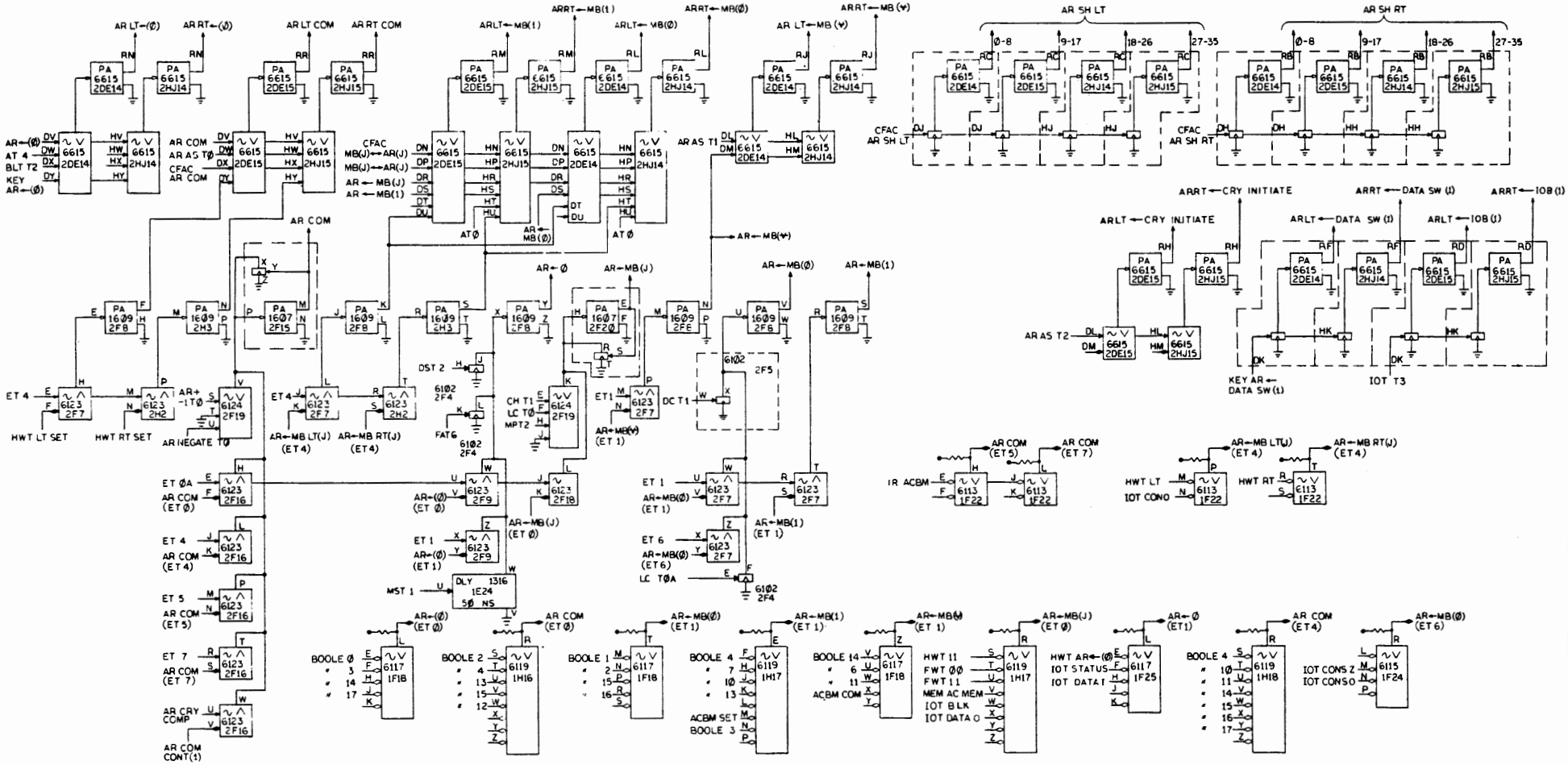
1. ALL FF PACKAGES ARE 62Ø5
2. * INDICATES REAR CONNECTOR PIN
3. GROUND PIN D AND Z IN ALL 62Ø5's AND 6615's.

ARRT Register 18-35 (BS-D-166-0-AR-5)

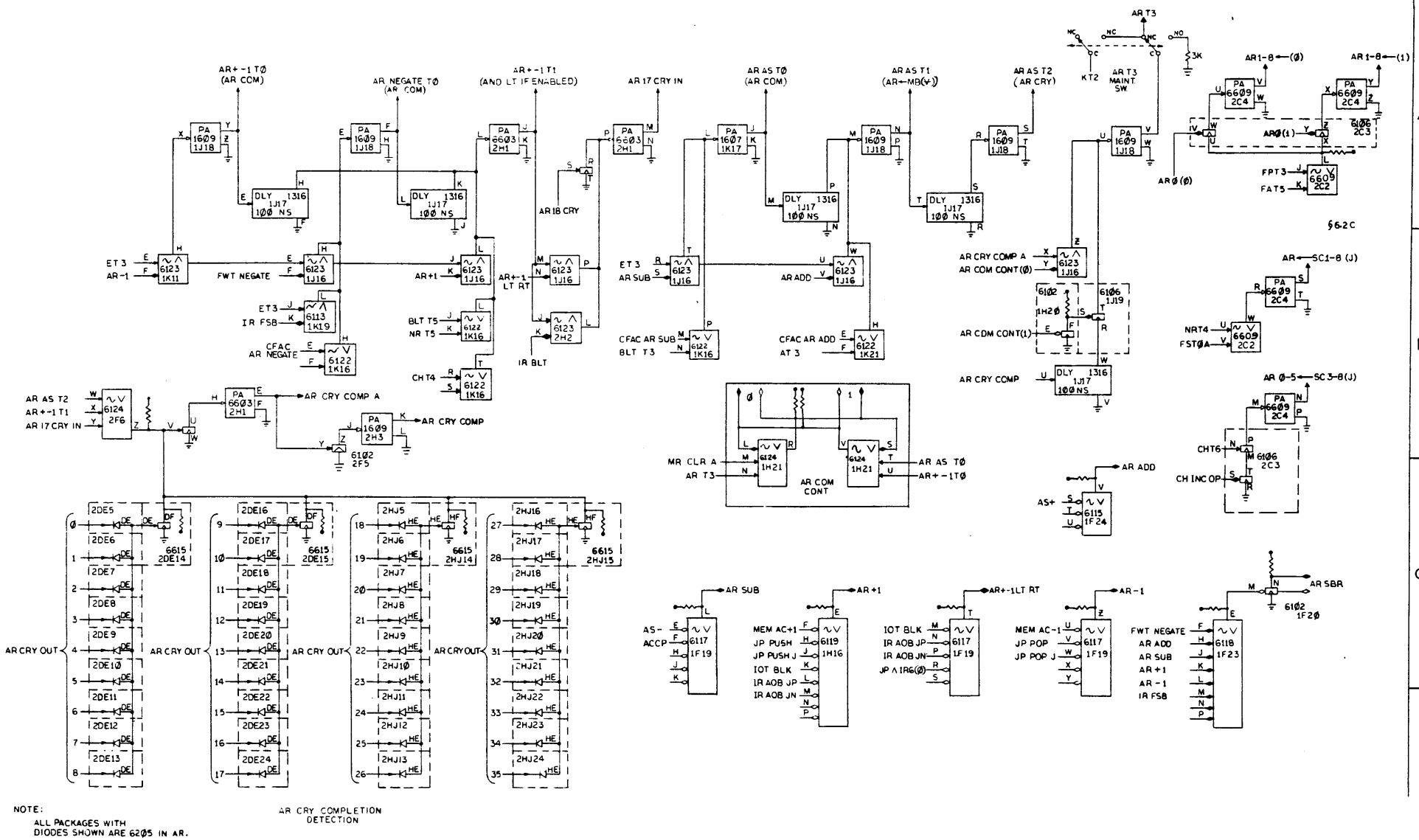
D-166-Ø-ARMQ REV LTR D



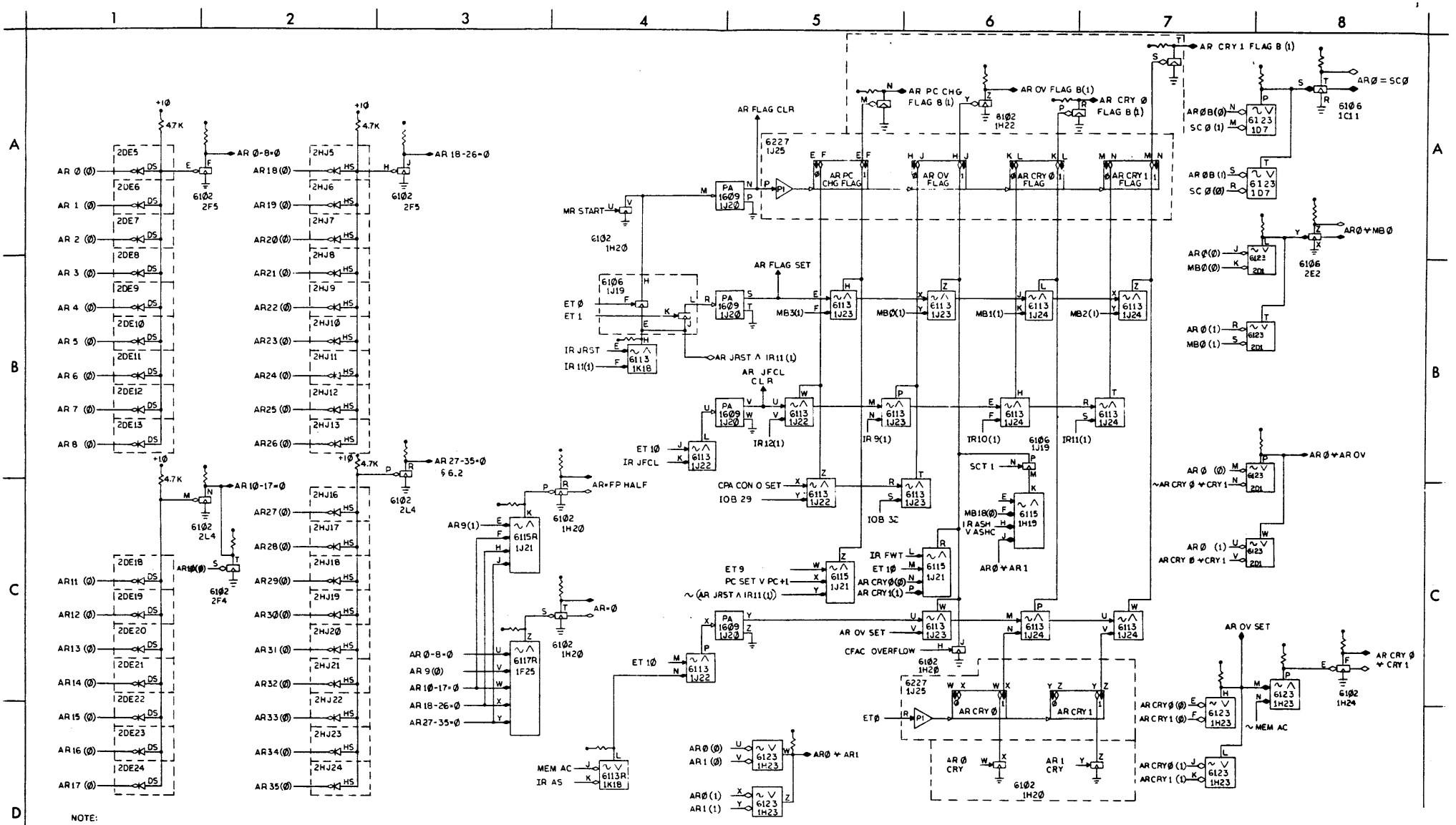
AR, MQ Shift Connections (BS-D-166-0-ARMQ)



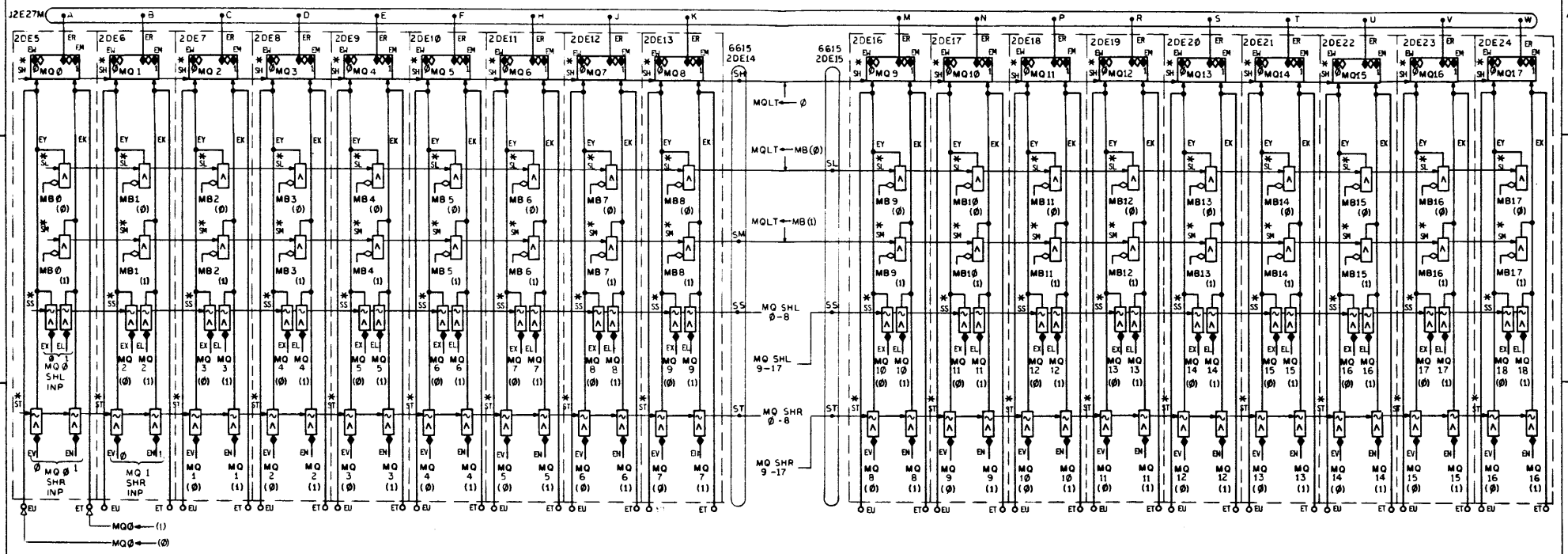
AR Control 1 (BS-D-166-0-AR-1)



AR Control 2 (BS-D-166-0-AR-2)

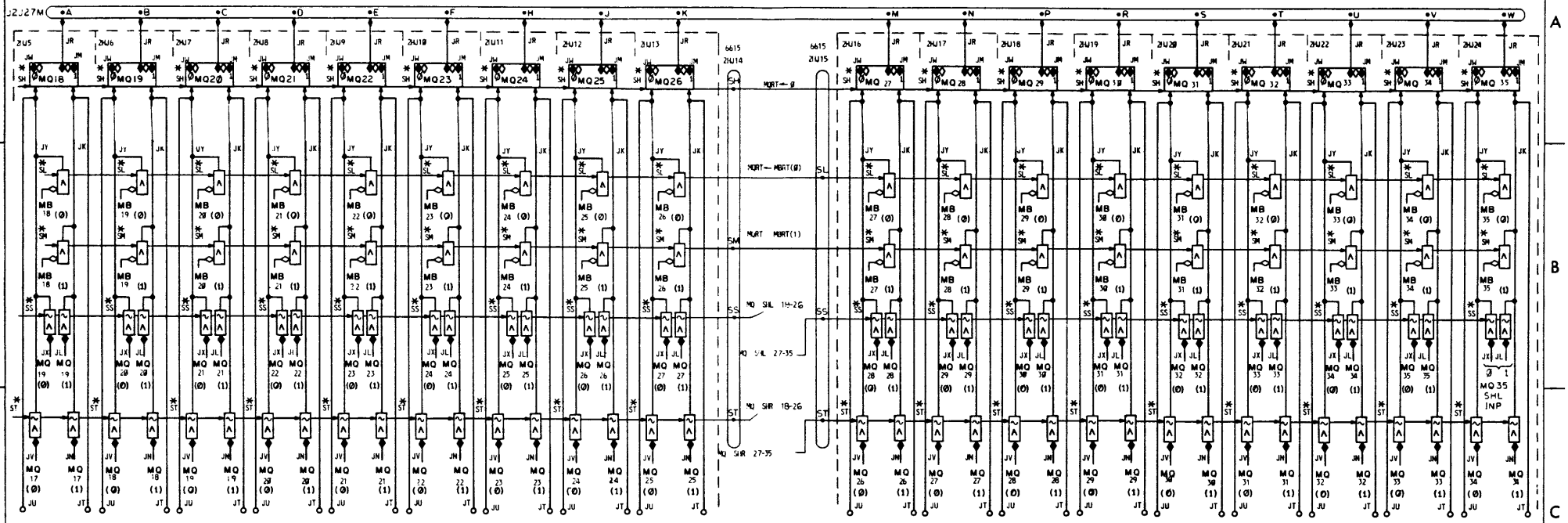


AR Control 3 (BS-D-166-0-AR-3)



NOTES:
1. ALL FF PACKAGES ARE 6205
2. * INDICATES REAR CONNECTOR PIN
3. GROUND PIN D,P AND Z IN ALL 6205'S AND 6615'S.

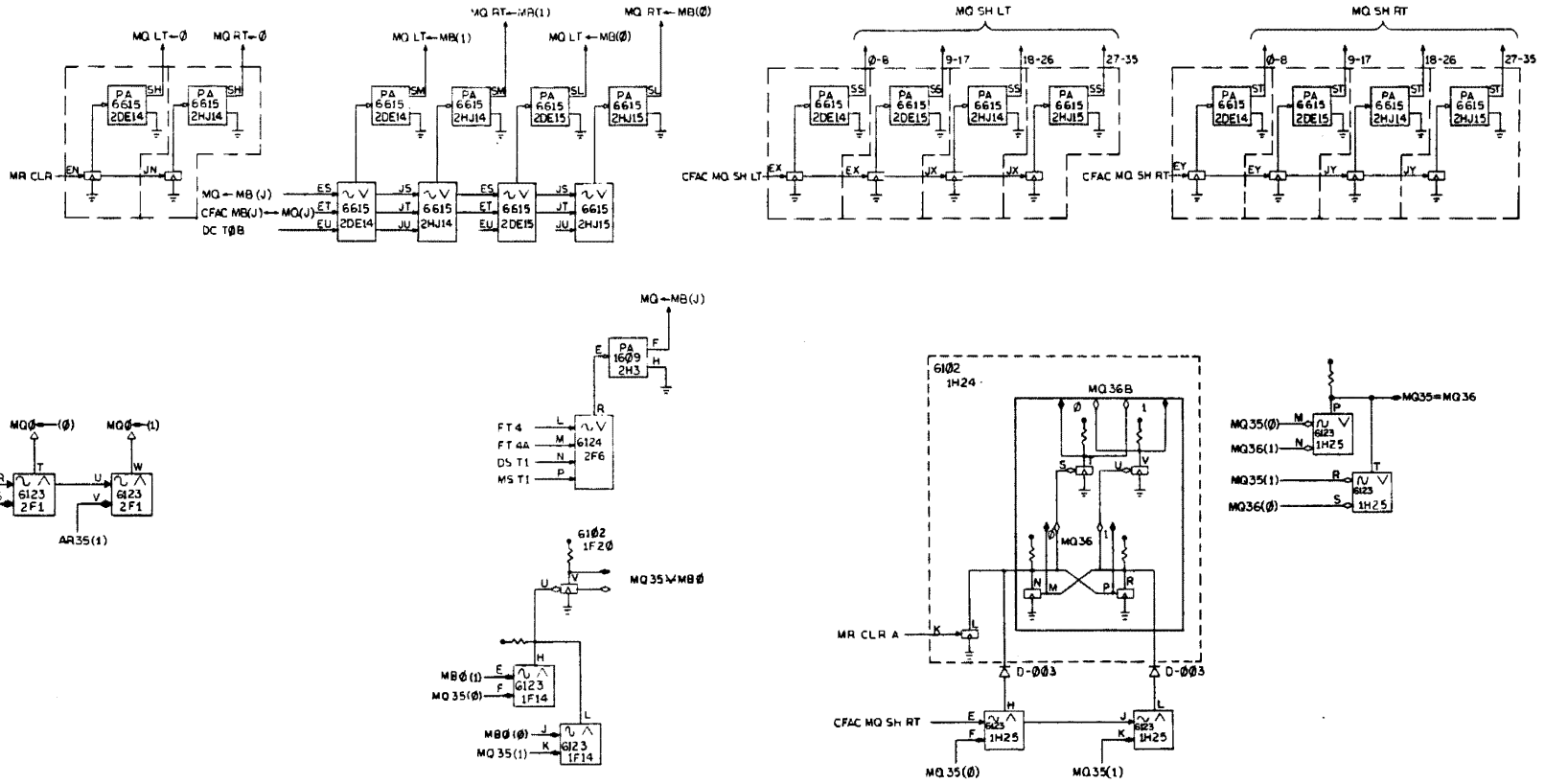
MQLT Register 0-17 (BS-D-166-0-MQ-2)



