Not approved Sept 9 '68 2/13/68

Engineering Publications Dept. No. PTP 792

1 C -
+ L + s. Poems

MICRO-PROGRAMMED IMPLEMENTATION OF THE IBM SYSTEM/360 MODEL 30 MACHINE ORGANIZATION

by.

J. E. Greene

R. F. Dean

B. M. Updike

.81 pages/page

rev. 81

9 pictures

14½
9
23½

IBM
General Products Division
Development Laboratory
Endicott, N.Y.
April 2, 1964
607-755-4576

MICRO-PROGRAMMED IMPLEMENTATION OF THE IBM SYSTEM/360 MODEL 30 MACHINE ORGANIZATION

ABSTRACT

The model 30 processor of the IBM System/360 product line is examined in some detail. Considerations leading to the processor design are discussed, with particular emphasis placed on the employment of read-only storage techniques. Micro-programming of the Read-Only Storage is employed in implementing the instruction set as well as in supervision of the input/output channels.

The detailed description of the Model 30 data flow and control is preceded by a brief examination of those aspects of System/360 conceptual definition which allow its several processor models to compatibly span a range of prices and performance.

MICRO-PROGRAMMED IMPLEMENTATION OF THE IBM SYSTEM/360 MODEL 30 MACHINE ORGANIZATION

by

J. E. Greene

R. F. Dean

B. M. Updike

INTRODUCTION

THE IBM SYSTEM/360

Earlier this year the International Business Machines Corporation announced the development of its System/360 line of processors. It will be the purpose of this discussion to describe some aspects of the design of the smallest member of the System/360 family — the Model 30 processor.

The IBM System/360 Model 30 is implemented in the latest solid state technology, Solid Logic Technology. It employs the read-only storage unit to control instruction interpretation and execution cycles as well as a number of input/output channel control operations. This versatility of implementation is reflected in the achievement of the extensive System/360 features in the comparatively low priced Model 30.

Features

Table I enumerates several features of the System/360 processors — features which are shared by all processor models within the framework of compatibility.

Conceptual Data Flow

A conceptual System/360 data flow diagram (Figure 1) illustrates, at the level of defined compatibility, the basic function of all the processors. Elements of data flow such as the instruction counter, storage address register, and instruction register are employed in the conventional manner. The 16 general registers have multiple functions and are key elements in address manipulation. These registers are used as:

- fixed point binary accumulators
- indexing registers
- sources for addresses
- operands in shifting and logical operations

The arithmetic and logical unit provides control for fixed-point binary operations in the general registers, floating point operations in the four floating point registers and variable field length logical and decimal operations in core storage.

TABLE I

SYSTEM/360 CHARACTERISTICS

- 8 bit alphanumeric coding
- Core storage addressed by character
- Signed two's complement fixed point binary arithmetic
- 16 general purpose registers
- Interruption control features

- Multiplexor I/O channel
- Up to two levels of indexing
- From two to six Selector channels available (overlap channels)
- Non stop operation
- Extensive set of arithmetic and logical commands

Instruction formats in lengths of two, four and six 8-bit "bytes" provide the ability for register-to-register, storage-to-storage, register-to-storage, and immediate-to-storage operations.

Points of Variation

Figure 1 (a) indicates the sections of this conceptual data flow wherein major design decisions were made by the development engineers responsible for each of the System/360 processors. As indicated, these decisions include:

- width and speed of data busses and registers.
- width, speed, and maximum addressable size of main core storage.
- speed of arithmetic and logical unit.
- implementation of floating-point and general registers: as an adjoint to main core storage, as a faster block of core storage, or by the employment of transistor registers.
- instruction register width. A form of "look ahead" can be accomplished in some processor models by collecting several instructions in alternate instruction registers.
- method of data flow control. Gating between registers is accomplished conventionally via transistor logic or by micro-programmed control as sequenced and determined by a read-only storage program.

MODEL 30 PROCESSOR

The IBM System/360 Model 30 processor is the smallest member of the new family. It is intended for market areas currently served by systems in the 1401, 1440 and 1620 class, as well as new market areas not previously able to profitably employ stored-program computing systems.