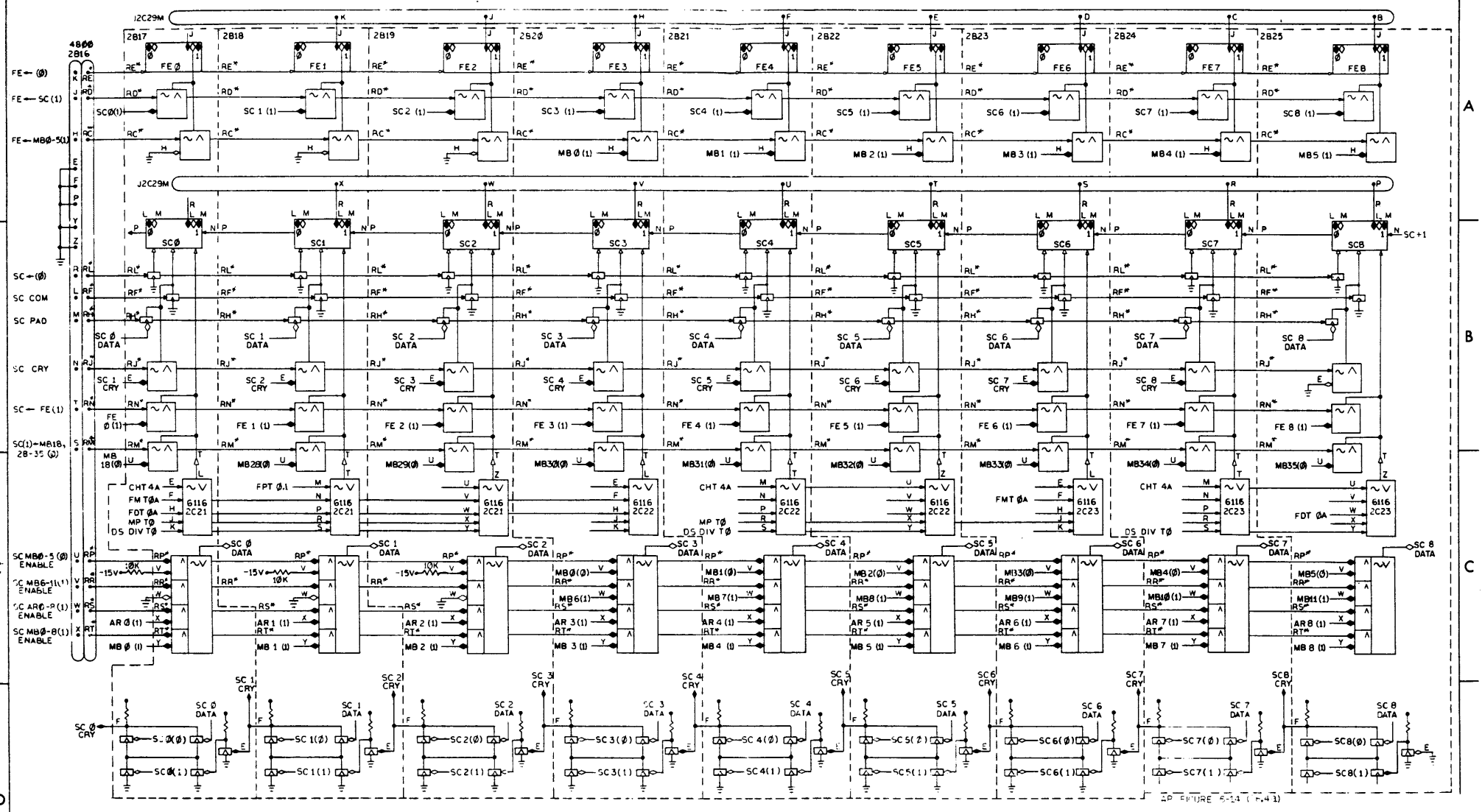
NOTE:

1. ALL FF PACKAGES ARE 6205.
2. * INDICATES REAR CONNECTOR PIN.
3. GROUND PIN D,P AND Z IN ALL 6205s AND 6615s.

MQRT Register 18-35 (BS-D-166-0-MQ-3)



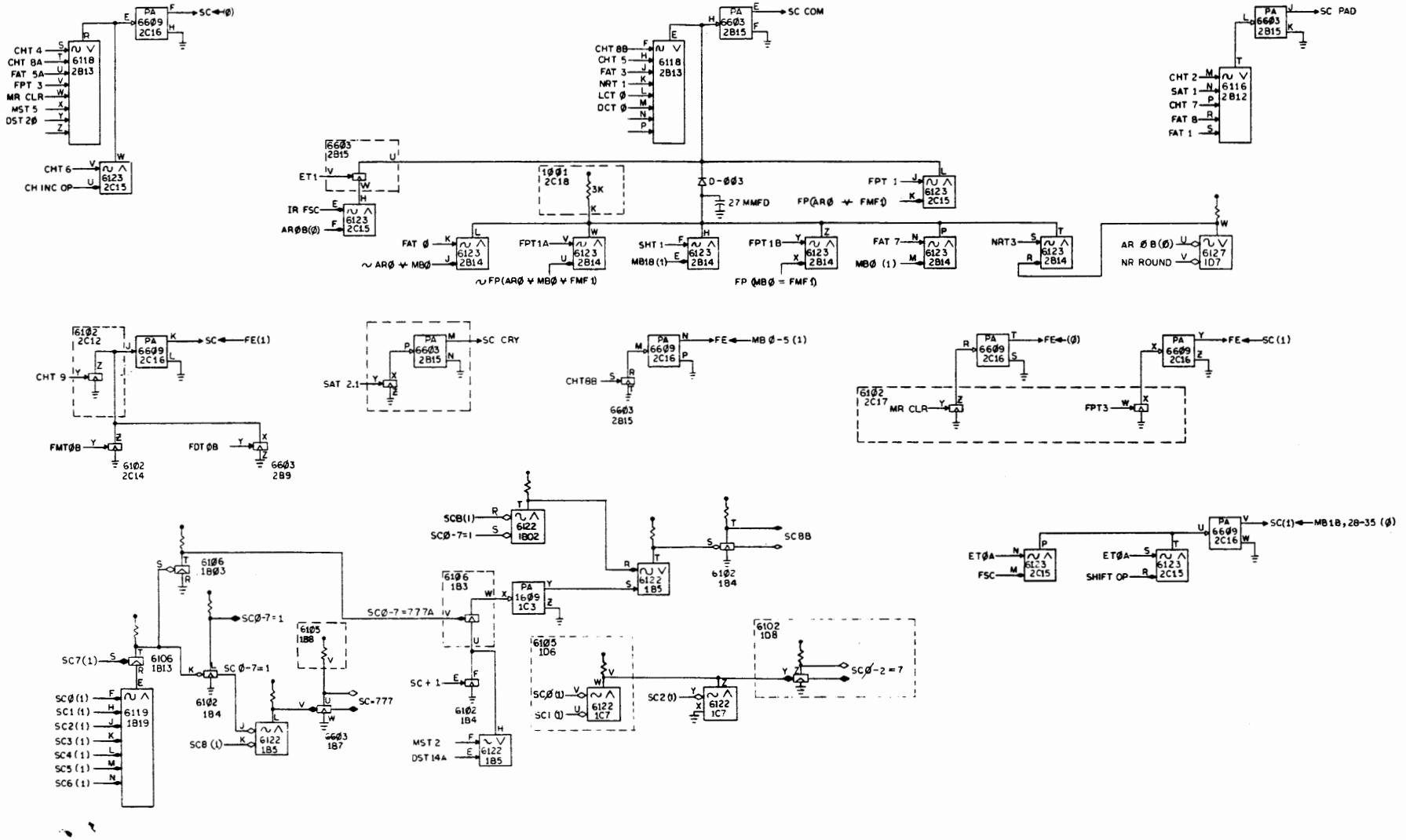
MQ - Multiplier Quotient Control
(BS-D-166-0-MQ-1)



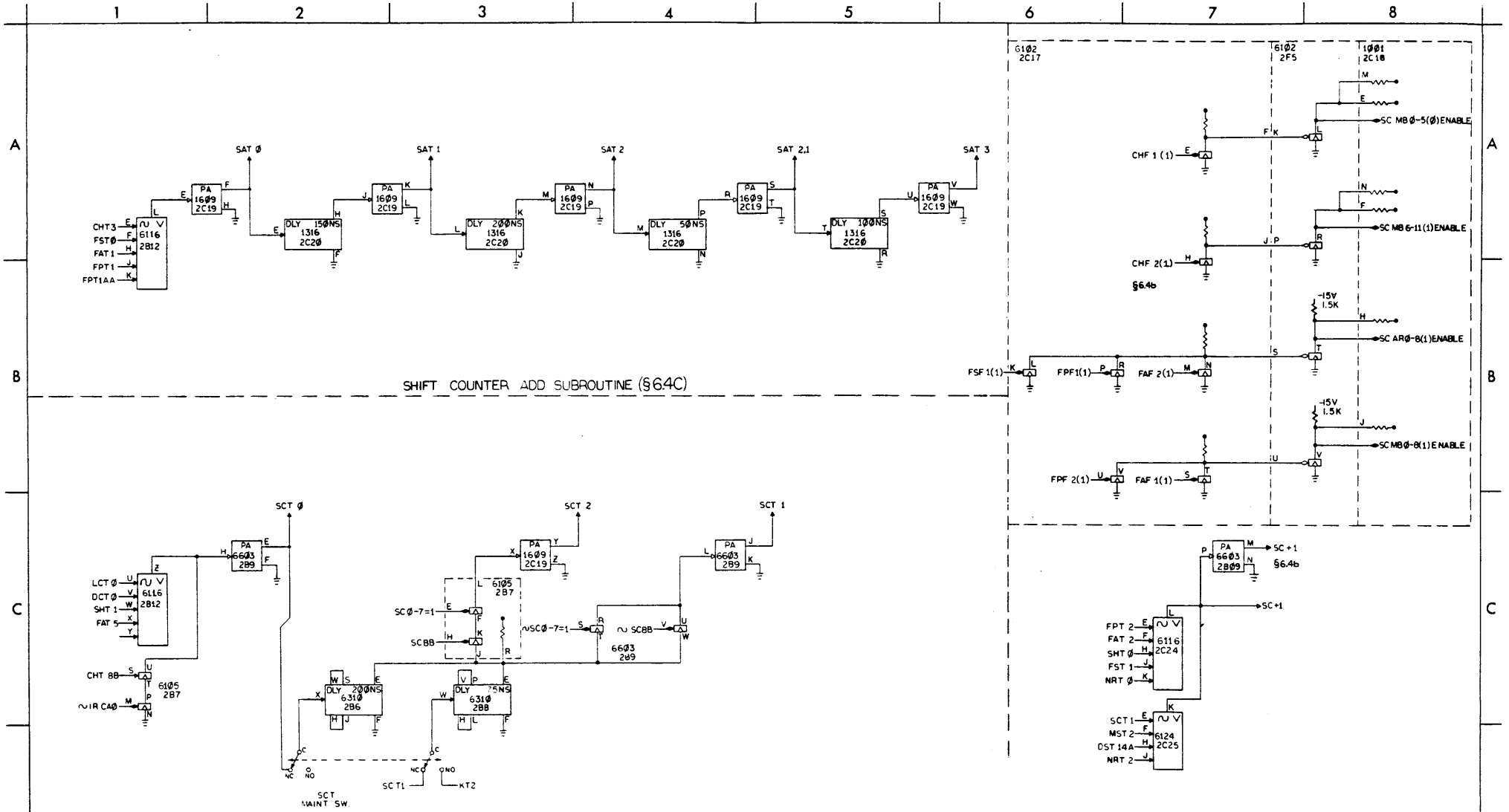
NOTE:

1. UNLESS OTHERWISE INDICATED ALL PACKAGES ARE 6203s.
2. * INDICATES REAR CONNECTOR PIN.

SC, FE - Shift Counter, Floating Exponent
(BS-D-166-0-SCFE)

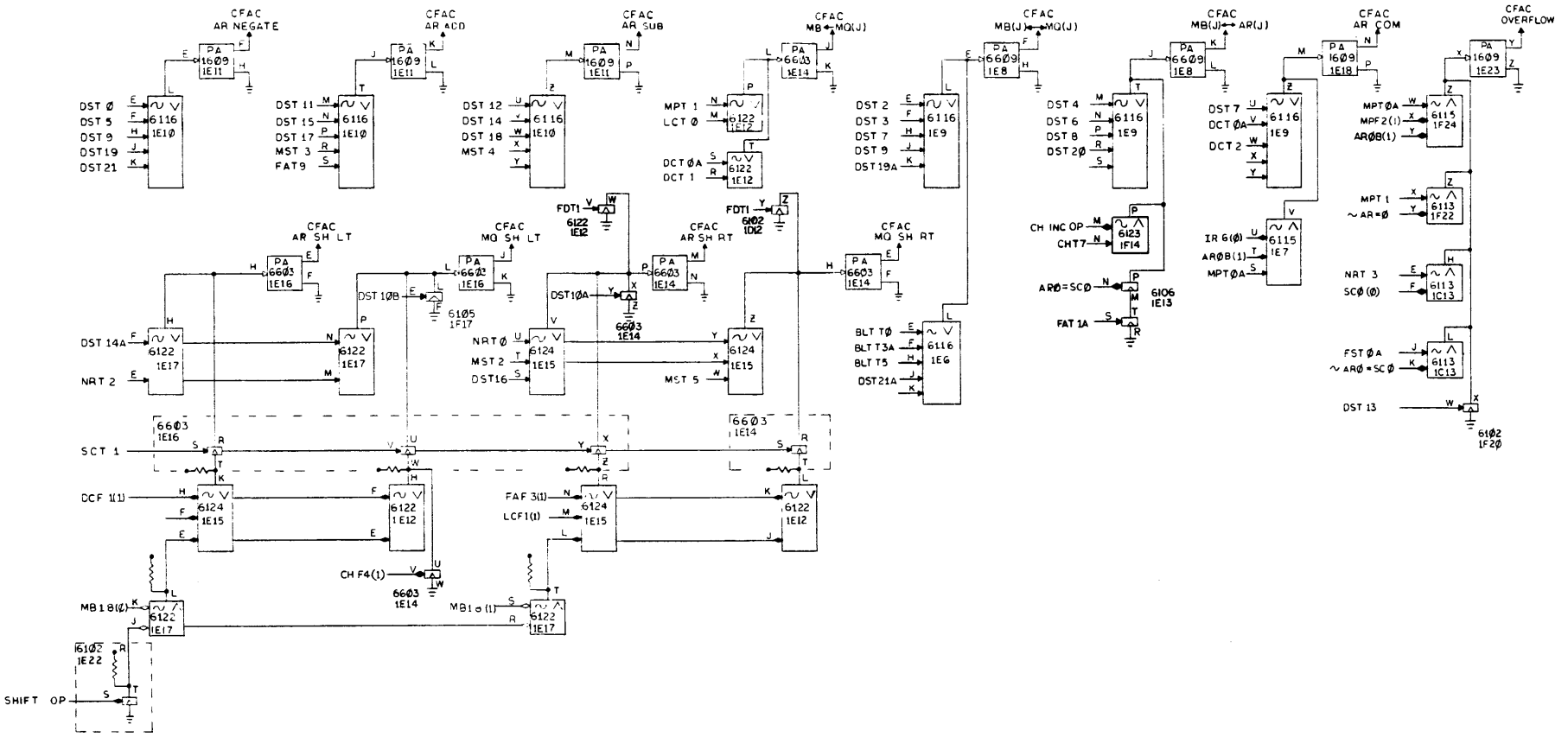


SC Control (BS-D-166-0-SC-1)

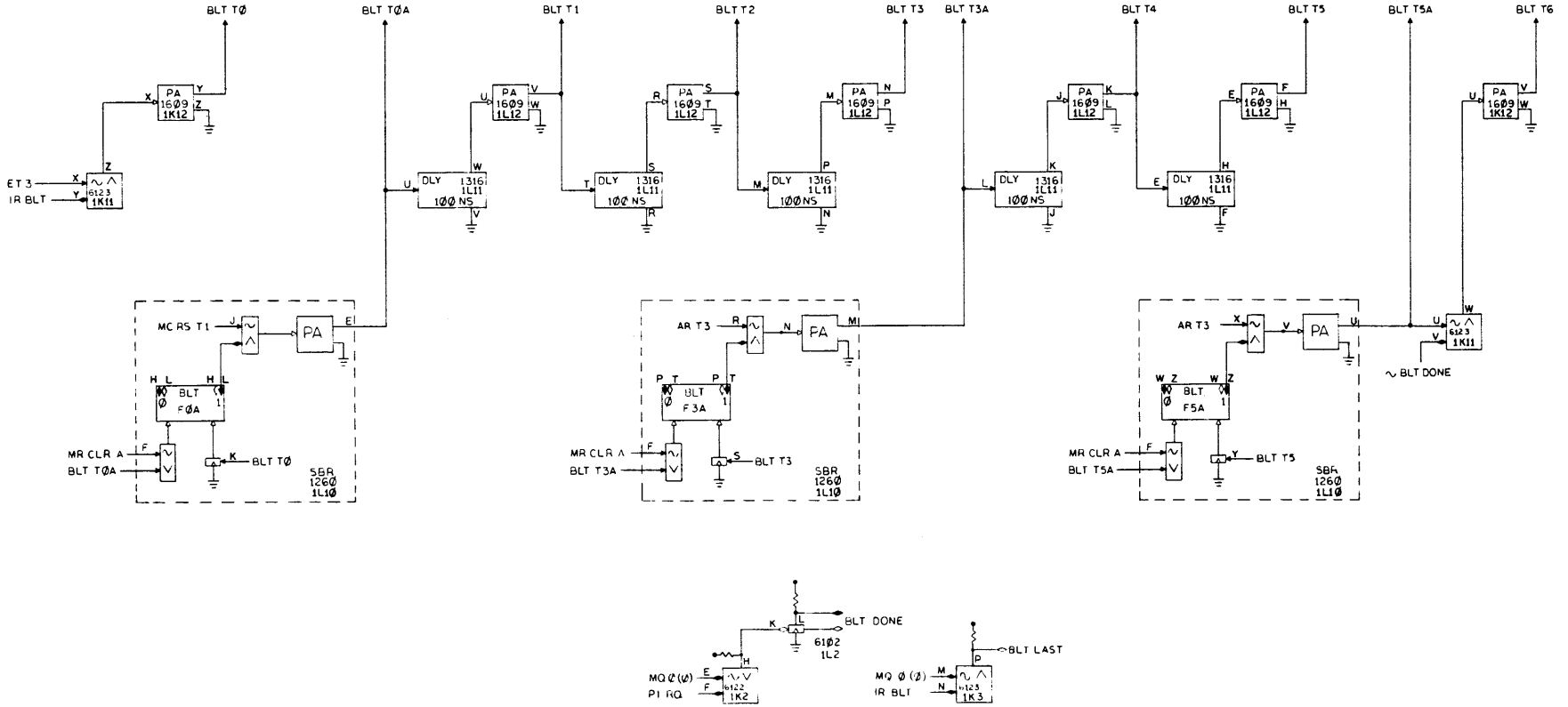


SHIFT AND COUNT SUBROUTINE(§6.4C)

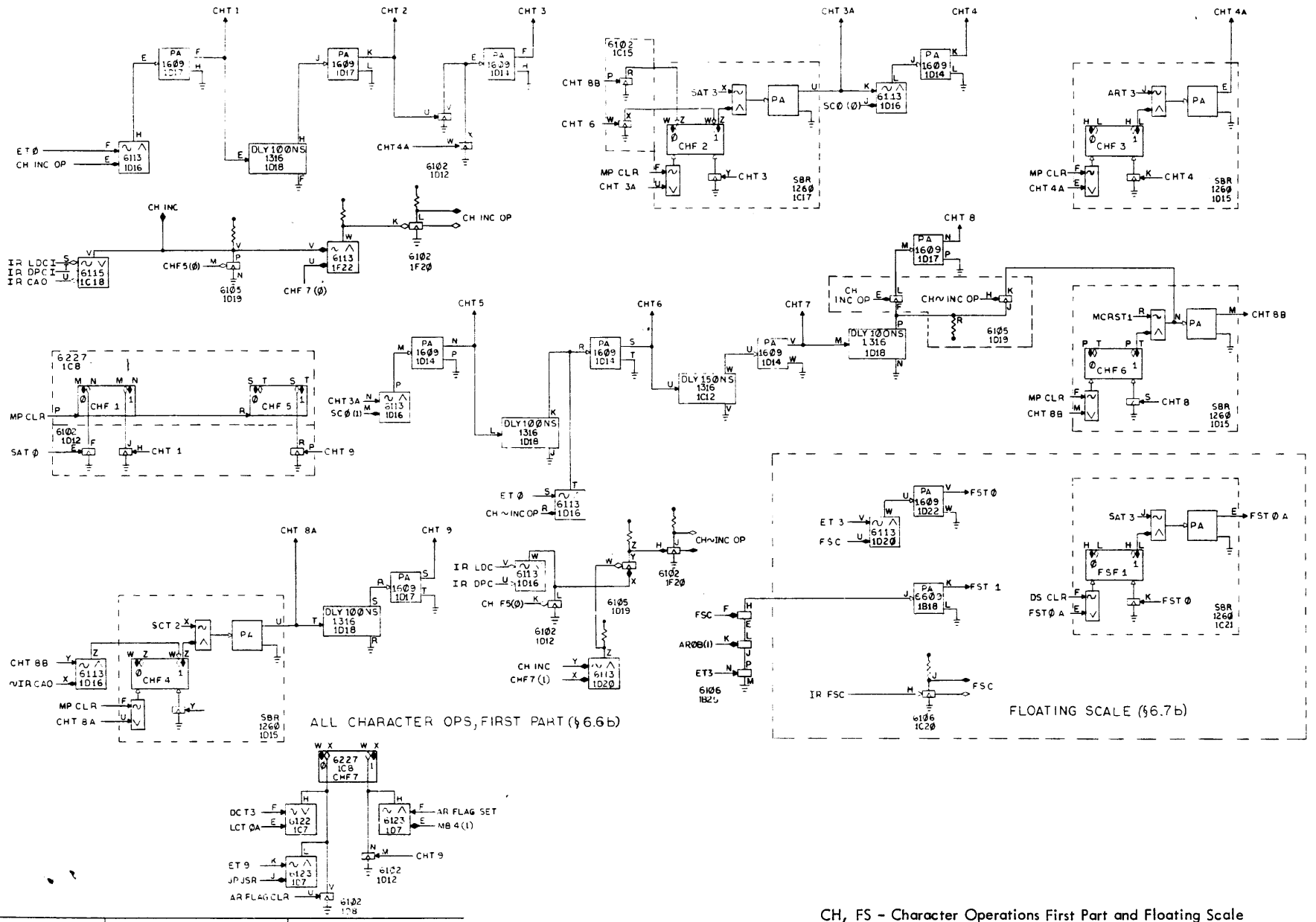
SC Control (BS-D-166-0-SC-2)