Model 30 — Points of Variation

We can best provide an overview of model 30 characteristics by reference to the System/360 data flow diagram and by noting the design decisions made for purposes of Model 30 development (Figure 1(b)):

- The data buss (and related registers) is 9 bits wide, providing transfer for 8 data bits plus a parity bit.
- The main core storage is 8 bits wide (plus parity). It is available in four sizes: 8K, 16K, 32K and 64K bytes. Address calculations are performed on the basis of 24 bits of address, consistent with System/360 design, but only 16 bits are actually used in the storage address register. Main core storage is operated at 2 microseconds for read/write cycles and 3 microseconds for read/compute/write cycles.
- The arithmetic and logical unit consists of an adder/logical unit capable of handling two 8 bit inputs at a time. Thus, during packed decimal arithmetic operations for example, two pairs of decimal digits from the two data fields are operated upon at the same time, yielding a two digit result byte for each cycle of the operation.

- Floating point and general registers are accomplished as an augment to main core storage. Additional registers peculiar to the Model 30 function are contained within this 512 byte core storage augmentation.
- The instruction register is 8 bits wide, allowing serial-by-byte read out, interpretation, and execution of instructions.
- Control sequencing is accomplished largely by means of a read-only storage. This control technique, described in a subsequent section, offers a more economical means of implementation of the System/360 instruction set and I/O facilities than could be realized with conventional hardware controls. Additionally, it offers a design flexibility which is exemplified by the IBM 1401, 1440 and 1460 compatibility features available for the Model 30. Each of these features is accomplished by micro-programmed interpretation of the applicable machine organization definition, with minimal alterations to the Model 30 data flow itself.

Data Flow Description

Figure 2 illustrates the data flow of the Model 30. There are ten 8-bit transistor registers connected to busses to and from the arithmetic and logic unit. These registers are all connected to buss A, which is a "latched-up" entry into the ALU, and three registers are connected to buss B, which is also latched up. The ALU output, Z, connects back to the inputs on each of the ten registers. The main Storage Address Register, STAR, is 16 bits wide and is entered from registers I, J and U, V and T. When T is used, the high order 8 bits are set to zero. There are additional entries into A, B, Z and STAR from other areas of the system. Transfers within the data flow are 8 bits wide (plus parity), except for transfers to STAR which are 16 bits plus parity.

Since all the registers are similarly connected to the ALU, they are general purpose and have different functions depending on the operation being performed. In general, they have the following functions:

I, J
U, V
T
} ------ address registers

L --- indicates length of data fields

D --- general purpose data register

R --- memory data register

G, S --- "stat" registers that retain machine conditions and status for testing by the control system

The ALU performs the functions of addition, subtraction, "and," "or," and exclusive "or". The input gating to the ALU allows either a 4 bit or an 8 bit entry, and the "A" buss can also criss-cross the high order 4 bits and the low order 4 bits. This is necessary in order to handle information that is inherently 4 bits wide. In the case of arithmetic, the data can be either decimal or hexadecimal (binary) and is done two digits (8 bits) at a time.

Employment of Local Storage

As discussed above, the general registers and floating point registers are contained within an augment to core storage referred to as local storage. Figure 3 illustrates this augmented core storage — the second 256 byte group is employed for working input-output unit control words when the Multiplexor channel is in operation. This area of core storage is, of course, not directly available to the model 30 programmer. Access to it is by generation of pseudo-addresses in the data flow and the employment of the microprogram to direct the information appropriately through the system and the conventionally addressed core storage.

READ-ONLY STORAGE AND CONTROL

The control for the Model 30 is designed around a read-only storage, ROS, and includes the hardware for addressing the ROS, sensing and decoding the output, and the basic clock.

Schematically, it is depicted in Figure 4.

The ROS is used as follows:

An OP Code such as ADD, when read from main core storage, is examined by the microprogram and interpretively employed in determining the function and format of the instruction to be performed. This interpretive process diverts program control to a location in ROSAR. When the 50 bit word at this location is read from the ROS, its contents are decoded to activate specified control points in the system, thus performing the first step in a sequence of steps (a microprogram) required to execute the OP Code. There are approximately 75 control points in the Model 30 data flow. The output (the addressed word) also sends back next address information, which when coupled with the branch control (which enters information relative to machine status changes) forms the address in the ROSAR for the next and succeeding steps of the sequence. The ROSAR is shown on the data flow diagram, Figure 2.