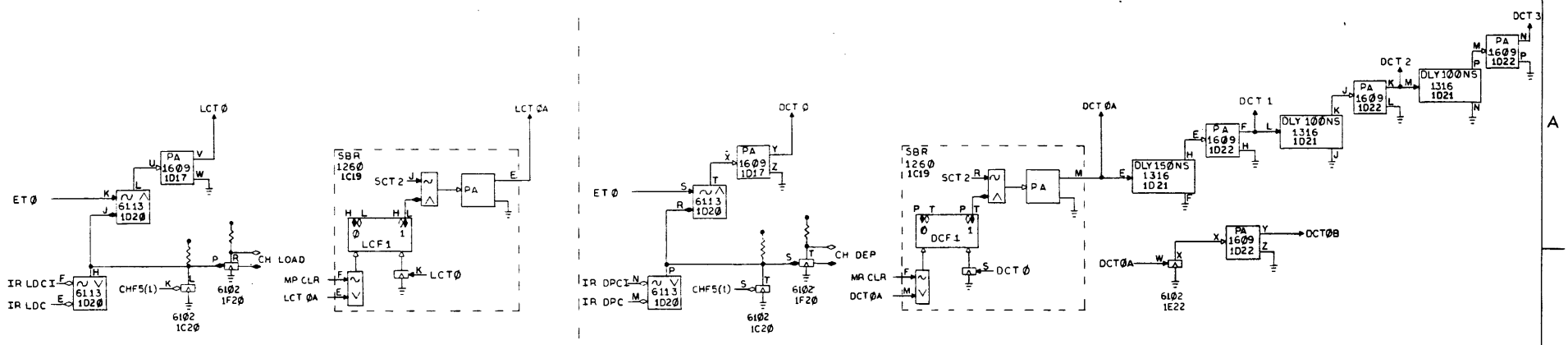
CFAC - Floating Arithmetic Connection
(BS-D-166-0-CFAC)



BLT - Block Transfer
(BS-D-166-0-BLT)

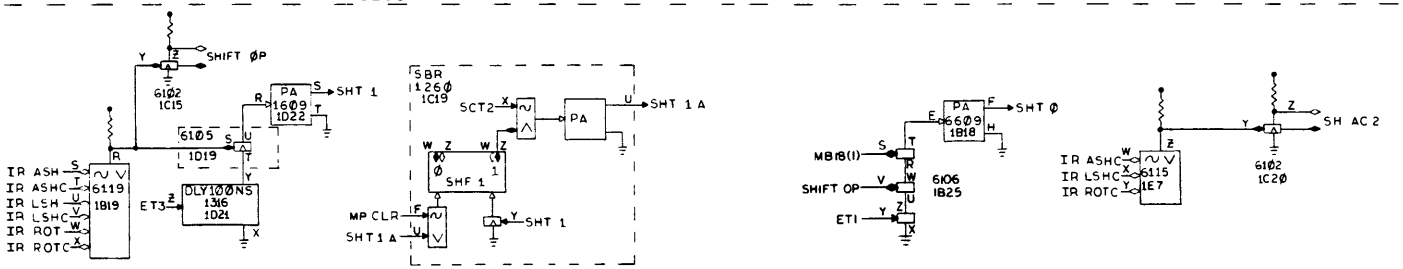


CH, FS - Character Operations First Part and Floating Scale
(BS-D-166-0-CHFS)



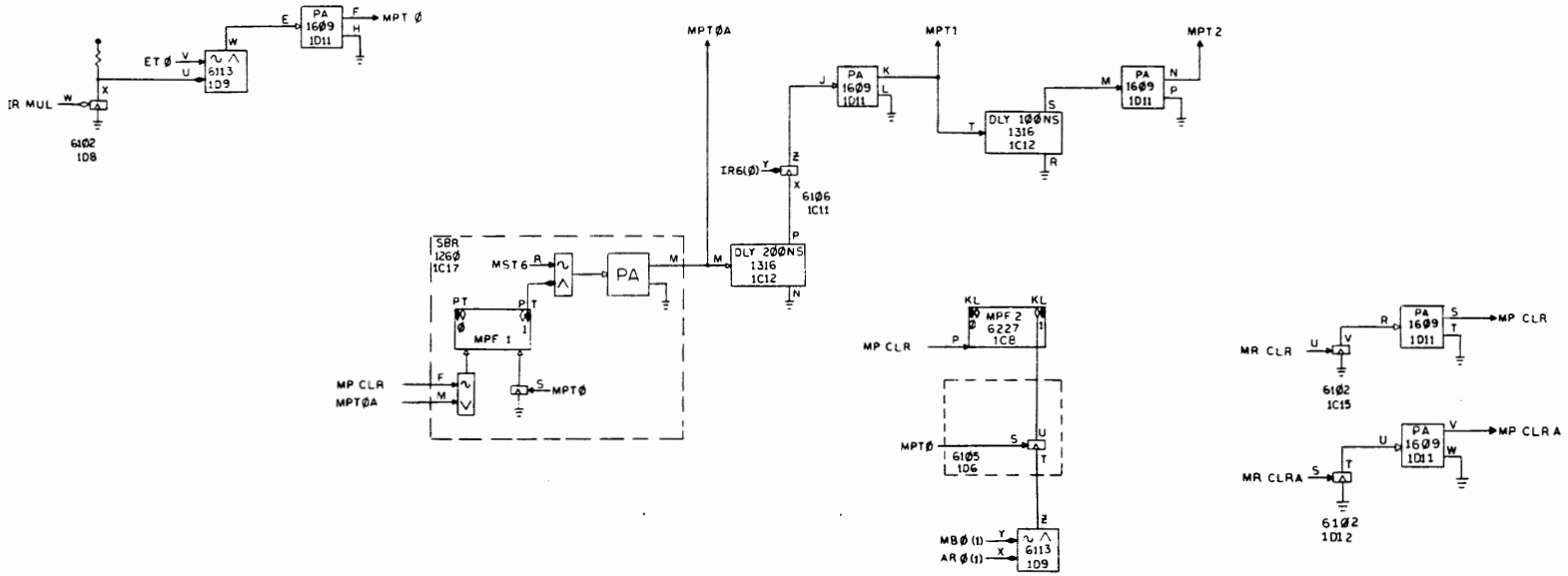
LOAD CHARACTER SECOND PART

DEPOSIT CHARACTER SECOND PART



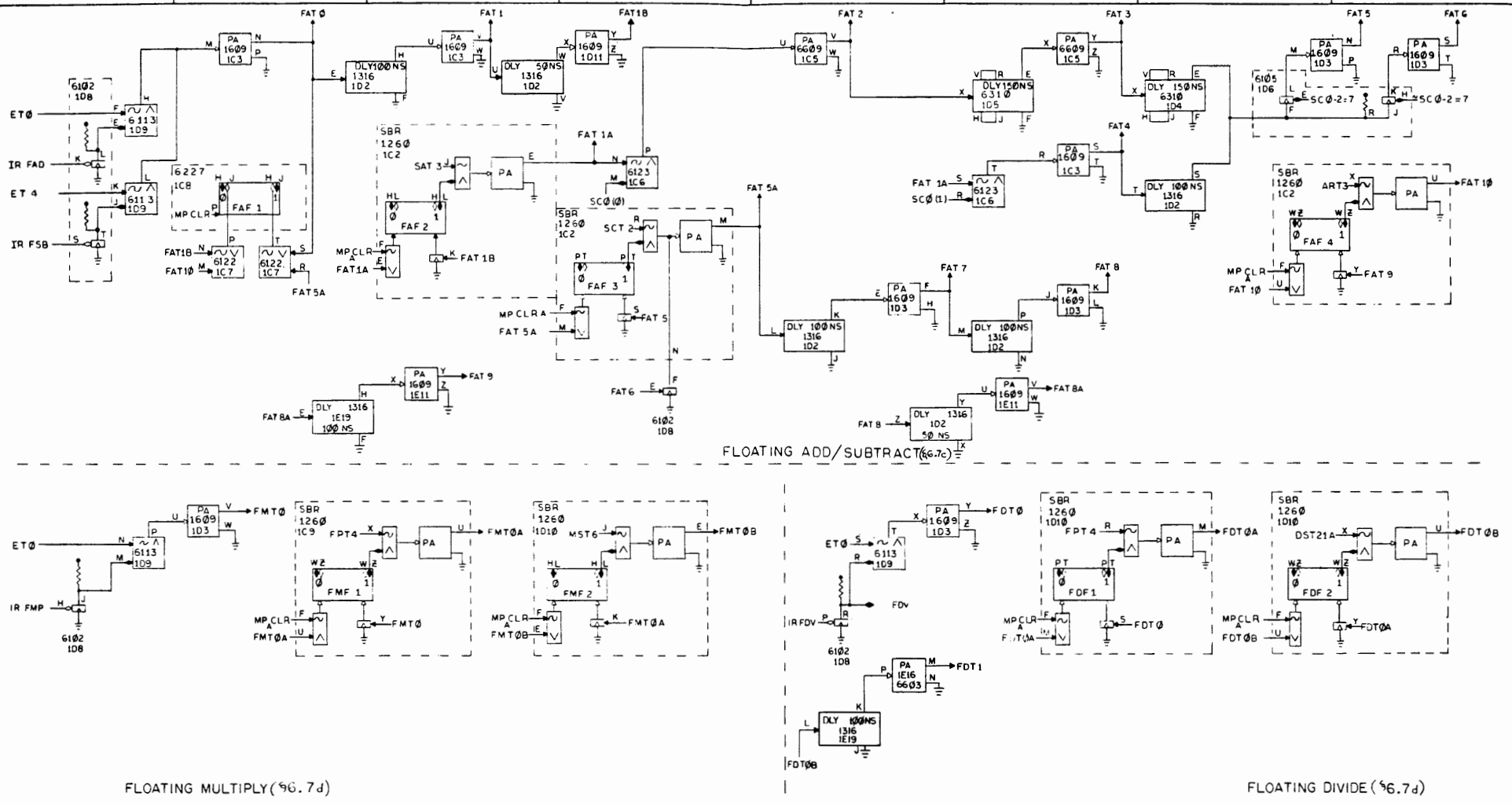
SHIFT OPERATIONS

LC, DC, SH - Character Operations Second Part
and Shift Operations
(BS-D-166-0-LDCS)

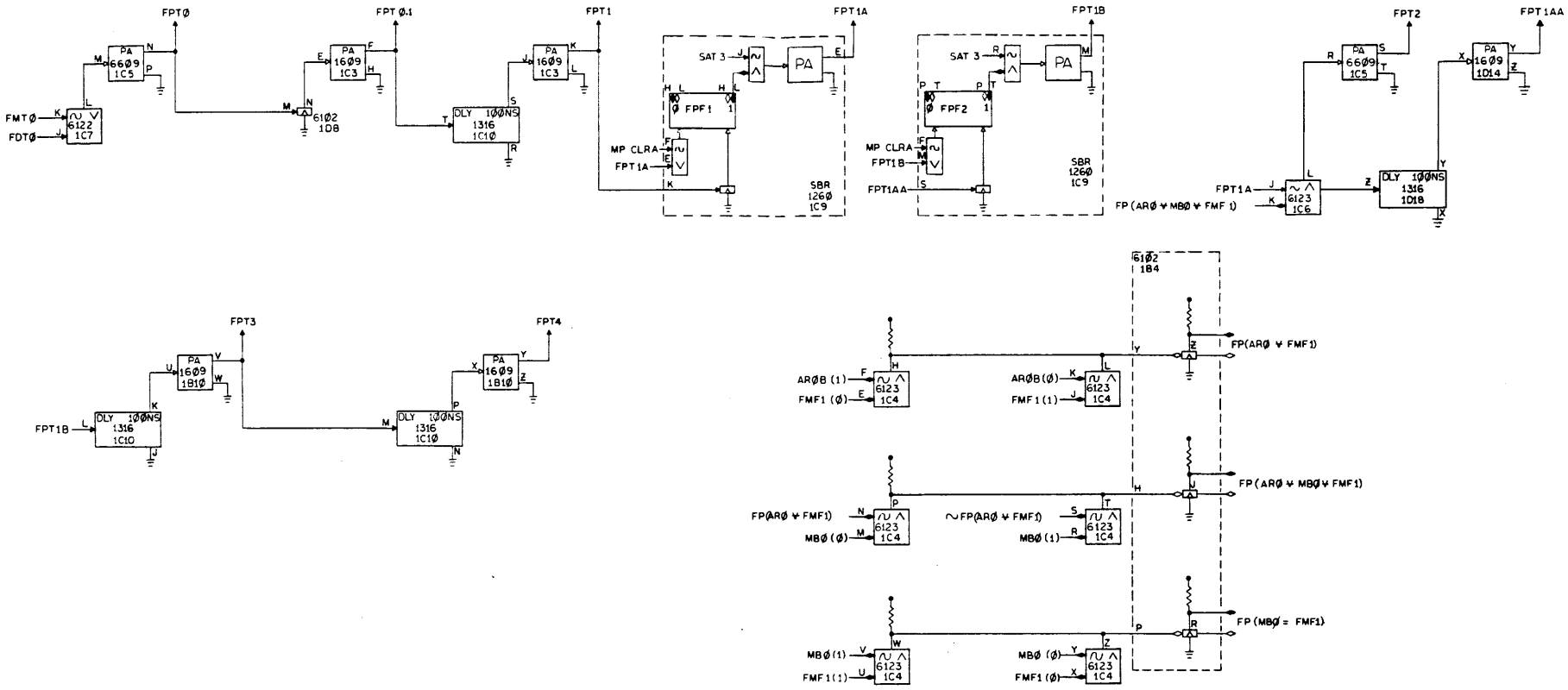


FIXED POINT MULTIPLY

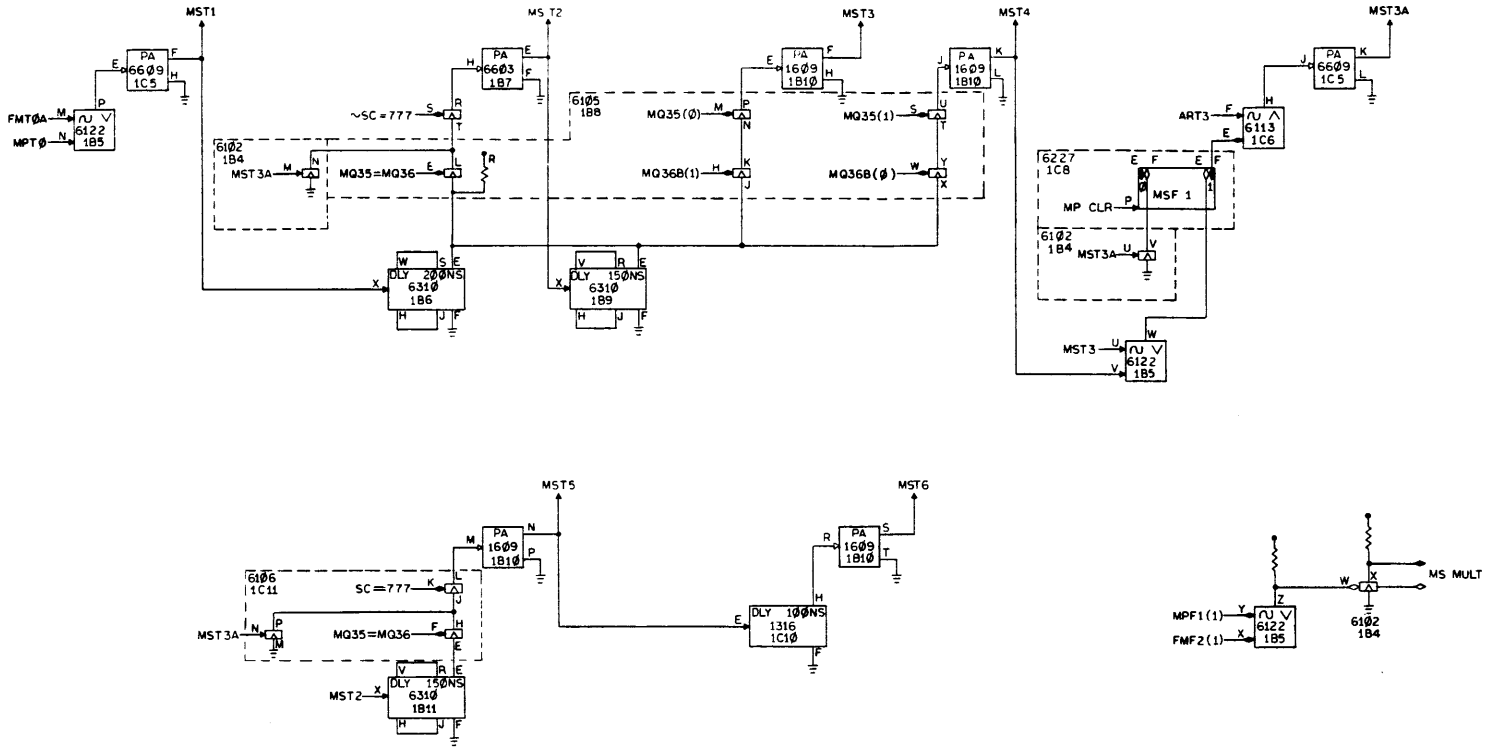
MP - Fixed Point Multiply
(BS-D-166-0-MP)



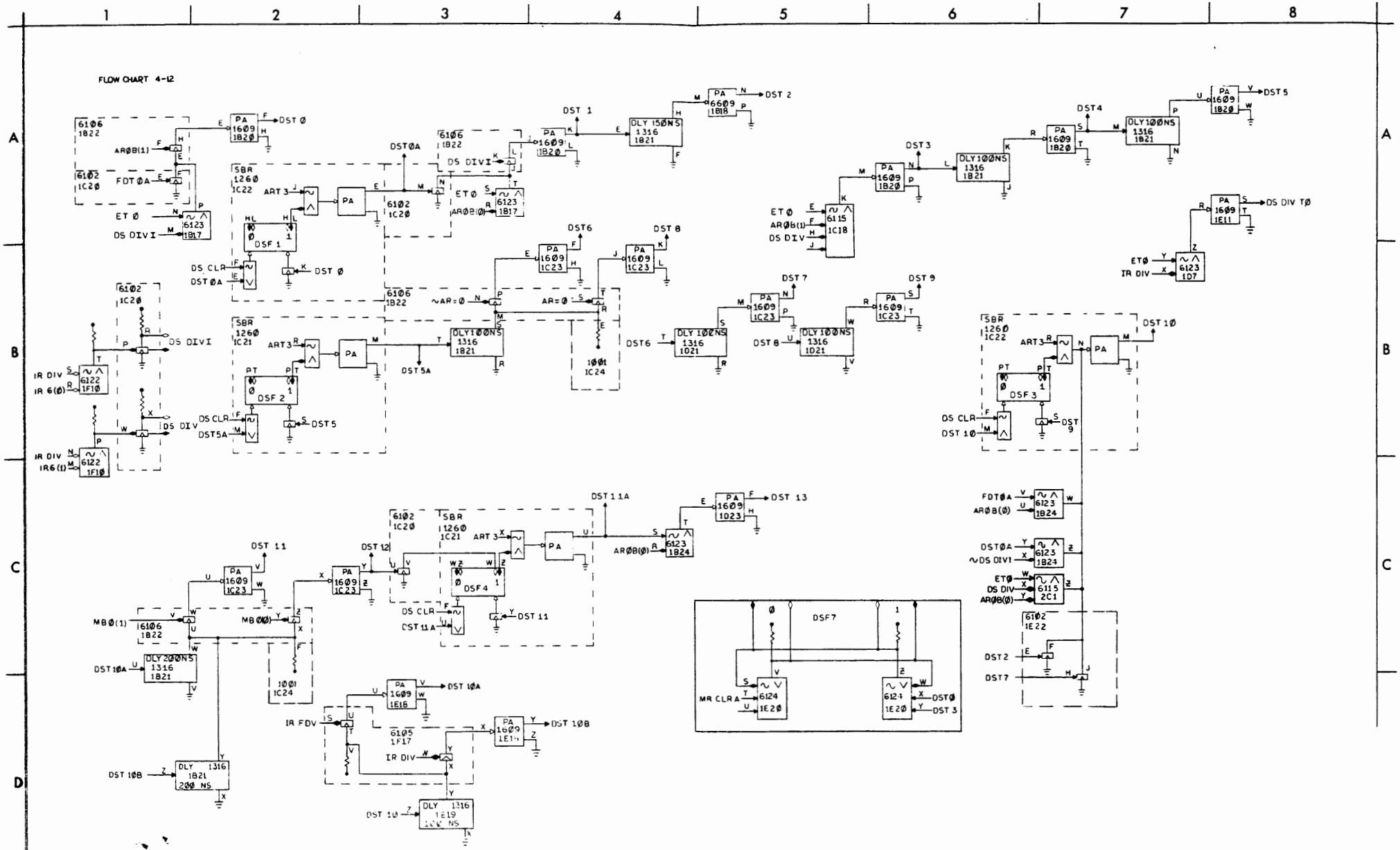
FA, FD, FM - Floating Point Instructions
(BS-D-166-0-FADM)

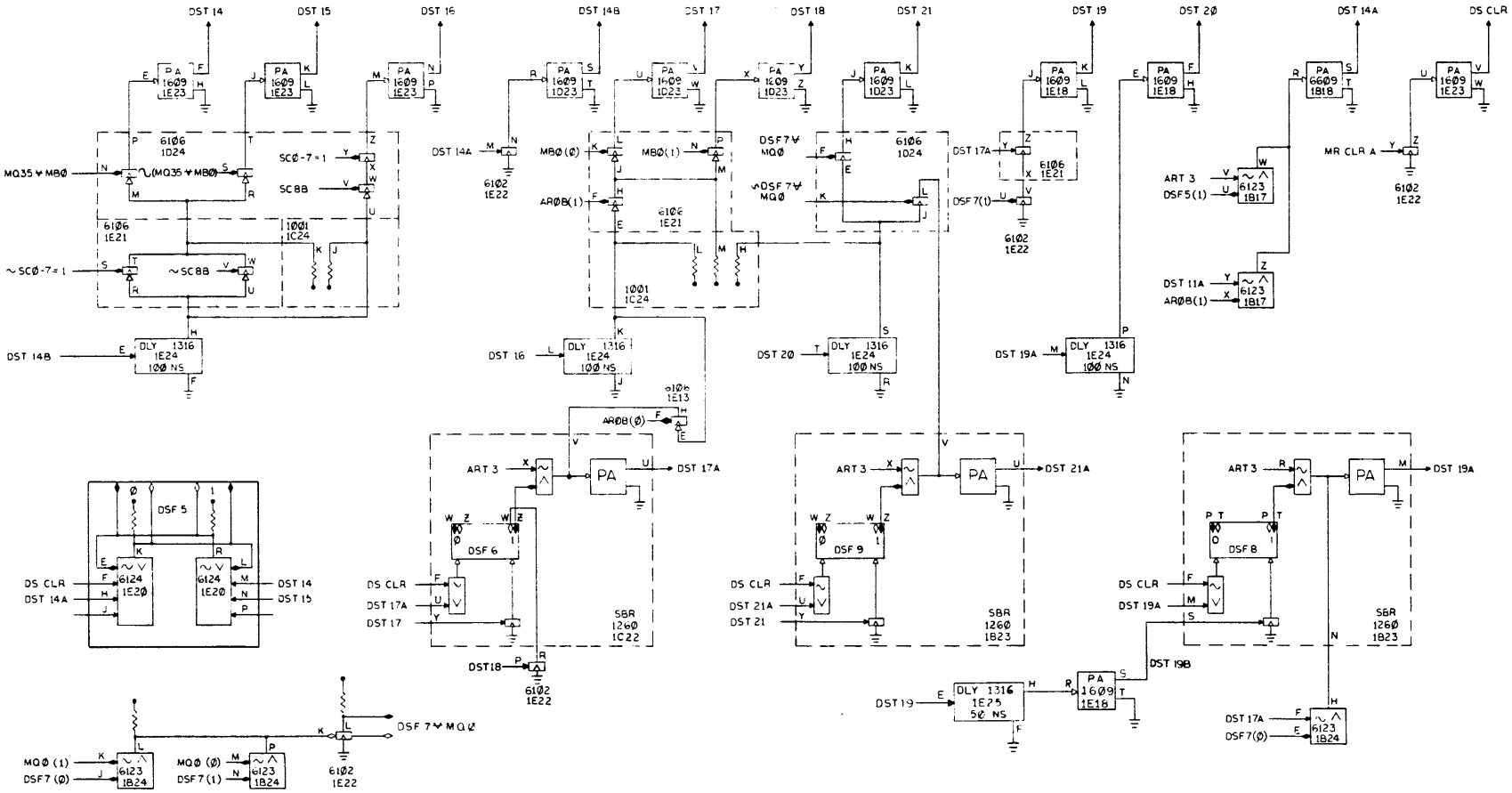


FP - Exponent Calculate Subroutine
(BS-D-166-0-FP)

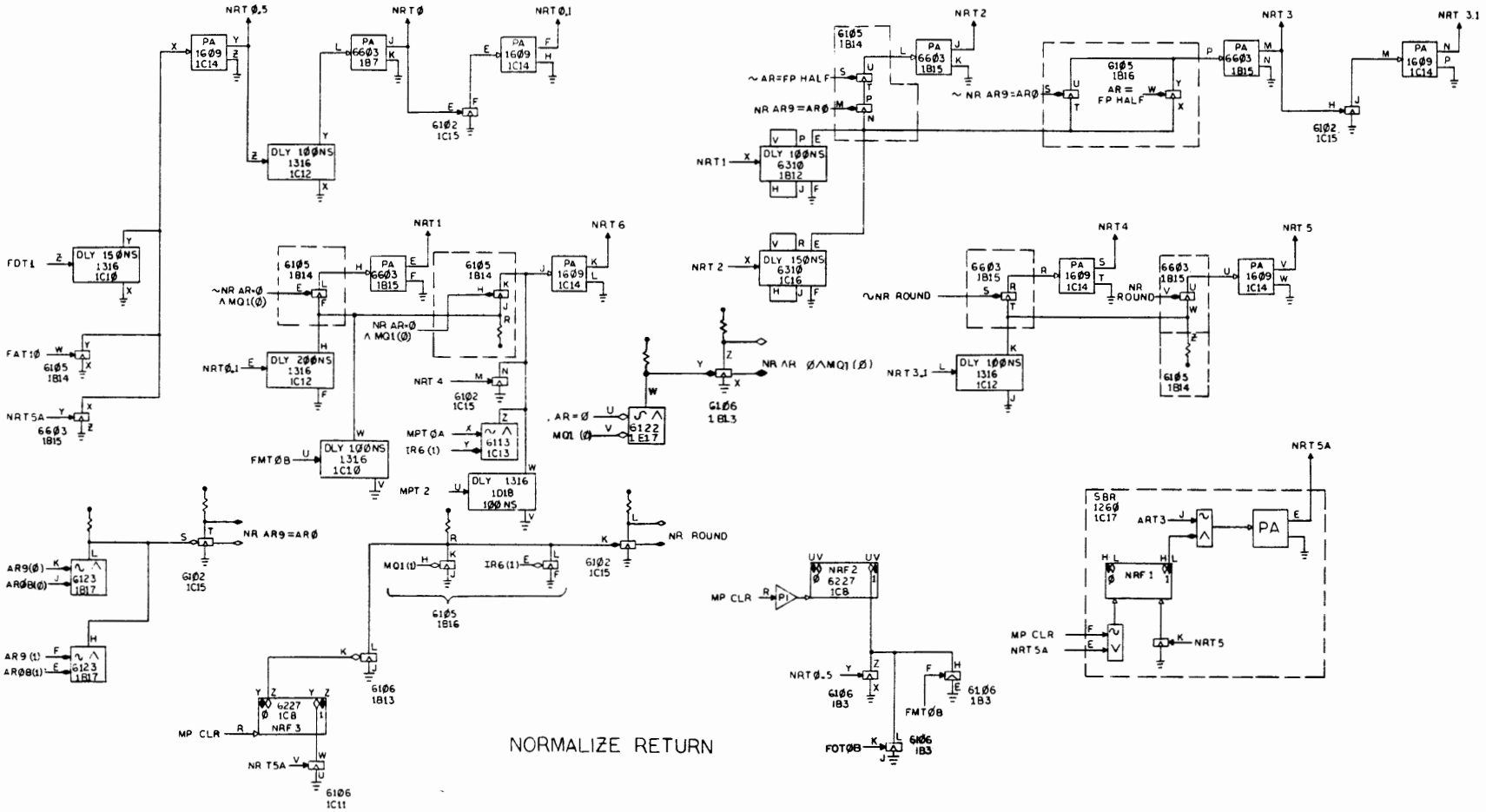


MS - Multiply Subroutine (BS-D-166-0-MS)

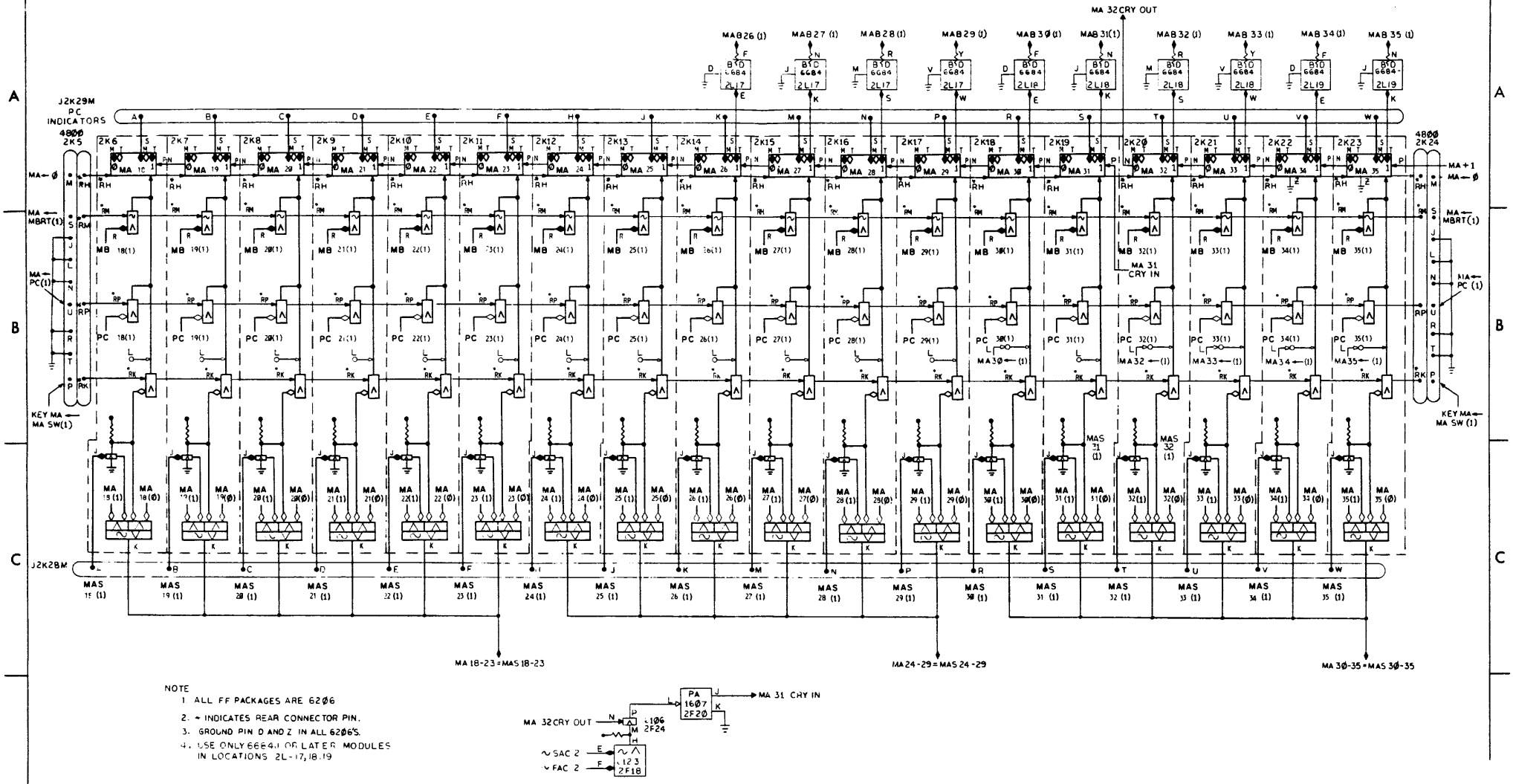

FLOW CHART 4-12
DS - Divide Subroutine 1 (BS-D-166-0-DS-1)



DS - Divide Subroutine 2 (BS-D-166-0-DS-2)



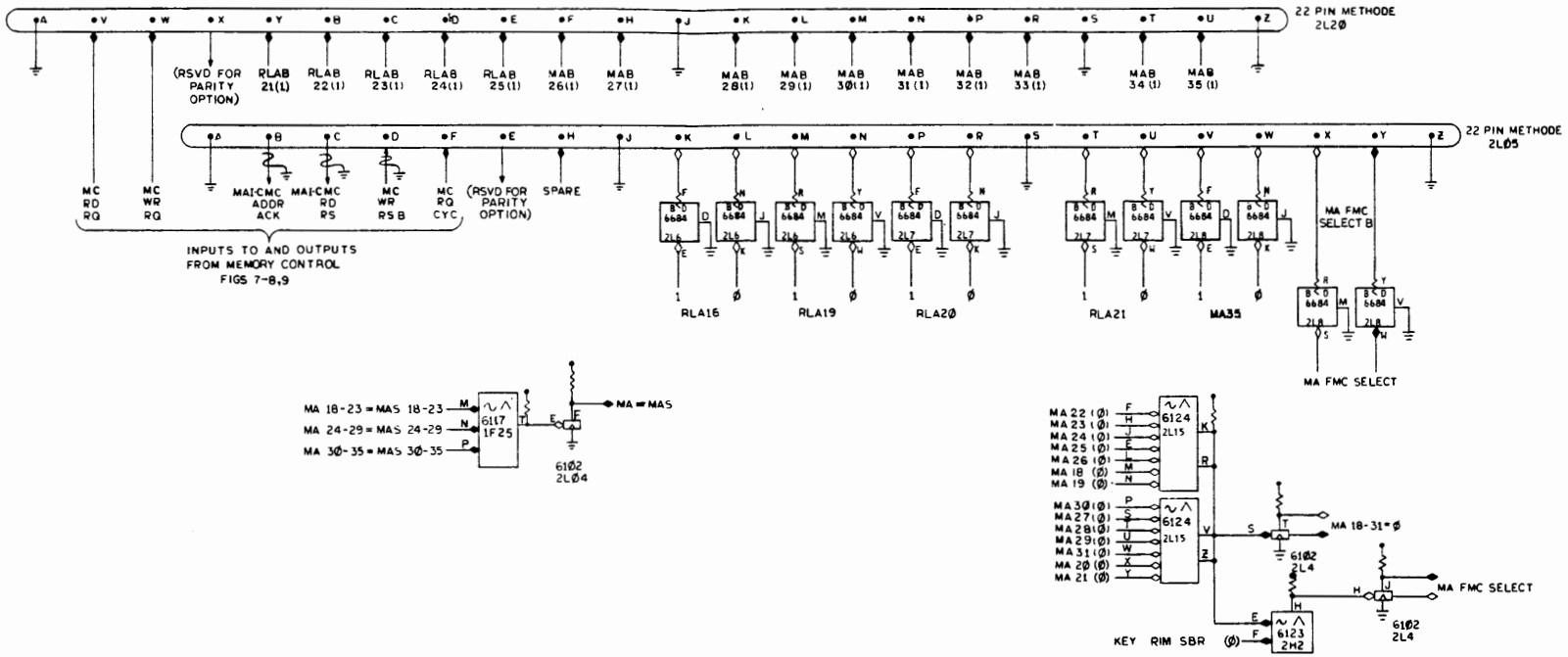
NR - Normalize Return Subroutine
(BS-D-166-0-NR)



NOTE

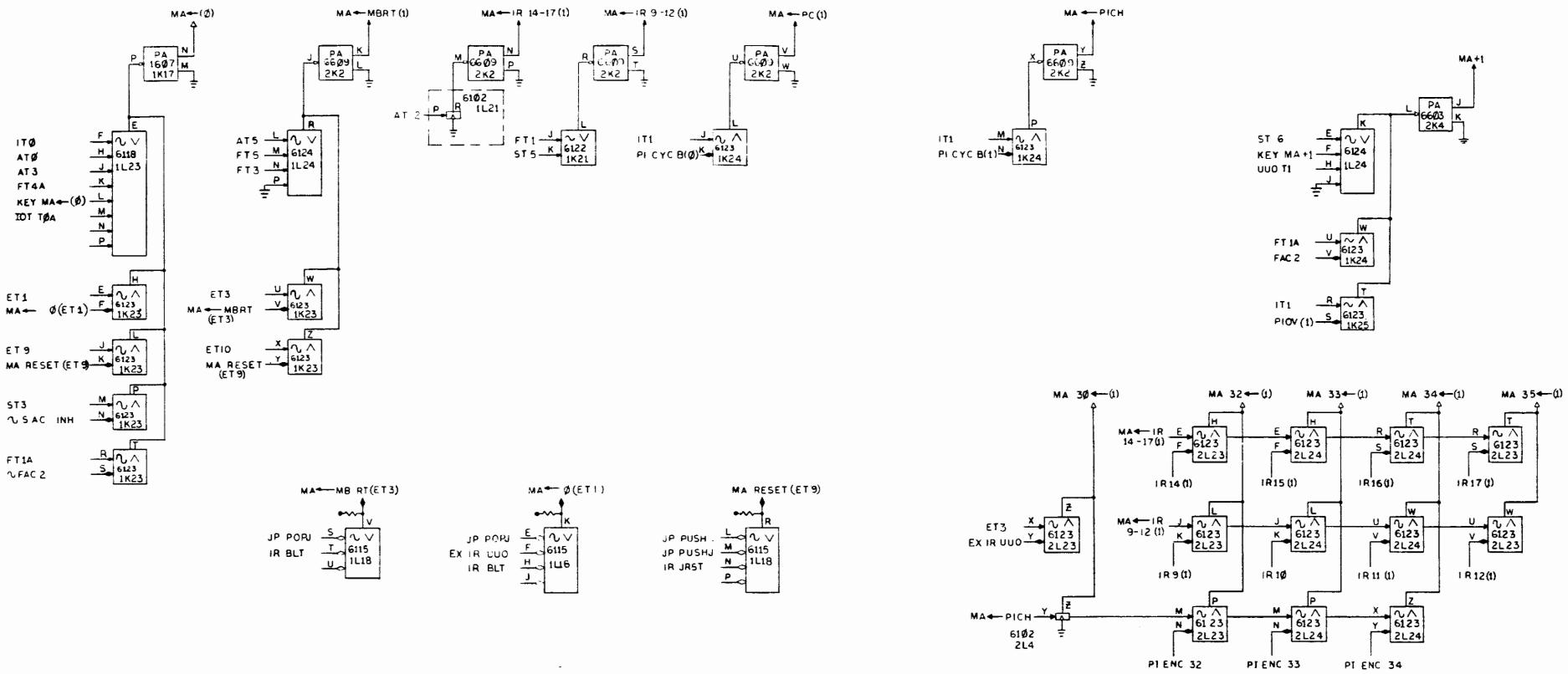
1. ALL FF PACKAGES ARE 6206
2. * INDICATES REAR CONNECTOR PIN.
3. GROUND PIN D AND Z IN ALL 6206'S.
4. USE ONLY 6684-1 OR LATER MODULES IN LOCATIONS 2L-17, 18, 19

MA - Memory Address 18-35
(BS-D-166-0-MA-2)

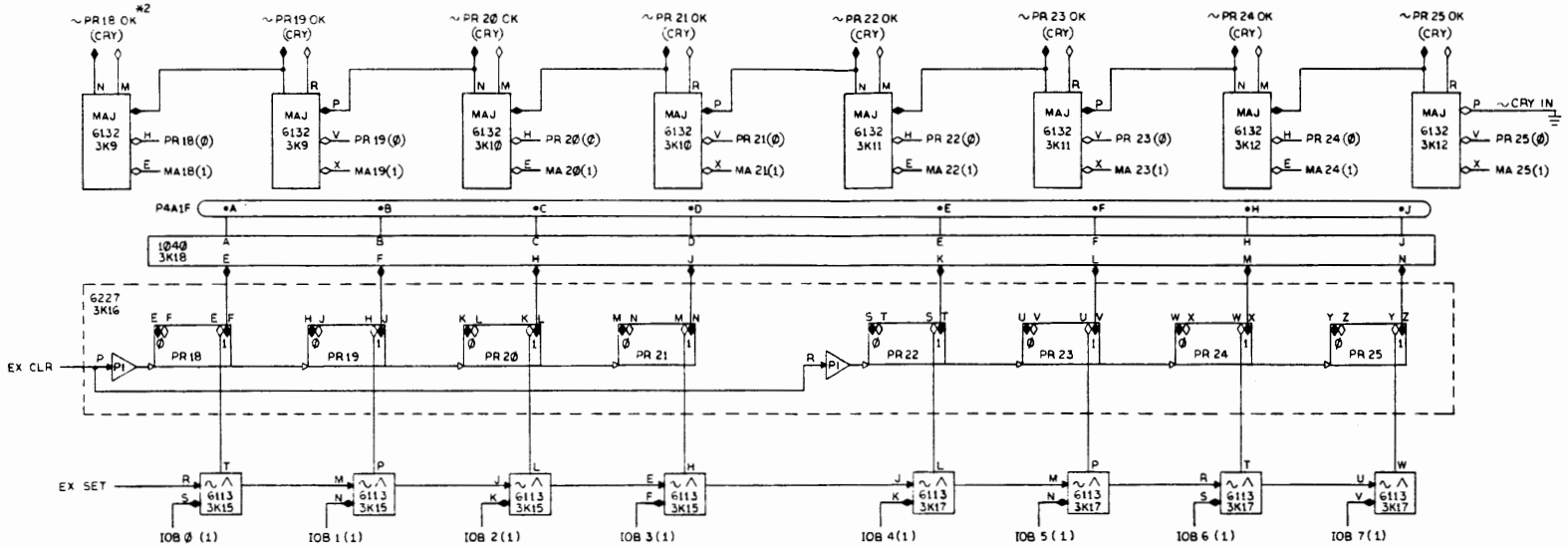


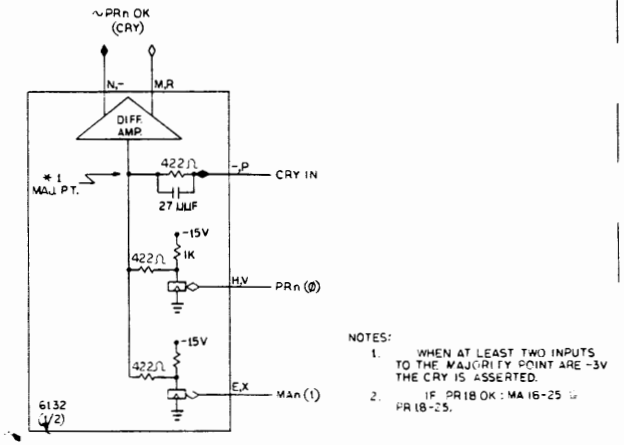
NOTE: 1 USE ONLY 6684J OR LATER MODULES IN LOCATIONS 2L-6,7,8

MA, MAI - Memory Address Interface
(BS-D-166-0-MA-3)

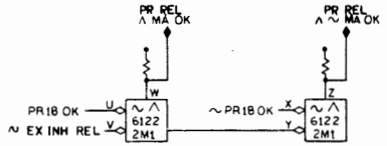


MA Control (BS-D-166-0-MA-1)