The ROS cycle time is 1 microsecond and the basic clock is a 4 position ring of 250 nanoseconds per step. The output word of 50 bits is subdivided into 14 control fields, each of which controls a specific portion of the system.

These 14 control fields are shown in Figure 2. They can be separated into three broad function groups: "Branch Control", "Function Control" and "Storage Control".

The names of these fields are:

Name	# of Bits	Description
Branch Control		
CN	6	Next ROS address
CL	3	ROS branching
CH	4	ROS branching
CS	4	Stat Set
Function Control		
CA	4	Source for A
CF	3	A input Hi/Lo, Crossed/Straight
CB	2	Source for B
CG	2	B input Hi/Lo
CK	5	Constant generator
CC	3	Carry
CV	2	T/C ALU control
CD	4	Destination
Storage Controls		
CM	3	Address register/inhibit select
CU	2	Local storage/data destination storage control

47 + 3 Parity bits = 50 bits

Branch Control

The Branch Control fields (CN, CL, CH, CS) provide the address of the next ROS word to be executed. An ROS address is a 12 bit binary number, the high order bit of which is 0. Normally, the branch control group provides only 8 bits (leaving 4 bits unchanged) of next address information. Of these 8 bits, the low order two are called "branch" bits and the remaining 6 are called "next address" bits. The 6 "next address" bits are specified directly in a 6-bit-field CN. The two "branch" bits are specified by a 3-bit-field CL, and a 4-bit-field CH. These two fields are decoded and used in masking and extracting machine conditions and status conditions contained in the data flow registers G and S.

One other function is included in the Branch Control group. This is the function of setting the variables $w_1 \ldots w_n$ and $(x_n + 1, y_n + 1) \ldots . (x_{31}, y_{31})$ to desired values for later use in microprogram branching. This function is controlled by a field CS, containing 4 bits.

In summary, every ROS word provides a branching ability. The branch can be a 4-way branch, a 2-way branch, or a 1-way branch (simple next address). A partial next address is normally used, but provision is made for obtaining a full 12 bit next address when required. The total length of the Branch Control group is 17 bits plus 1 parity bit.

Function Control

The Function Control group is subdivided into four fields: Source A, Source B, Operation, and Destination.

Source A (CA) This 4-bit field selects one of the 10 hardware registers to be gated to the A input of the ALU.

(CF) The 8 data bits from a register can be presented to the A input "straight" or they can be presented "crossed". The term "crossed" means that the high four bits of the source register enter the low order four bits of the ALU, and the low order four bits of the source register enter the high order four bits of the ALU. The A input can further be controlled by presenting all eight bits, the low order four only, the high order four only, or none, to the ALU. This is called HI/LO gating. When a given four bits are not permitted to reach the ALU, the ALU "sees" zeros on those input lines. The HI/LO gating occurs after the straight/crossed control so the terms HI/LO refer to the actual ALU inputs and not to the source register.

Source B (CB) This 3-bit field selects one of three registers to be presented to the B input of the ALU. The B input is the "True/Complement" input and has HI/LO controls but no straight/crossed controls.

(CG) This field controls the HI/LO gating of the B input to the ALU. That is, the low order four bits only, the high order four bits only, all eight bits or none of the eight bits of B may be presented to the ALU.

Constant Generator (CK) This field is gated to the B buss, main core STAR and ROSAR, thus providing a source for constants, mask configurations, and address constants.

Carry (CC) This three bit field controls carry in, carry out, AND, OR, XOR functions and permits the setting of carry out into the carry latch, if desired.

True/Complement & Binary/Decimal Control (CV) This two bit field controls the true/complement entry of the B input to the ALU, also whether the operation is decimal or binary.

Destination (CD) This four bit field selects one of the 10 hardware registers to receive the output of the ALU. A given register may be used both as a source and as the destination during a single ROS cycle.

In summary, the Operation group specifies one of ADD binary, ADD Decimal, AND, OR, or EXCLUSIVE OR. It also specifies True or Complement; 0 or 1 carry input; save or ignore resulting carry; use True/Complement latch; and use carry latch.

Storage Controls

(CM) (CU) These two fields control core storage operation. Either main storage, local storage, or MPX (for I/O) storage can be addressed for storage read/write calls; five values of CM are used to specify the address register to be gated to STAR.

An example of the sequence of ROS control is shown in Figure 5 in which an "add" cycle is illustrated using a simplified data flow.

Step 1 - As the routine is entered, the contents of UV are gated to the STAR, a read call is issued to main storage and register V is decremented by 1.

Step 2 - The A field data is regenerated in storage and the A field data byte is transferred from register R to D.

Step 3 - The contents of IJ are gated to STAR, a read call is issued and J (lower 4 bits) are put on Z.

Step 4 - Z is tested for zero to set up the branch condition at the next step, the B field data byte is read out (R) to the adder, as are the D register contents (A field data) and the carry from a previous cycle. The output (Z) is gated into R.

Step 5 - If the Z zero test in Step 4 is true, a Write into the B field is performed (the address is still in STAR), J is decremented by 1 and the routine is repeated. If the Z zero test in step 4 is false, then a branch is made to the Write call and the routine is exited.

AN EXAMPLE OF ROS CONTROL — INPUT/OUTPUT CHANNELS

System/360 input/output is accomplished in the same manner for all processors. The design intention in this respect was that a given input/output device could be designed for attachment to a defined channel rather than to a defined processor. Coupled with the instruction set compatibility, this device interchangeability results in processor interchangeability — any of five processors can act as, or replace, the processor portion of a total system.

System/360 Channels

While they are identical in their appearance to the input/output device, and to the program, two channel concepts are designed for processor implementation: the selector channel and the multiplexor channel. Figure 6 illustrates the basic difference between the two: the selector channel employs a dedicated segment of the processor to provide full-time service to that device which is currently selected. Hence, if a byte requests entry, the processor is allowed to employ one cycle of main core storage time in accepting that byte and adjusting count and address registers accordingly.

Multiplexor Channel

The multiplexor channel offers an economical approach to the same function — the processor registers normally employed in internal data transfer and manipulation are saved, borrowed, employed in input byte introduction (or output) and then restored to their initial condition for continuation of internal processor functions. In this manner, the effect of 32 independent channels can be accomplished for slower serial input and output devices.

The multiplexor channel is particularly interesting to this discussion of micro-programmed control in that a large part of the control sequencing for the subchannels is accomplished via the read-only storage of the processor.

Each of the 32 subchannels has a unit control word of 8 bytes in the reserved section of main core storage described previously. The format of the unit control word is illustrated in Figure 7.

A "Start I/O" instruction encountered in the execution of a conventional program causes the following sequence of micro-programmed events to occur in initiation of the required operation:

1. The Channel and Unit Address bits from the Start I/O Instruction are interpreted and it is determined that the unit address is one of those on the Multiplexor Channel.

2. From the specified Unit Address, the address of the corresponding Unit Control Word in the MPX storage is developed by the ROS micro program.

3. The "Status" Byte and "OP-Flag" byte of this UCW are then examined to determine the availability of the addressed unit. If there is no indication that the unit is in use, access is made to the Channel Command Word at a location specified in the Command Address Word (bits 8 - 31) at locations 72 - 75. This address is that of the OP byte of the Channel Command to be performed and this OP is sent to the selected unit, in a sequence of control signals and answering responses over the interface.

4. At the completion of this selection sequence, the Channel is able to recognize, from the responses of the Unit, if there are any improper conditions which prevent the required operation from being initiated. If not, it proceeds to generate and store a new 64-bit Unit Control Word for the unit. The UCW is stored in eight sequential byte locations in MPX storage.