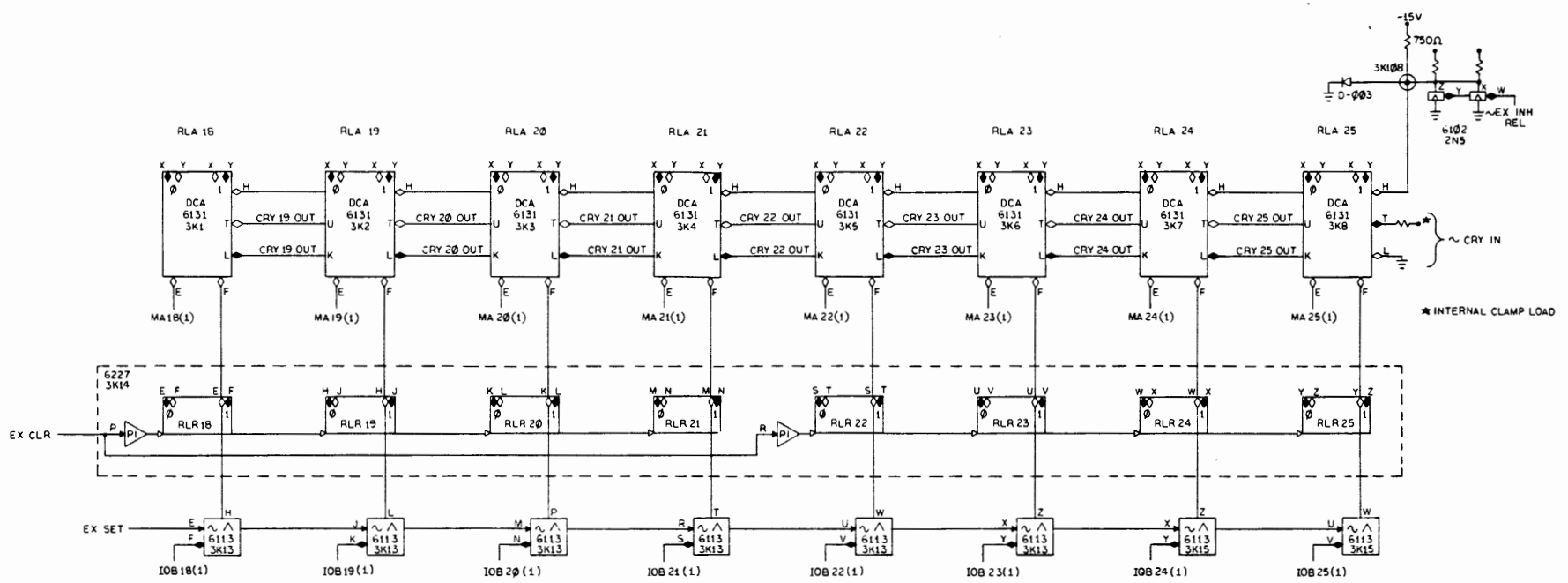


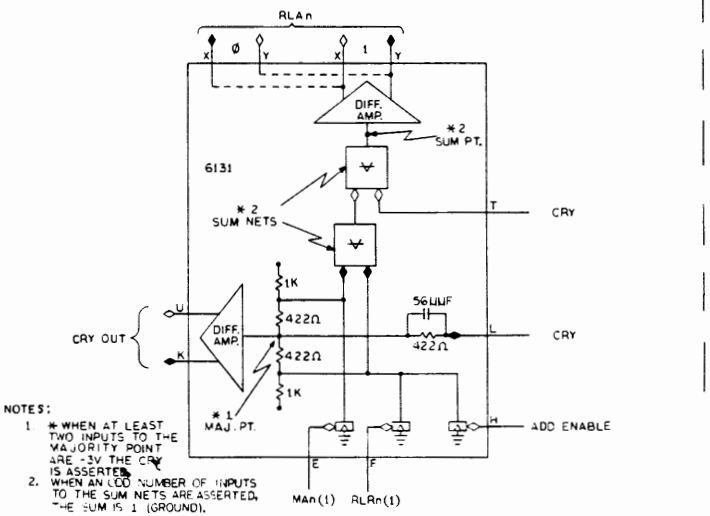


NOTES:
1. WHEN AT LEAST TWO INPUTS TO THE MAJORITY POINT ARE -3V THE CRY IS ASSERTED.
2. IF PR18 OK : MA 16-25 = PR 18-25.

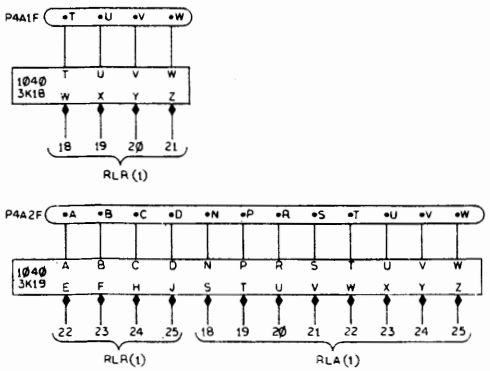


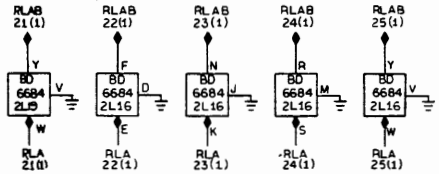
PR - Protection (BS-D-166-0-PR)



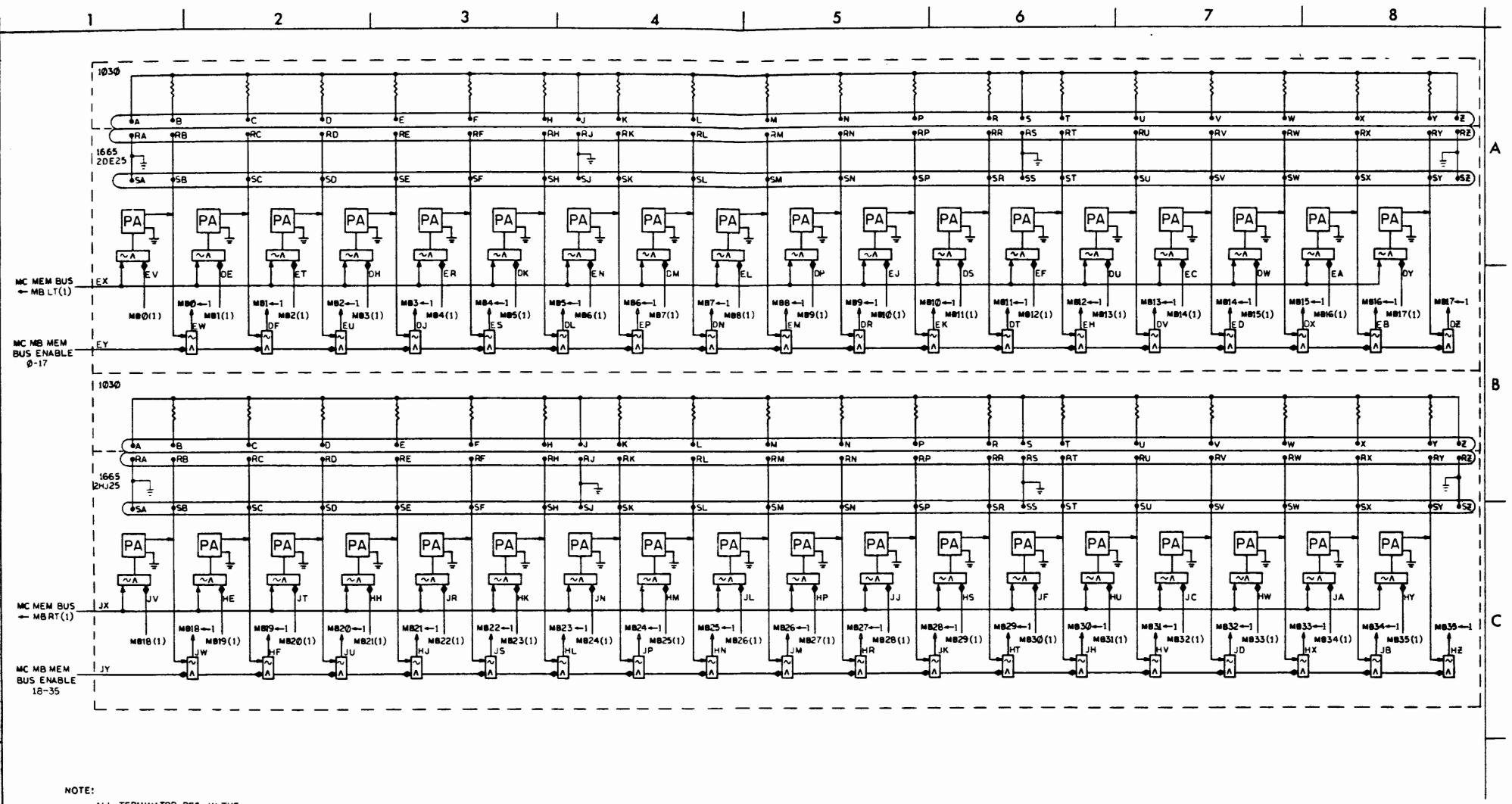


NOTES:
1. ★ WHEN AT LEAST TWO INPUTS TO THE MAJORITY POINT ARE -3V THE CRY IS ASSERTED.
2. WHEN AN ODD NUMBER OF INPUTS TO THE SUM NETS ARE ASSERTED, THE SUM IS 1 (GROUND).
3. USE ONLY 6684J OR LATER MODULES IN LOCATIONS 2L-16,19.

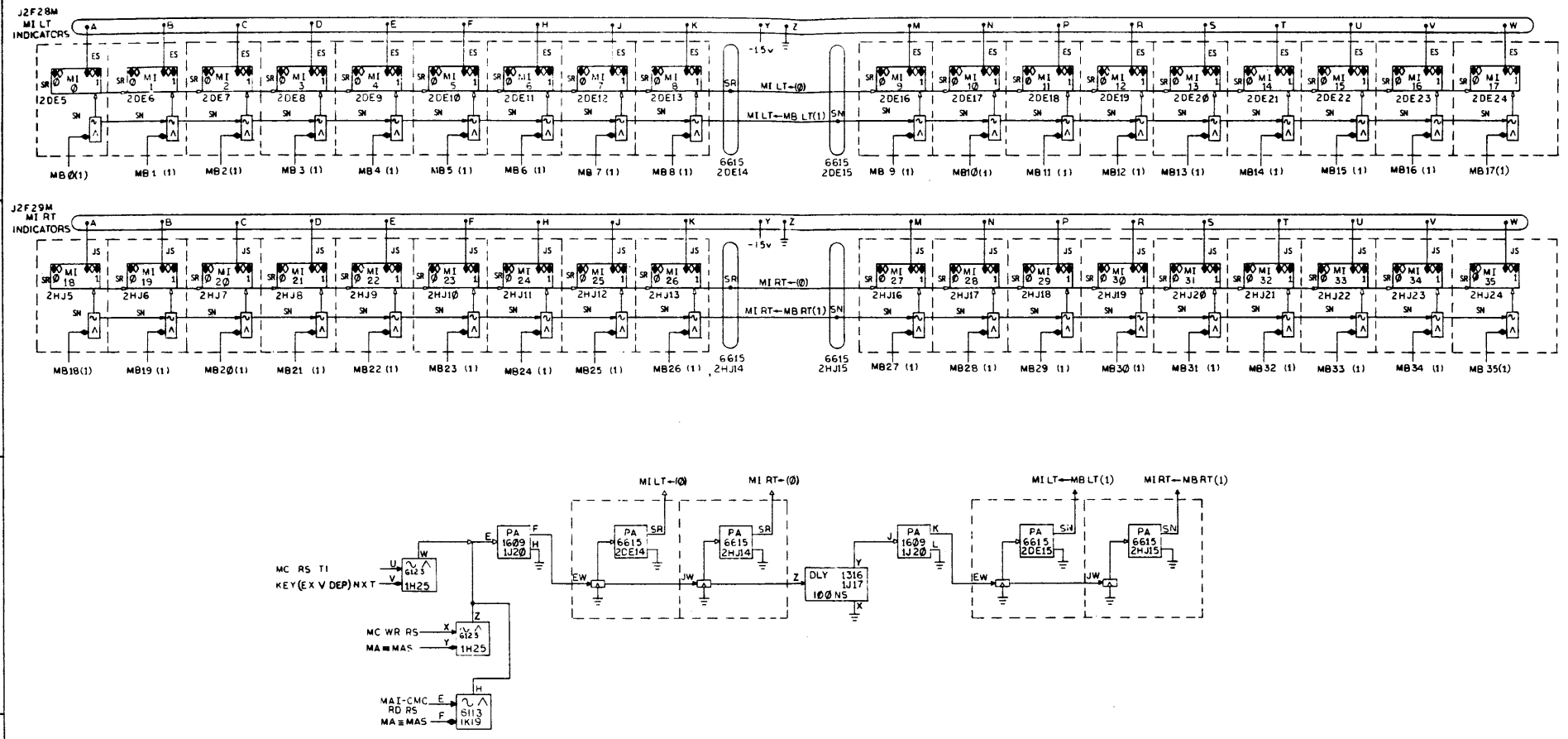




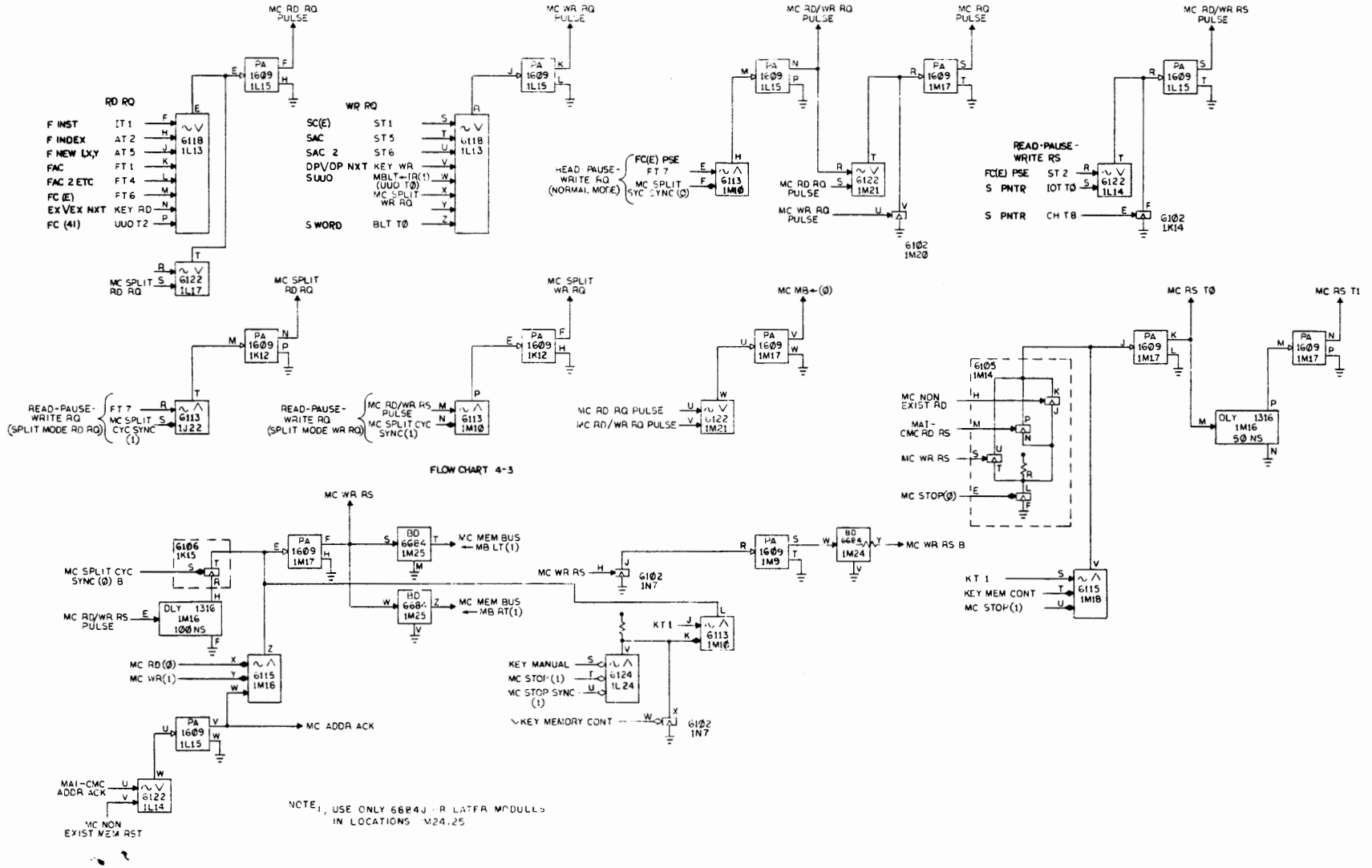
RLA, RLR - Relocation Adder and Register
(BS-D-166-0-RLAR)

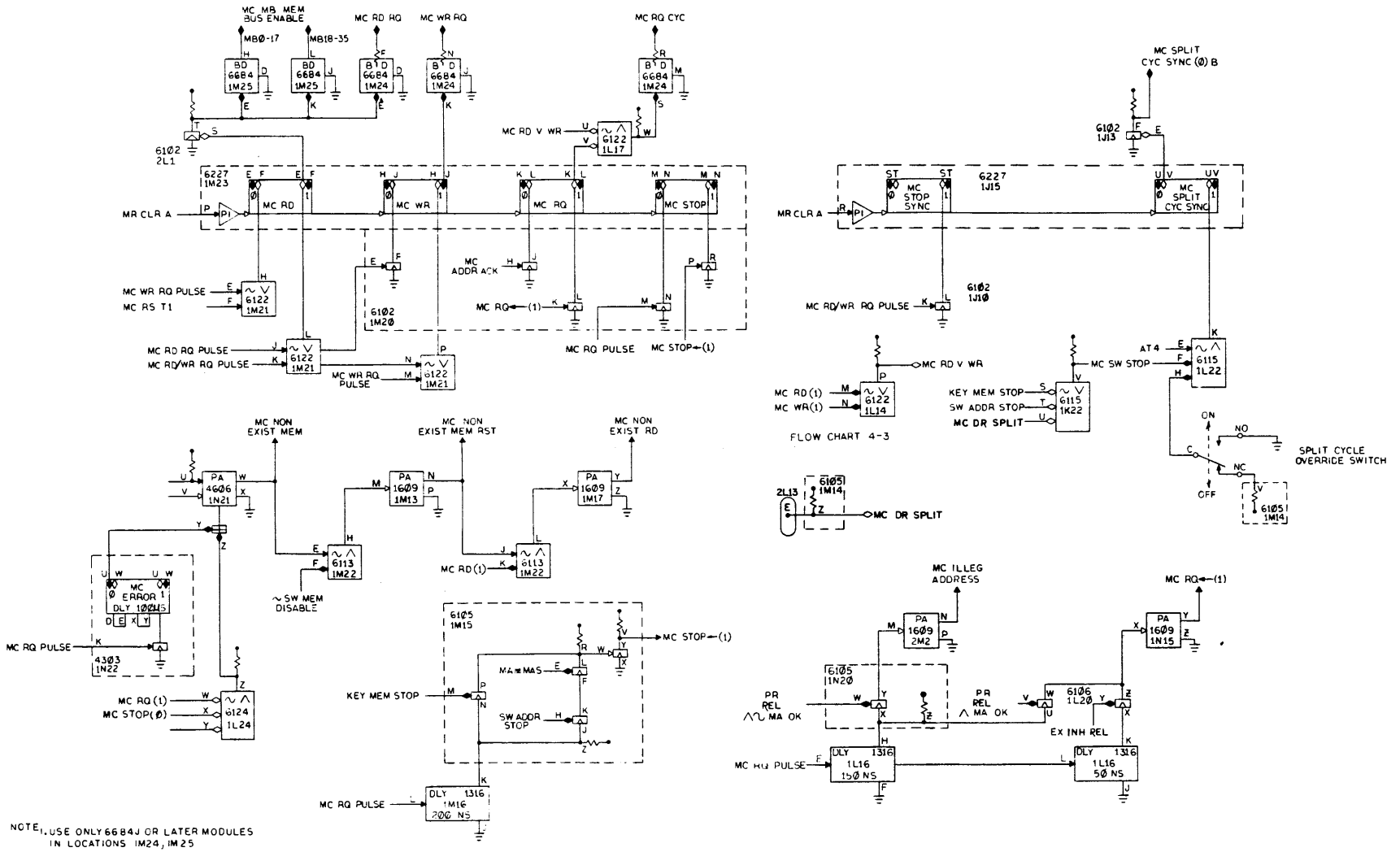


MB - Memory Buffer Interface
(BS-D-166-0-MB-4)



MI - Memory Indicators (BS-D-166-0-MI)


MC RD RQ PULSE
MC WR RQ PULSE
MC RD/WR RQ PULSE
MC RQ PULSE
MC RD/WR RS PULSE
RD RQ
WR RQ
F INST
F INDEX
F NEW LX,Y
FAC
FAC 2 ETC
FC (E)
EX VEX NXT
FC (41)
SC(E)
SAC
SAC 2
DPVDP NXT
SUUO
S WORD
READ-PAUSE-WRITE RQ (NORMAL MODE)
FC(E) PSE FT 7
MC SPLIT SYC SYNC (Ø)
MC RD RQ PULSE
MC WR RQ PULSE
READ-PAUSE-WRITE RS
FC(E) PSE
S PNTR
MC SPLIT RD RQ
MC SPLIT WR RQ
MC MB←(Ø)
READ-PAUSE-WRITE RQ (SPLIT MODE RD RQ)
READ-PAUSE-WRITE RQ (SPLIT MODE WR RQ)
MC RD/WR RS PULSE
MC SPLIT CYC SYNC(1)
MC RD RQ PULSE
MC RD/WR RQ PULSE
MC RS TØ
MC RS T1
MC NON EXIST RD
MAI-CMC RD RS
MC WR RS
MC STOP(Ø)
FLOW CHART 4-3
MC WR RS
MC SPLIT CYC SYNC (Ø) B
MC RD/WR RS PULSE
MC RD(Ø)
MC WR(1)
MC ADDR ACK
MAI-CMC ADDR ACK
MC NON EXIST MEM RST
MC MEM BUS MB LT(1)
MC MEM BUS MB RT(1)
MC WR RS B
KT 1
KEY MEM CONT
MC STOP(1)
KEY MANUAL
MC STOP(1)
MC STOP SYNC (1)
KEY MEMORY CONT
NOTE: USE ONLY 6684J OR LATER MODULES IN LOCATIONS 1M24,25