5. During the formation and storage of a UCW, the UCW status byte is set to zeros, the CCW OP and Flag bytes are merged and stored as the UCW OP byte, and the count, data address, and next command address fields are extracted and set into their corresponding locations in the UCW. Since these latter three fields are shorter than their Channel Command Word and Start I/O counterparts, the microprogram checks to assure that no significant count or address bits are present above 16 low-order bits (Model 30 physical addresses are 16 bits). Any error detected by the microprogram in the course of the execute half of the Start I/O Instruction is registered by setting the condition bits of the active Program Status Word. In the event that such an error is detected, the I/O operation is not performed and the program continues to the next CPU instruction. If there are no detected errors, the Condition Register is set to zero and the operation is initiated by the addressed I/O Unit. The program then continues to the next instruction while the I/O Unit is getting underway.

6. The same basic sequence of events occurs for a Start I/O instruction whose unit address specified a Selector Channel I/O Unit. The difference is that instead of testing and loading a UCW, the Start I/O microprogram tests and loads the Selector Channel Hardware which serves the same purpose as a UCW.

7. In the case of high speed devices attached to the multiplexor channel, the CPU registers are borrowed at this point and employed in "burst mode" introduction or output of an entire data record.

Multiplexor Channel Share Cycle

Once any "byte mode" operation has been initiated on the Multiplexor Channel by the Start I/O Instruction, further communication with the Channel required to handle data, error, ending, or attention signals is carried out by a system of requests and responses initiated by the device requiring service. Requests by devices are polled by the standard Interface Select Lines. Whenever an operating device requires service, it waits for the Select Out polling signal to rise at its adapter. It then takes over the Interface lines by raising the Operational In line, places its address on Buss In, and raises Address In. This condition is detected at the Channel as a request for service originating from the unit itself, i.e., a Share Request. The detection of a Share Request by the Channel circuits causes an automatic interruption of the CPU microprogram at the beginning of the next ROS microprogram step not involved in the control of an already initiated read/write main storage cycle. The address of this CPU "next micro-word" is automatically stored in a "Back-up ROSAR" (CW & CX) and a fixed address is forced into the Main ROSAR. This address is used as the start of the I/O Share microprogram routine which determines the kind of service the unit requires and acts to provide that service.

In order to provide an I/O Channel function at the lowest possible cost, the Multiplexor Channel uses the CPU data flow hardware for handling all data and address manipulations, etc., involved in its operation. At the start of any Share microprogram, the CPU registers contain information which is essential to the continuance of the displaced CPU microprogram. Before the CPU registers may be used, therefore, their contents must be stored away to preserve them for restoration when the Multiplexor Channel operation is completed. The Share routine can be divided into five parts (Figure 8):

1. Temporary storing of CPU Registers R, T, L, P, U, V, G, and S in CPU Local storage.

2. Development of the appropriate Unit Control Word Address in MPX storage and read-out of the UCW information to the CPU registers.

3. Creating the data access to main storage, updating the UCW count and working address in the ALU.

4. Restoration of updated UCW information back into the MPX storage area.

5. Read-out of the original CPU register contents from temporary storage and resumption of processing.

The Share routine also contains the sequence controls for performing other channel functions, such as data chaining and command chaining.

A microprogram in the ROS also provides these functions for the Selector Channels on the Model 30.

Time Sharing

The data flow and the main core storage of the Model 30 are time shared among several functions as should be clear from the preceding descriptions. This concept is exploited in the design of (1) the Selector Channels, (2) the Multiplexor Channel, (3) the Interval Timer Operation, (4) the Sequence Control and, (5) the Program Interruption Control. Basically, the allocation of storage cycles follows the following priority:

1. Selector Channel 1 (data requests)
2. Selector Channel 2 (data requests)
3. Selector Channel 1 (chaining requests)
4. Selector Channel 2 (chaining requests)
5. Multiplexor Channel (data or chaining)
6. Interval Timer Update Cycle
7. CPU requests.

SUMMARY

The employment of read-only storage control has been reviewed as an approach toward the accomplishment of defined system/360 functions in the Model 30 processor. The micro-programmed interpretation of instructions and multiplexor channel functions has provided sophistication of system/360 concepts at the small processor level. This is considered a significant advance in small processor design implementation.

ACKNOWLEDGEMENTS

The authors are indebted to the following members of the engineering team which was largely responsible for conception, design and development of the Model 30 implementation described herein:

Arthur F. Collins

Dr. William P. Hanf

Albert A. Magdall

E. Robert Marsh

Charles B. Perkins, Jr.

Dr. John W. Rood

List of Figure Captions

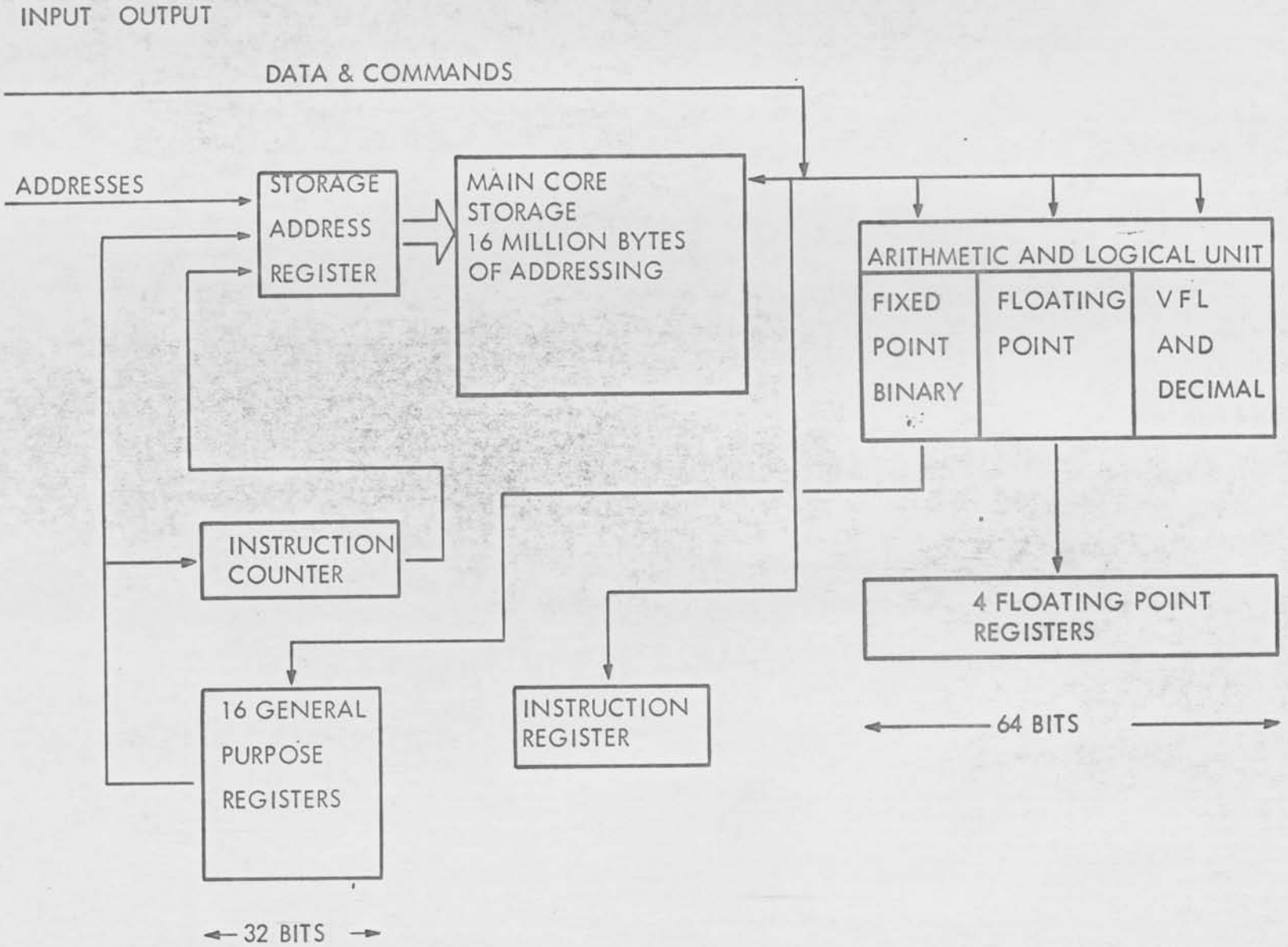


Fig. 1 — System/360 Data Flow

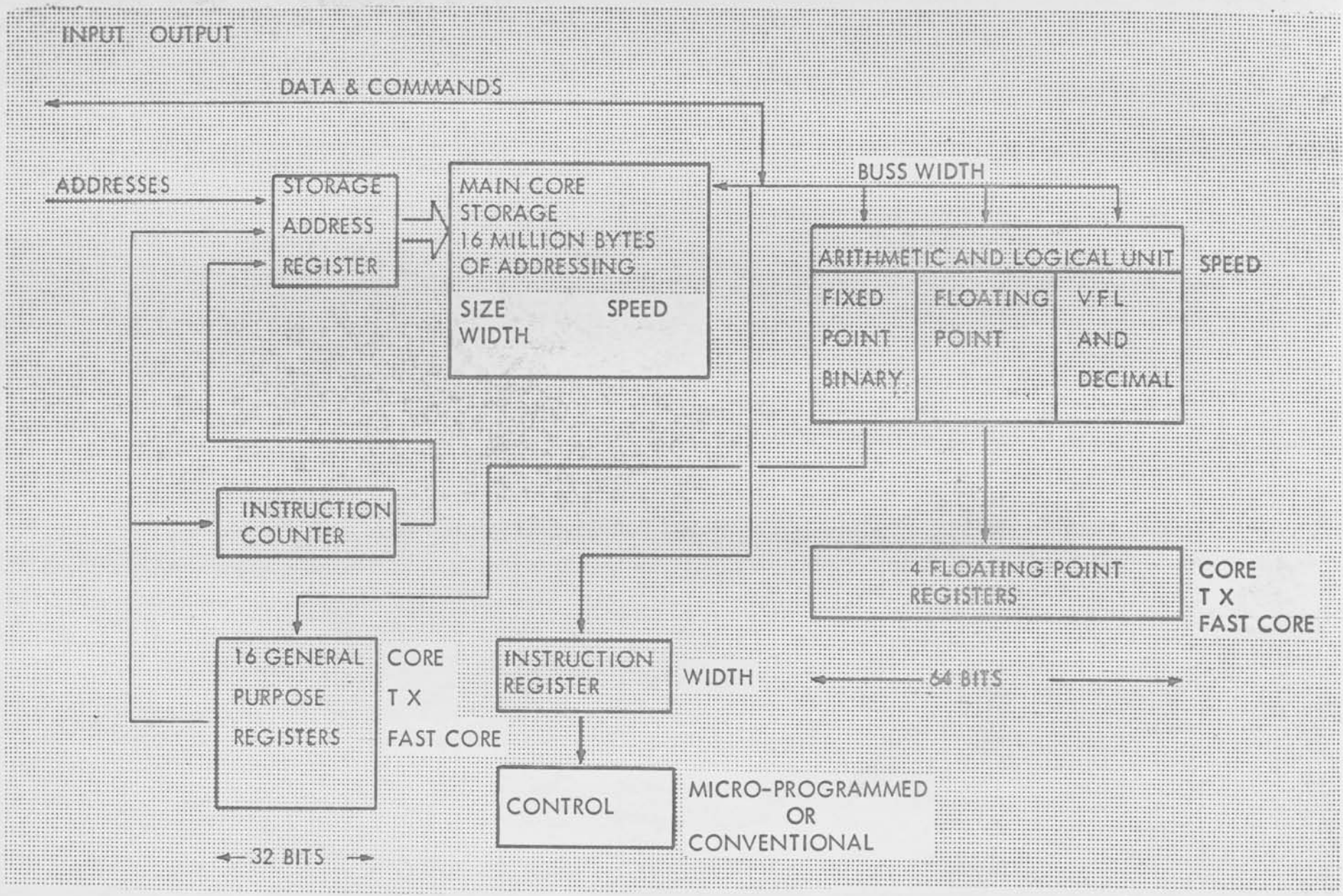


Fig. 1 (a) — Data Flow Diagram Indicating Design Decisions for All System/360 Processors