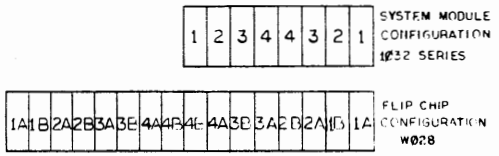


NOTE, USE ONLY 6684J OR LATER MODULES IN LOCATIONS 1M24, 1M25

MC - Memory Control (BS-D-166-0-MC-2)

IO BUS(§§8.4)

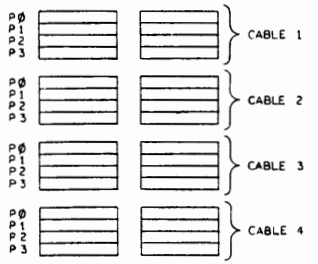
FLIP CHIP PIN	SYSTEM MODULE PIN	IO CABLE 1	IO CABLE 2	IO CABLE 3	IO CABLE 4
C	A	GND	GND	GND	GND
D	B	IOB Ø (1) →	IOB 18 (1) →	IOB RESET →	DATAO CLEAR →
E	C	IOB 1 (1) →	IOB 19 (1) →	POWER ON -15V	DATAO SET →
H	D	IOB 2 (1) →	IOB 20 (1) →		CONO CLEAR →
K	E	IOB 3 (1) →	IOB 21 (1) →	MC DR SPLIT →	CONO SET →
M	F	IOB 4 (1) →	IOB 22 (1) →	IOS 3 (Ø) →	IOB ← DATAI →
P	H	IOB 5 (1) →	IOB 23 (1) →	IOS 3 (1) →	IOB ← STATUS →
N	J	GND	GND	GND	GND
S	K	IOB 6 (1) →	IOB 24 (1) →	IOS 4 (Ø) →	
T	L	IOB 7 (1) →	IOB 25 (1) →	IOS 4 (1) →	
V	M	IOB 8 (1) →	IOB 26 (1) →	IOS 5 (Ø) →	
D	N	IOB 9 (1) →	IOB 27 (1) →	IOS 5 (1) →	
E	P	IOB 10 (1) →	IOB 28 (1) →	IOS 6 (Ø) →	
H	R	IOB 11 (1) →	IOB 29 (1) →	IOS 6 (1) →	PI REQ 1 →
J	S	GND	GND	GND	GND
K	T	IOB 12 (1) →	IOB 30 (1) →	IOS 7 (Ø) →	PI REQ 2 →
M	U	IOB 13 (1) →	IOB 31 (1) →	IOS 7 (1) →	PI REQ 3 →
P	V	IOB 14 (1) →	IOB 32 (1) →	IOS 8 (Ø) →	PI REQ 4 →
S	W	IOB 15 (1) →	IOB 33 (1) →	IOS 8 (1) →	PI REQ 5 →
T	X	IOB 16 (1) →	IOB 34 (1) →	IOS 9 (Ø) →	PI REQ 6 →
V	Y	IOB 17 (1) →	IOB 35 (1) →	IOS 9 (1) →	PI REQ 7 →
U	Z	GND	GND	GND	GND
		(SOURCE 3L6)	(SOURCE 3L7)	(SOURCE 2L13)	(SOURCE 2N25)

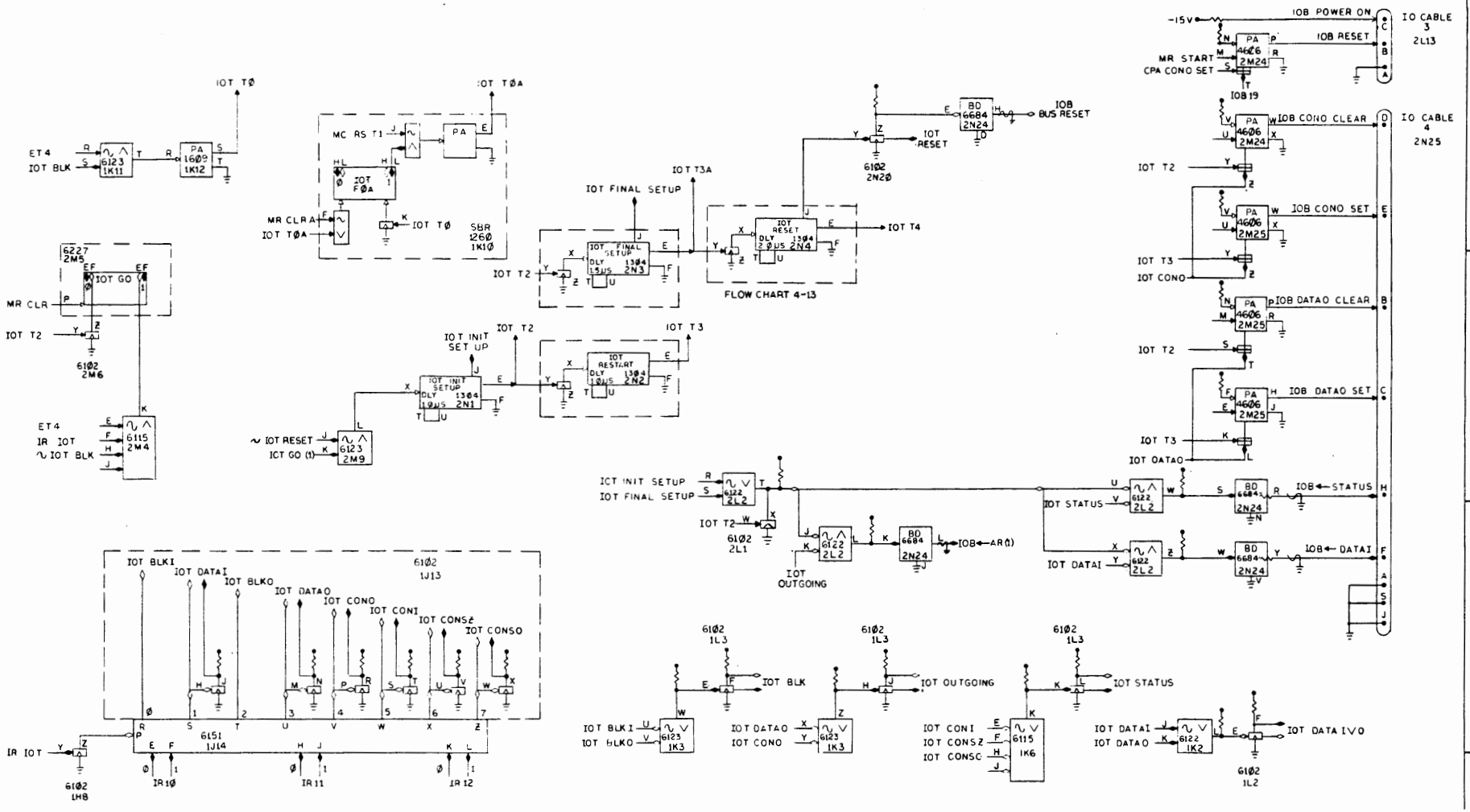


NOTE: 1 SYSTEM MODULE CONNECTOR (1Ø32) = 2 FLIP CHIP CONNECTORS (WØ28)

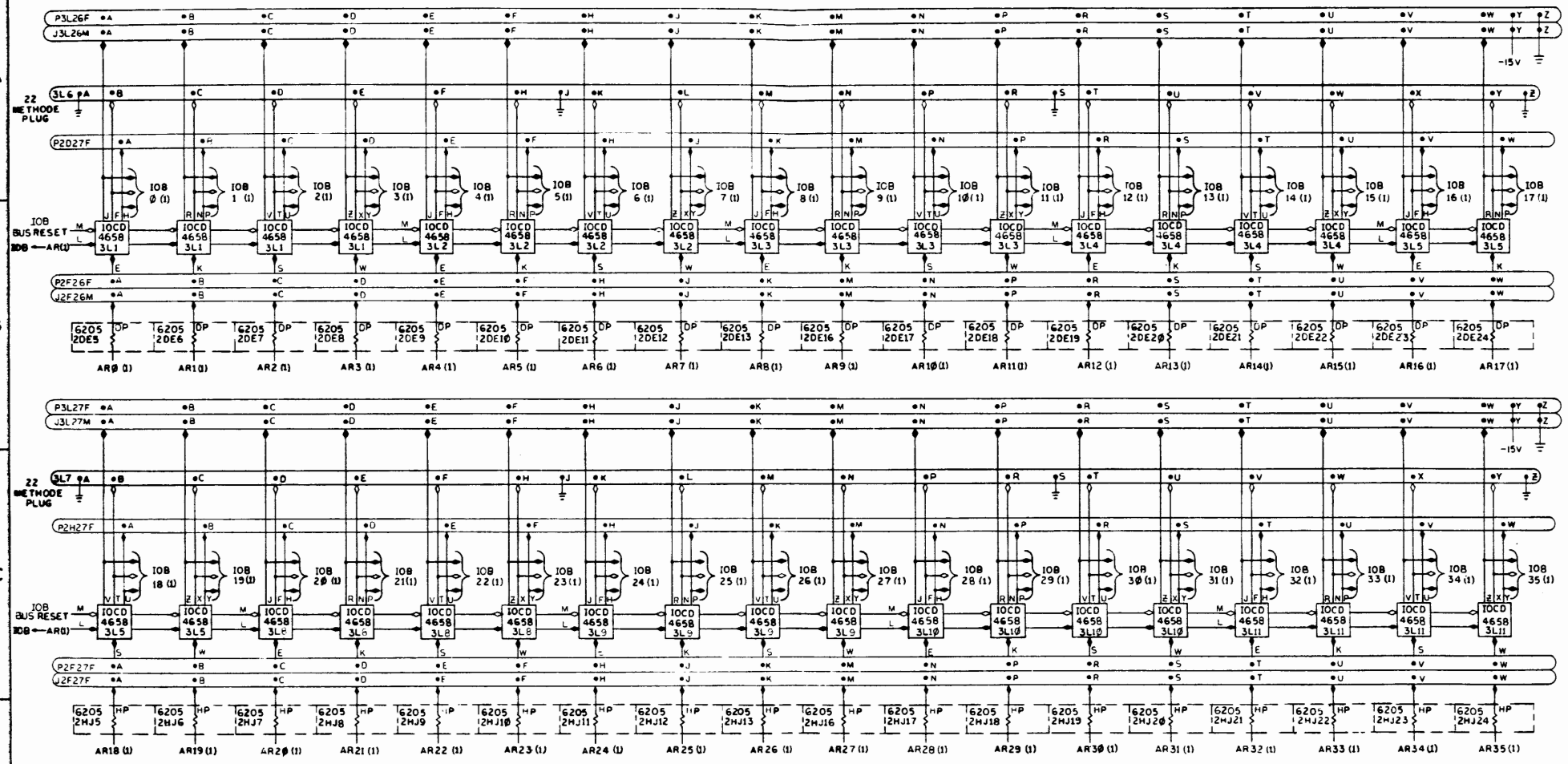
MEM BUS(§§7.1, 7.2)

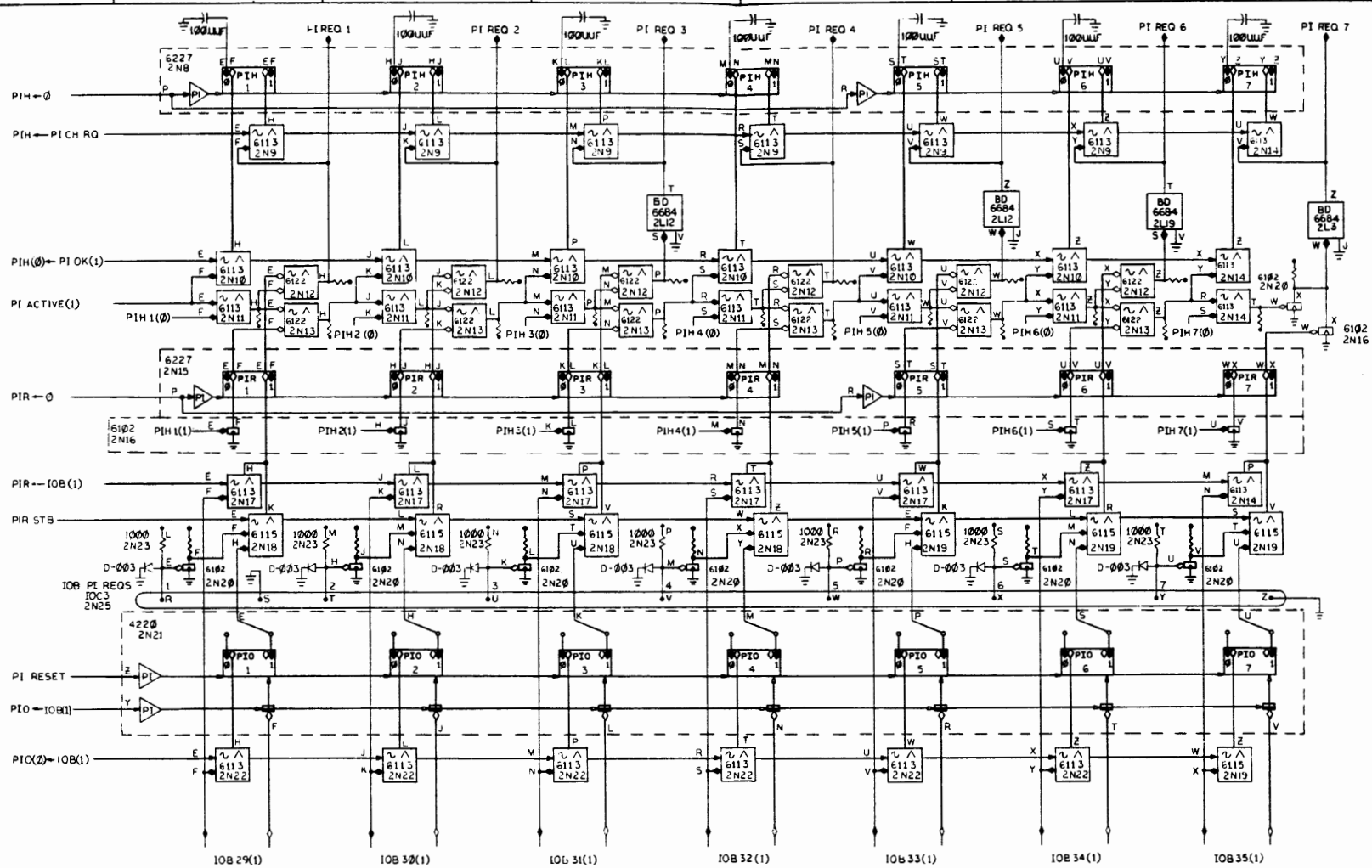
FLIP CHIP PIN	SYSTEM MODULE PIN	MEM CABLE 1	MEM CABLE 2	MEM CABLE 3	MEM CABLE 4
C	A	GND	GND	GND	GND
D	B	ADDR ACK →	MA 22 (1) →	MB Ø (1) →	MB 18 (1) →
E	C	RD RS →	MA 23 (1) →	MB 1 (1) →	MB 19 (1) →
H	D	WR RS →	MA 24 (1) →	MB 2 (1) →	MB 20 (1) →
K	E	PAR (1) →	MA 25 (1) →	MB 3 (1) →	MB 21 (1) →
M	F	RQ CYCLE →	MA 26 (1) →	MB 4 (1) →	MB 22 (1) →
P	H	SPARE	MA 27 (1) →	MB 5 (1) →	MB 23 (1) →
N	J	GND	GND	GND	GND
S	K	MA 18 (1) →	MA 28 (1) →	MB 6 (1) →	MB 24 (1) →
T	L	MA 18 (Ø) →	MA 29 (1) →	MB 7 (1) →	MB 25 (1) →
V	M	MA 19 (1) →	MA 30 (1) →	MB 8 (1) →	MB 26 (1) →
D	N	MA 19 (Ø) →	MA 31 (1) →	MB 9 (1) →	MB 27 (1) →
E	P	MA 20 (1) →	MA 32 (1) →	MB 10 (1) →	MB 28 (1) →
H	R	MA 20 (Ø) →	MA 33 (1) →	MB 11 (1) →	MB 29 (1) →
J	S	GND	GND	GND	GND
K	T	MA 21 (1) →	MA 34 (1) →	MB 12 (1) →	MB 30 (1) →
M	U	MA 21 (Ø) →	MA 35 (1) →	MB 13 (1) →	MB 31 (1) →
P	V	MA 35 (1) →	MC RD RQ →	MB 14 (1) →	MB 32 (1) →
S	W	MA 35 (Ø) →	MC WR RQ →	MB 15 (1) →	MB 33 (1) →
T	X	MAFMIC →	PAR OPTION →	MB 16 (1) →	MB 34 (1) →
V	Y	SELECT B →	MA 21 (1) →	MB 17 (1) →	MB 35 (1) →
U	Z	GND	GND	GND	GND
		(SOURCE 2L5)	(SOURCE 2L2Ø)	(SOURCE 2E25)	(SOURCE 2J25)



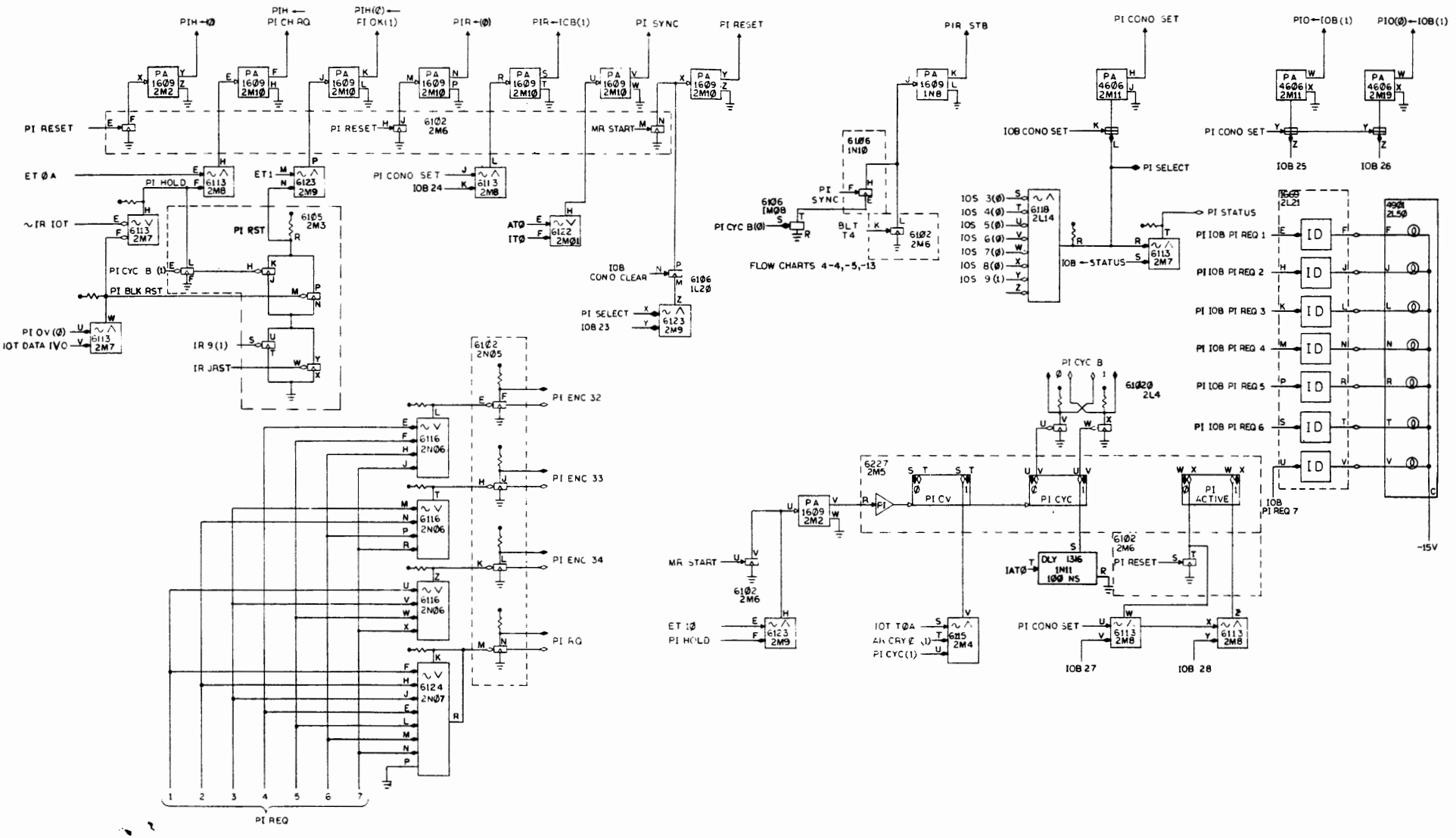


IOT, IOB Control (BS-D-166-0-IOTB)

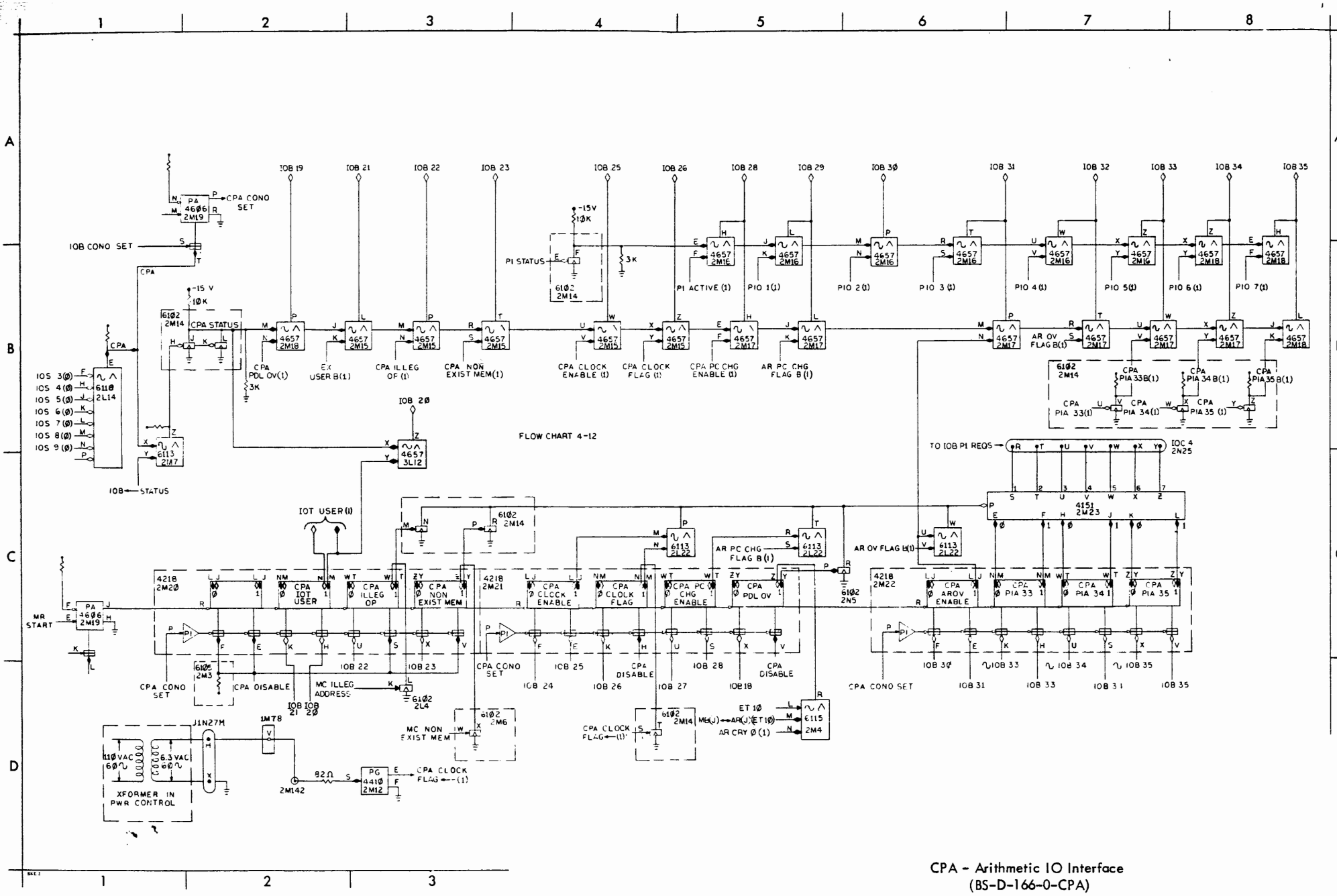

P3L26F
J3L26M
3L6
22 METHODE PLUG
P2D27F
IOB BUS RESET
IOB ← AR(1)
IOCD 4658 3L1
IOCD 4658 3L2
IOCD 4658 3L3
IOCD 4658 3L4
IOCD 4658 3L5
P2F26F
J2F26M
6205 2DE5
6205 2DE6
6205 2DE7
6205 2DE8
6205 2DE9
6205 2DE10
6205 2DE11
6205 2DE12
6205 2DE13
6205 2DE16
6205 2DE17
6205 2DE18
6205 2DE19
6205 2DE20
6205 2DE21
6205 2DE22
6205 2DE23
6205 2DE24
AR0 (1)
AR1 (1)
AR2 (1)
AR3 (1)
AR4 (1)
AR5 (1)
AR6 (1)
AR7 (1)
AR8 (1)
AR9 (1)
AR10 (1)
AR11 (1)
AR12 (1)
AR13 (1)
AR14 (1)
AR15 (1)
AR16 (1)
AR17 (1)
P3L27F
J3L27M
3L7
22 METHODE PLUG
P2H27F
P2F27F
J2F27F
IOB BUS RESET
IOB ← AR(1)
IOCD 4658 3L5
IOCD 4658 3L8
IOCD 4658 3L9
IOCD 4658 3L10
IOCD 4658 3L11
6205 2HJ5
6205 2HJ6
6205 2HJ7
6205 2HJ8
6205 2HJ9
6205 2HJ10
6205 2HJ11
6205 2HJ12
6205 2HJ13
6205 2HJ16
6205 2HJ17
6205 2HJ18
6205 2HJ19
6205 2HJ20
6205 2HJ21
6205 2HJ22
6205 2HJ23
6205 2HJ24
AR18 (1)
AR19 (1)
AR20 (1)
AR21 (1)
AR22 (1)
AR23 (1)
AR24 (1)
AR25 (1)
AR26 (1)
AR27 (1)
AR28 (1)
AR29 (1)
AR30 (1)
AR31 (1)
AR32 (1)
AR33 (1)
AR34 (1)
AR35 (1)
-15V



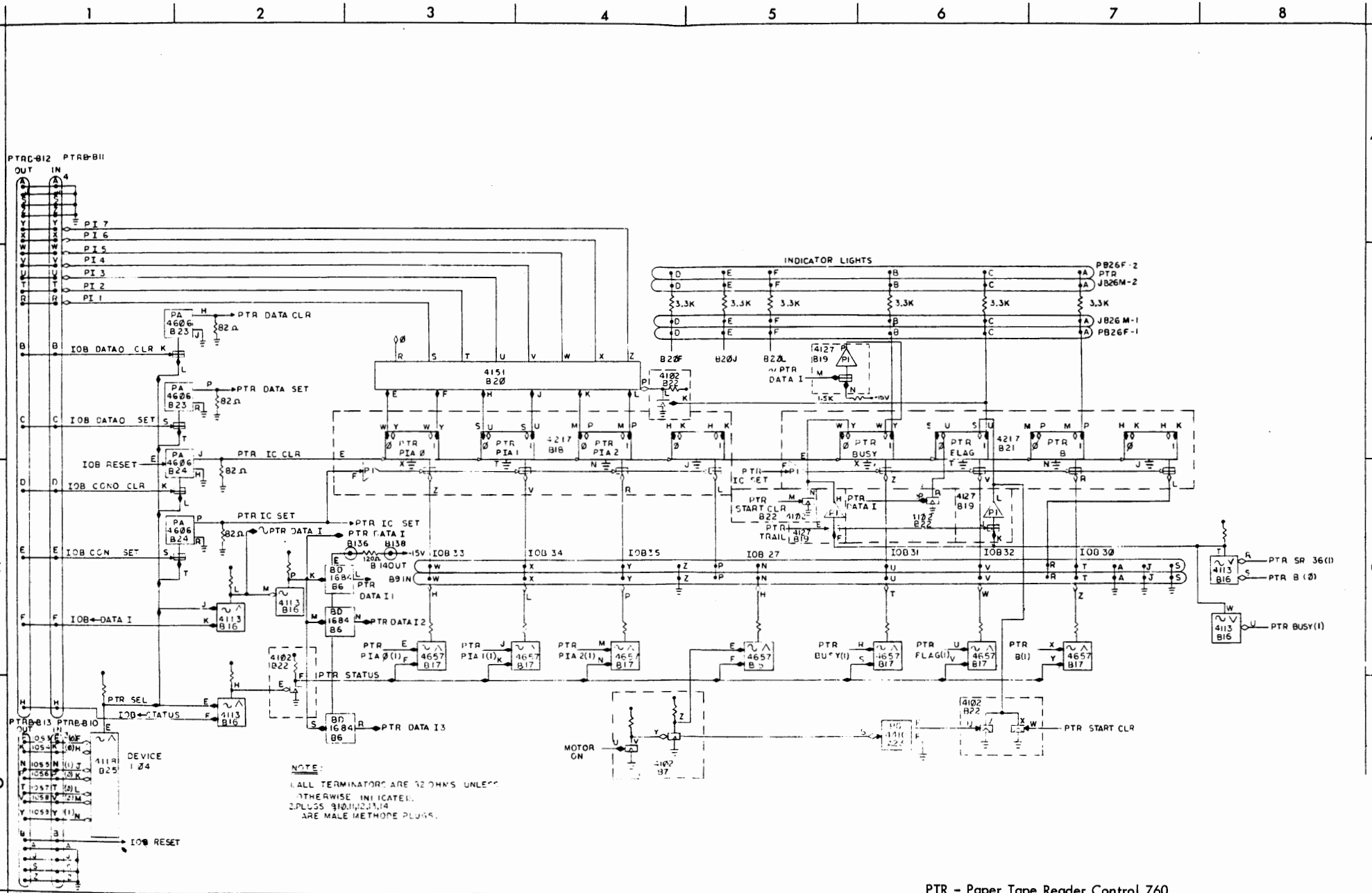
PI - PIH, R, O, 1-7 (BS-D-166-0-PI-2)



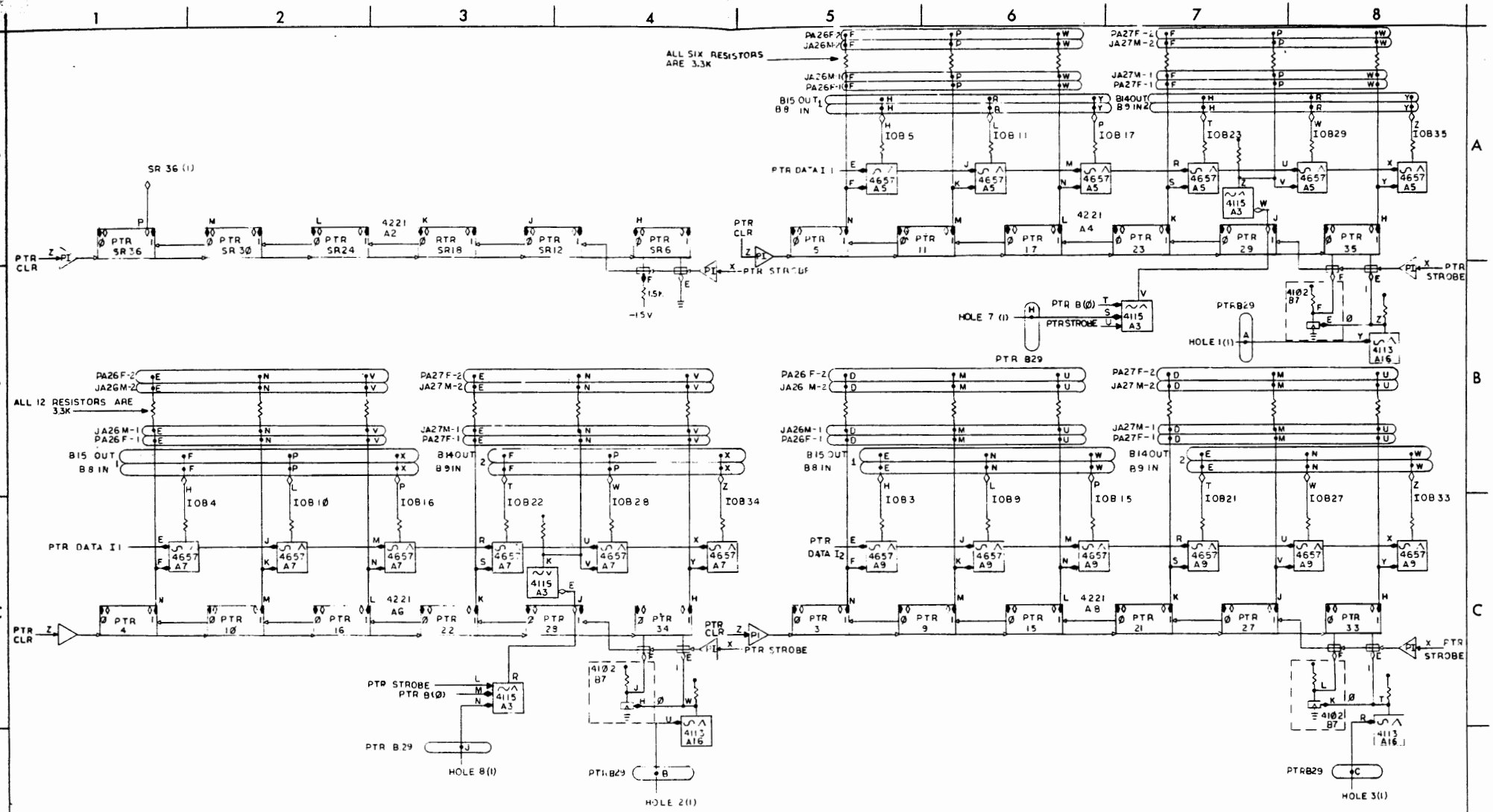
PI Control (BS-D-166-0-PI-1)



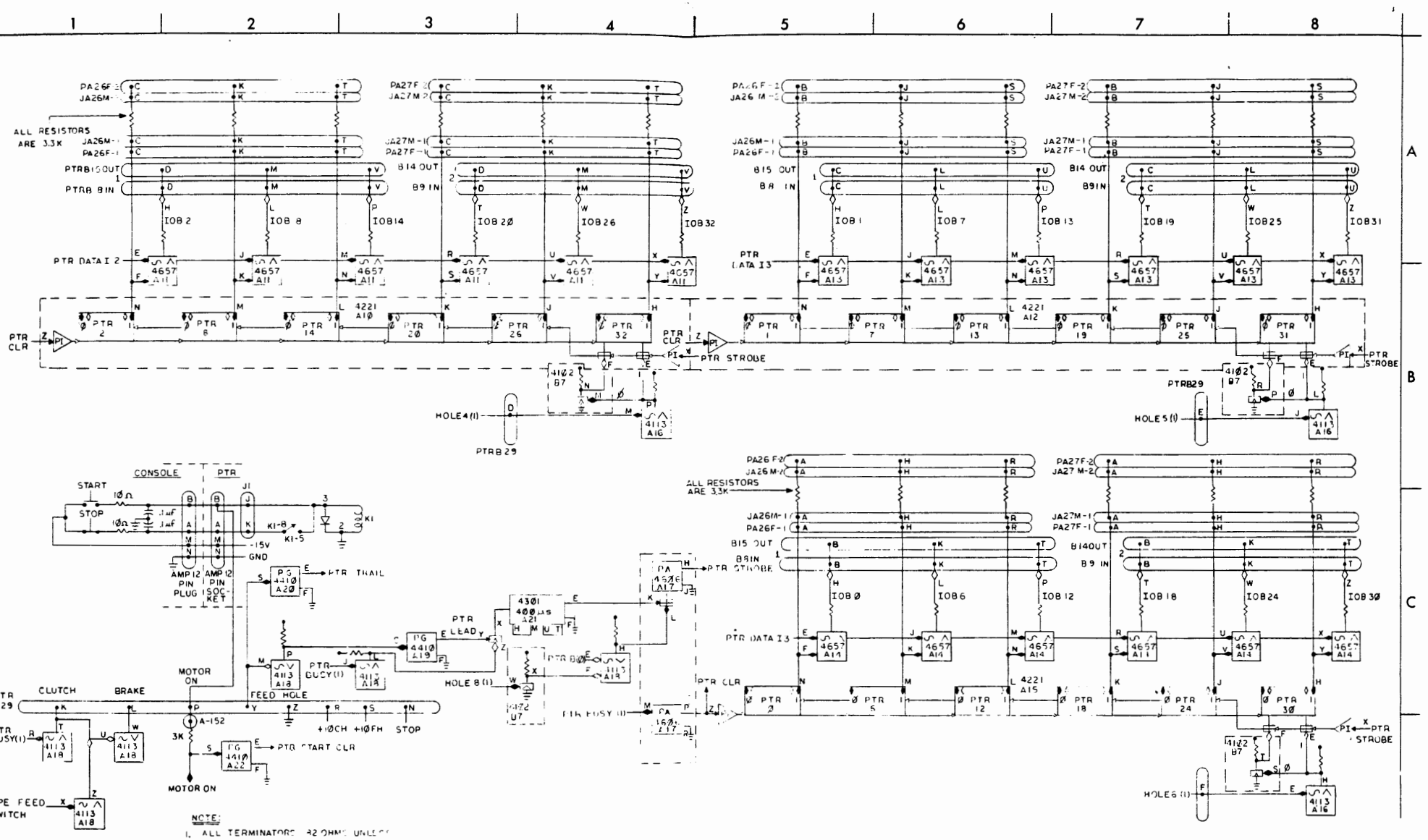
CPA - Arithmetic IO Interface
(BS-D-166-0-CPA)



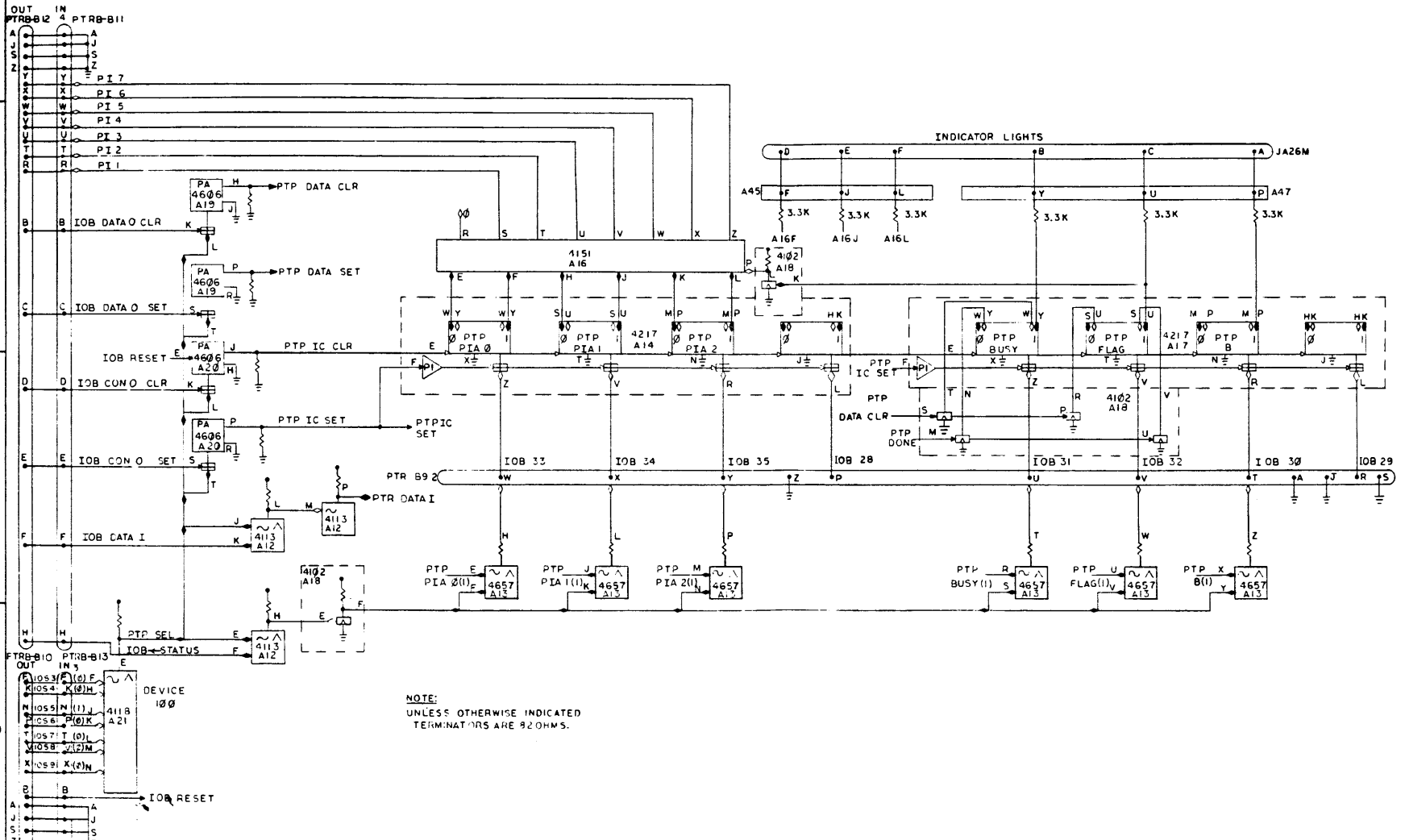
PTR - Paper Tape Reader Control 760
(BS-D-760-0-2, sheet 3)



PTR – Paper Tape Reader Control 760
(BS-D-760-0-2, sheet 1)

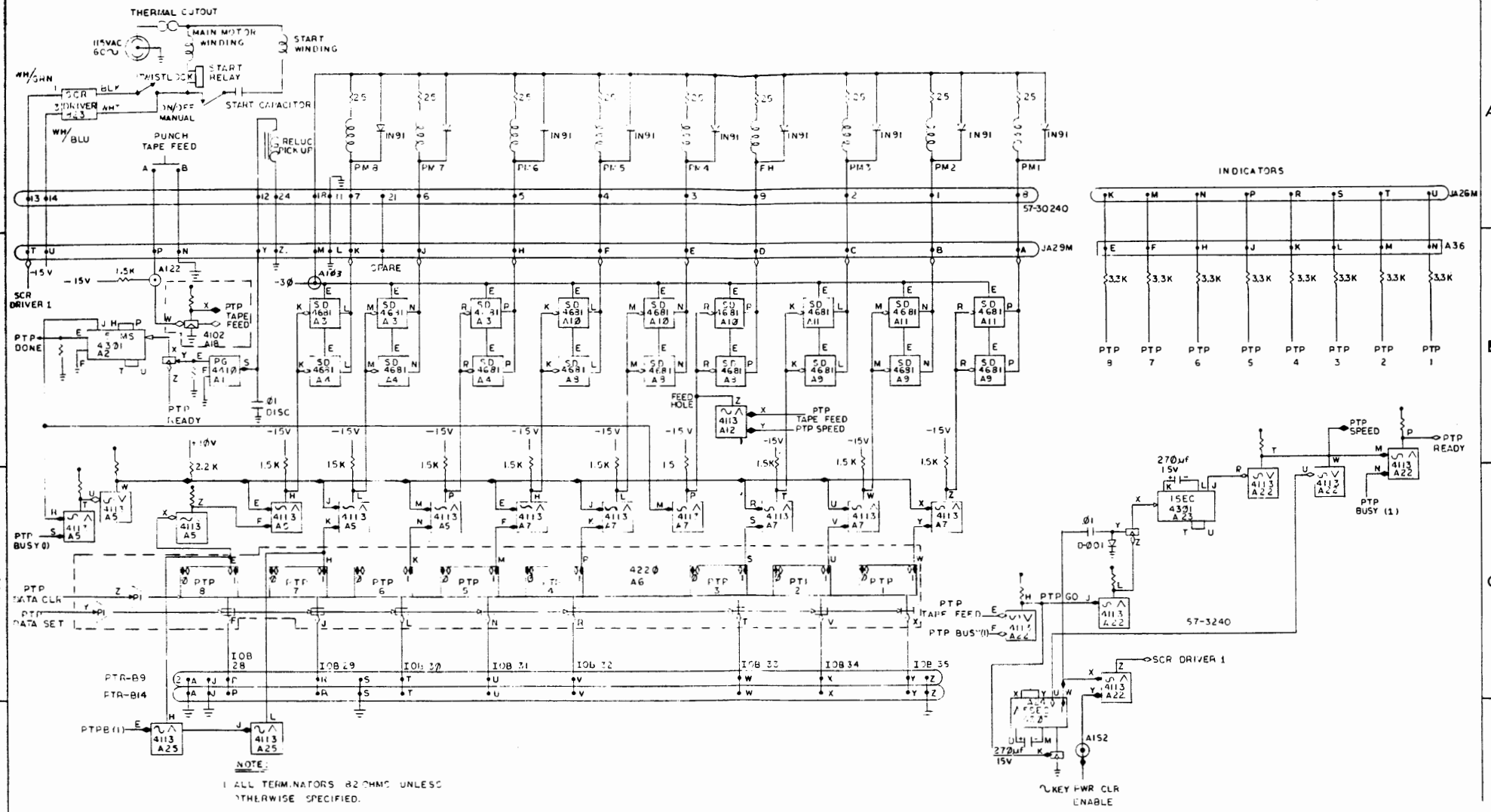


PTR – Paper Tape Reader Control 760
(BS-D-760-0-2, sheet 2)

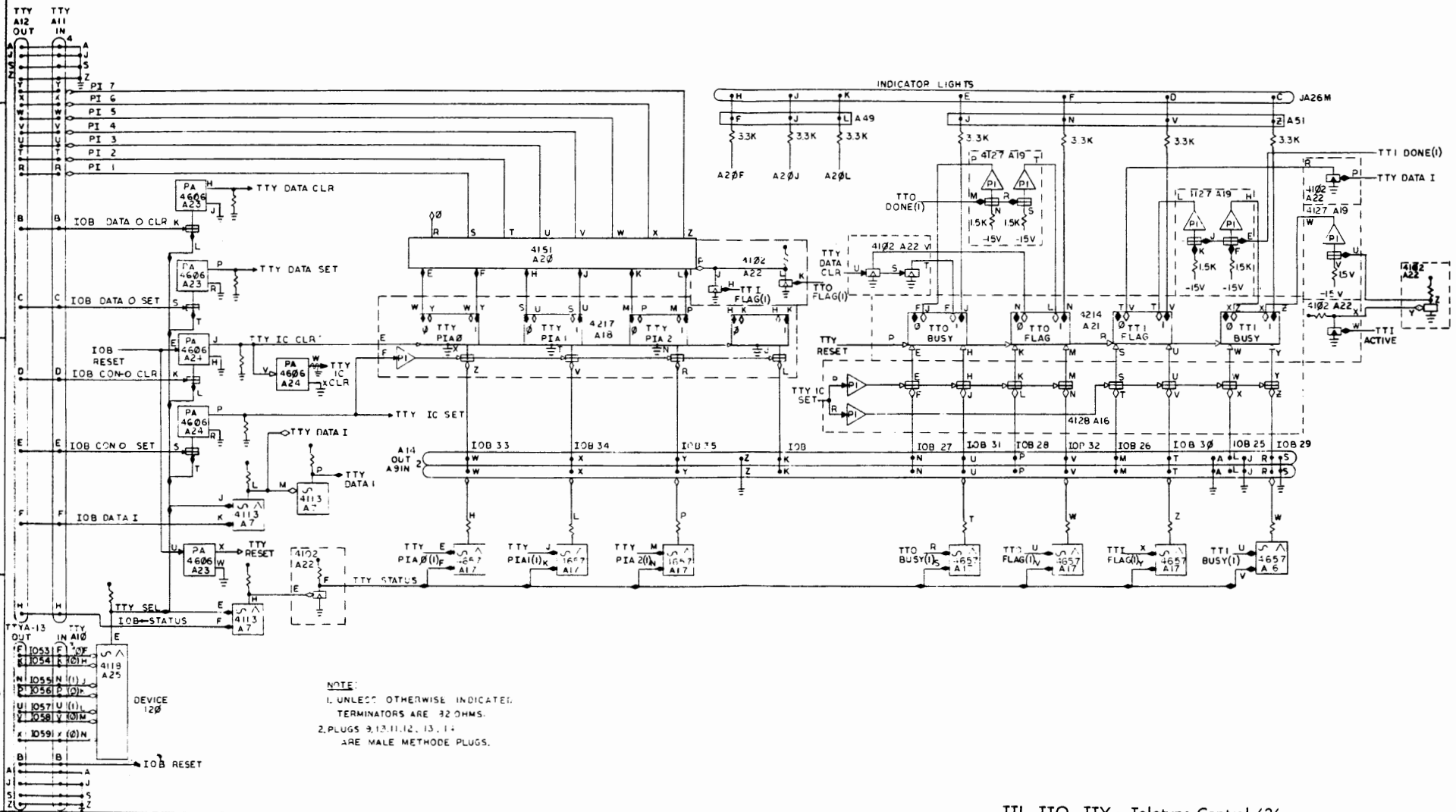


NOTE:
UNLESS OTHERWISE INDICATED
TERMINATORS ARE 82 OHMS.

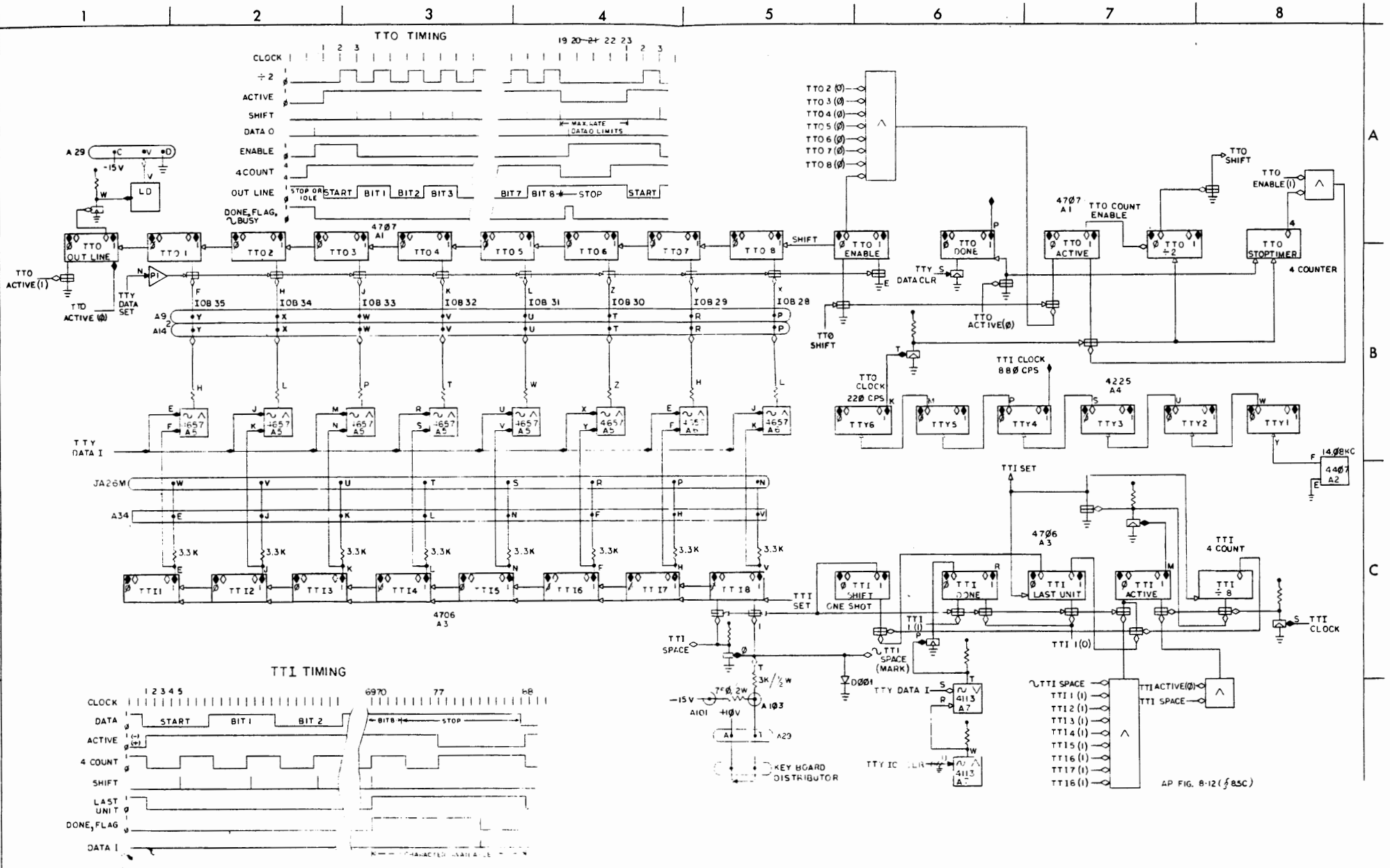
PTP – Paper Tape Punch Control 761
(BS-D-761-0-2, sheet 2)



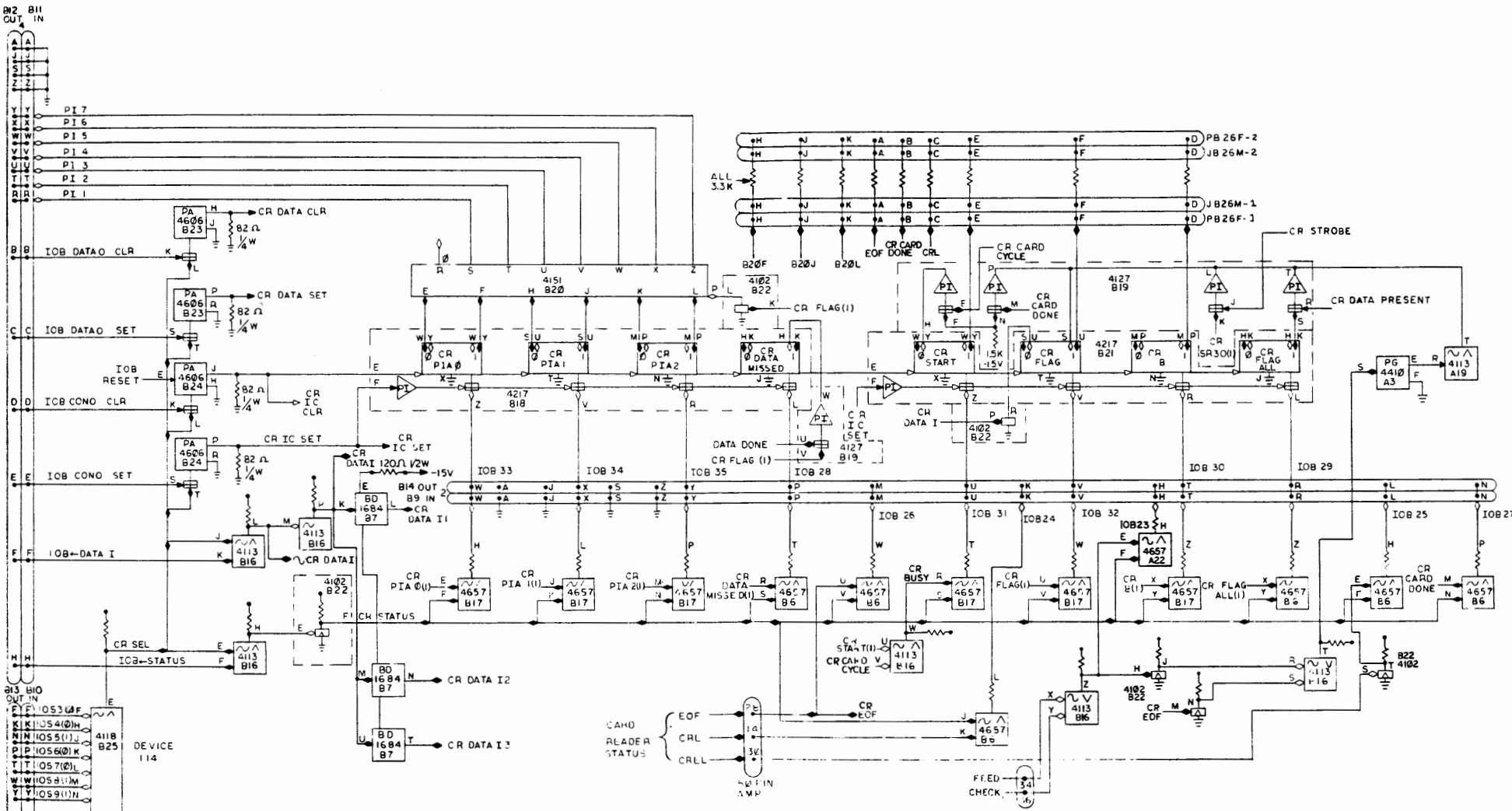
PTP – Paper Tape Punch Control 761
(BS-D-761-0-2, sheet 1)

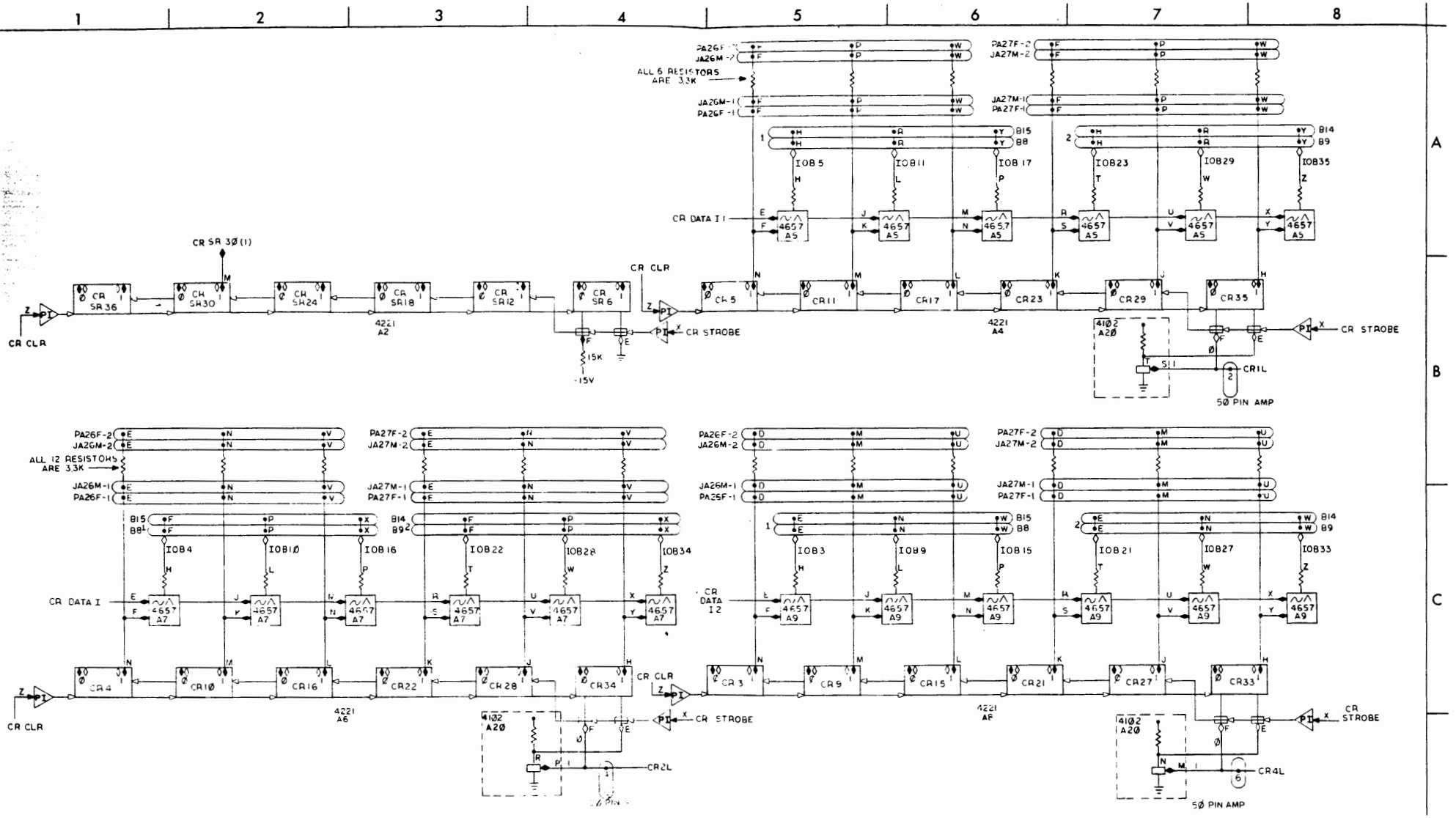


TTI, TTO, TTY - Teletype Control 626
(BS-D-626-0-2, sheet 1)

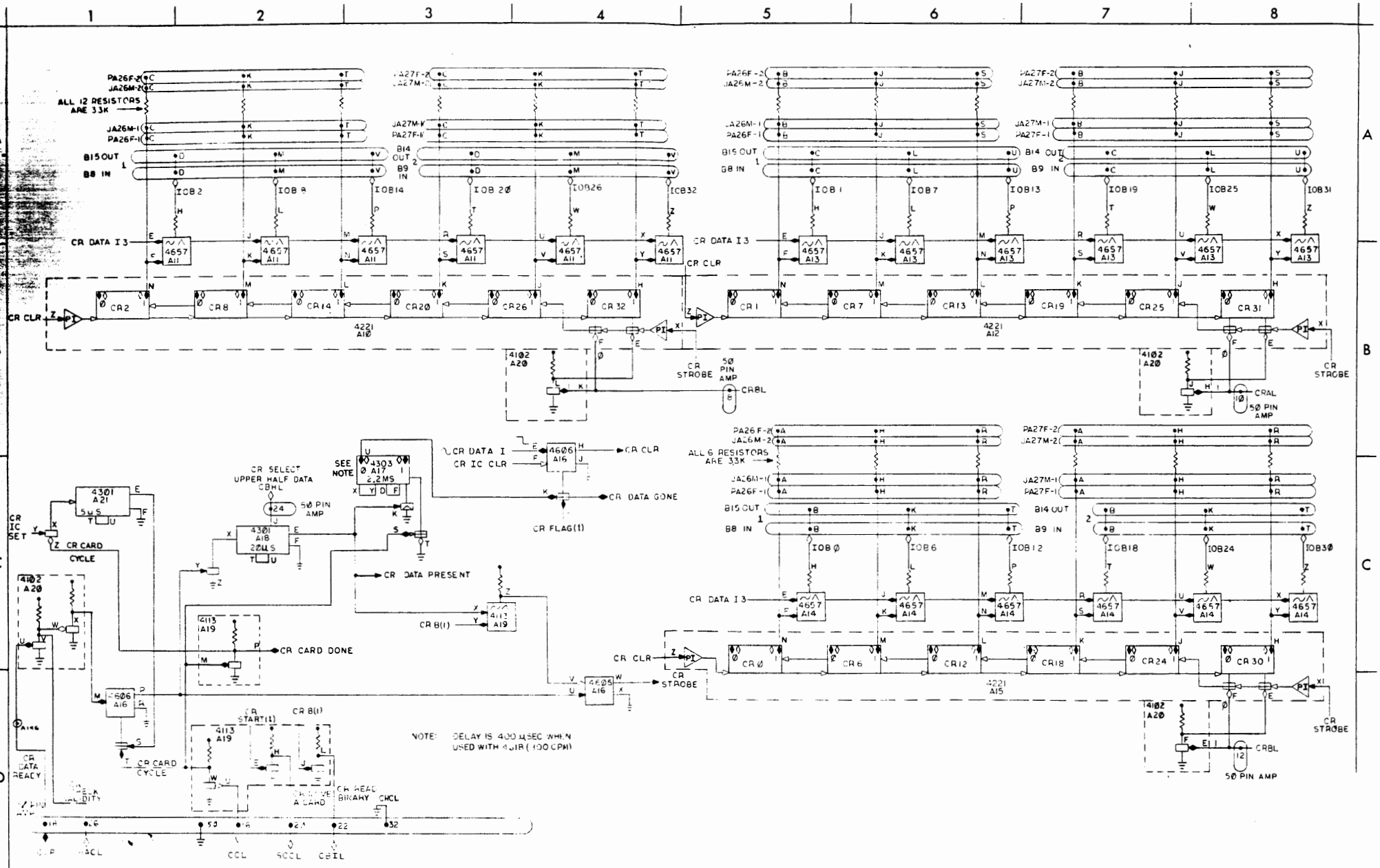


TTI, TTO, TTY – Teletype Control 626
(BS-D-626-0-2, sheet 2)


PI 7
PI 6
PI 5
PI 4
PI 3
PI 2
PI 1
IOB DATAO CLR
IOB DATAO SET
IOB RESET
IOB CONO CLR
IOB CONO SET
IOB←DATA I
IOB←STATUS
IOB RESET
CR DATA CLR
CR DATA SET
CR IC CLR
CR IC SET
CR DATA I
CR DATA I2
CR DATA I3
CR SEL
DEVICE 114
CR STROBE
CR DATA PRESENT
CR CARD CYCLE
CR CARD DONE
CR FLAG(I)
DATA DONE
CR FLAG (I)
CR BUSY
CR EOF
CARD READER STATUS
EOF
CRL
CRLL
FEED CHECK
CR - Card Reader Control 461 (BS-D-461-0-2, sheet 3)



CR - Card Reader Control 461 (BS-D-461-0-2, sheet 1)



CR - Card Reader Control 461 (BS-D-461-0-2, sheet 2)