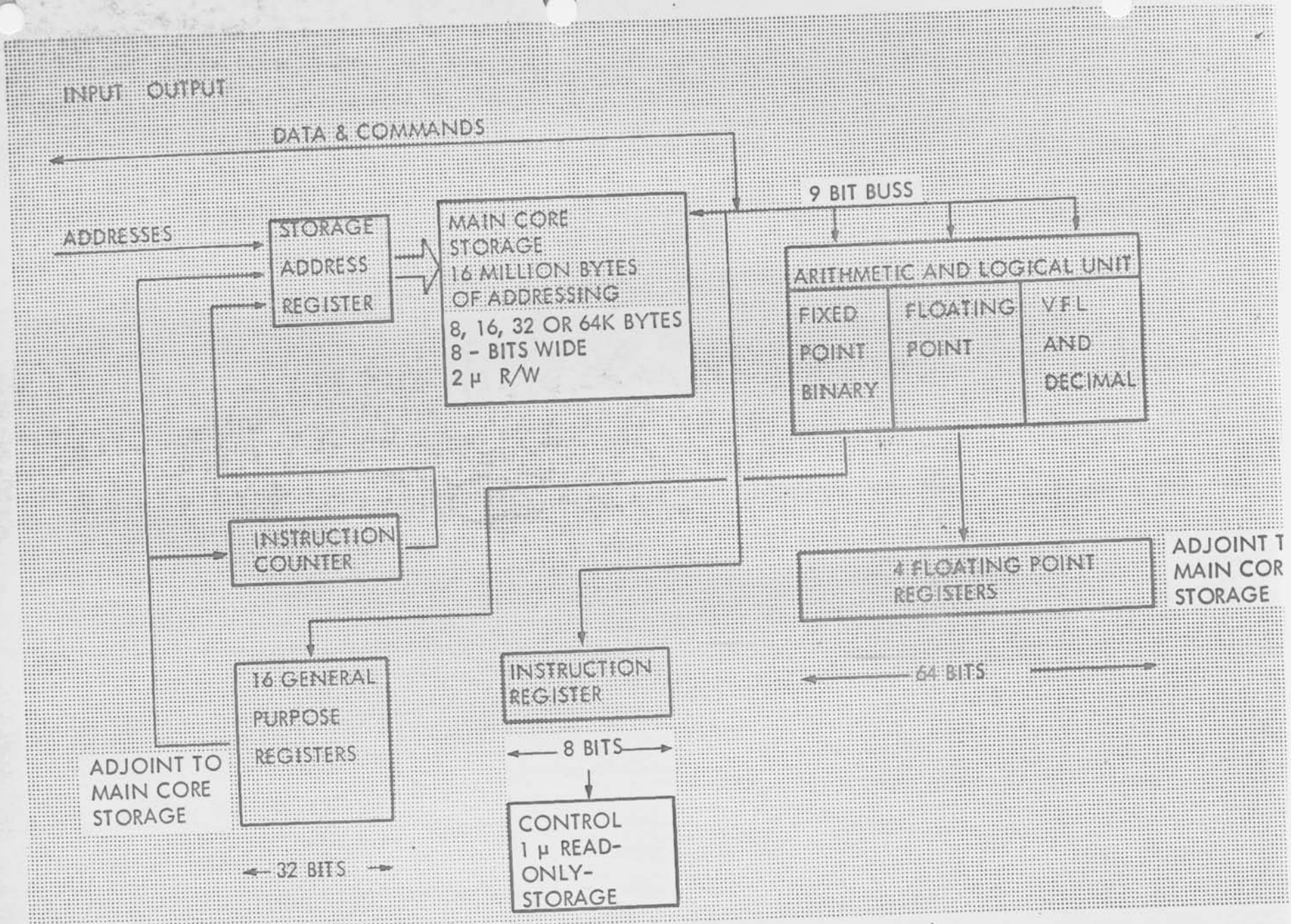


Fig. 1 (b) — Data Flow Diagram Indicating Model 30 Design Decisions

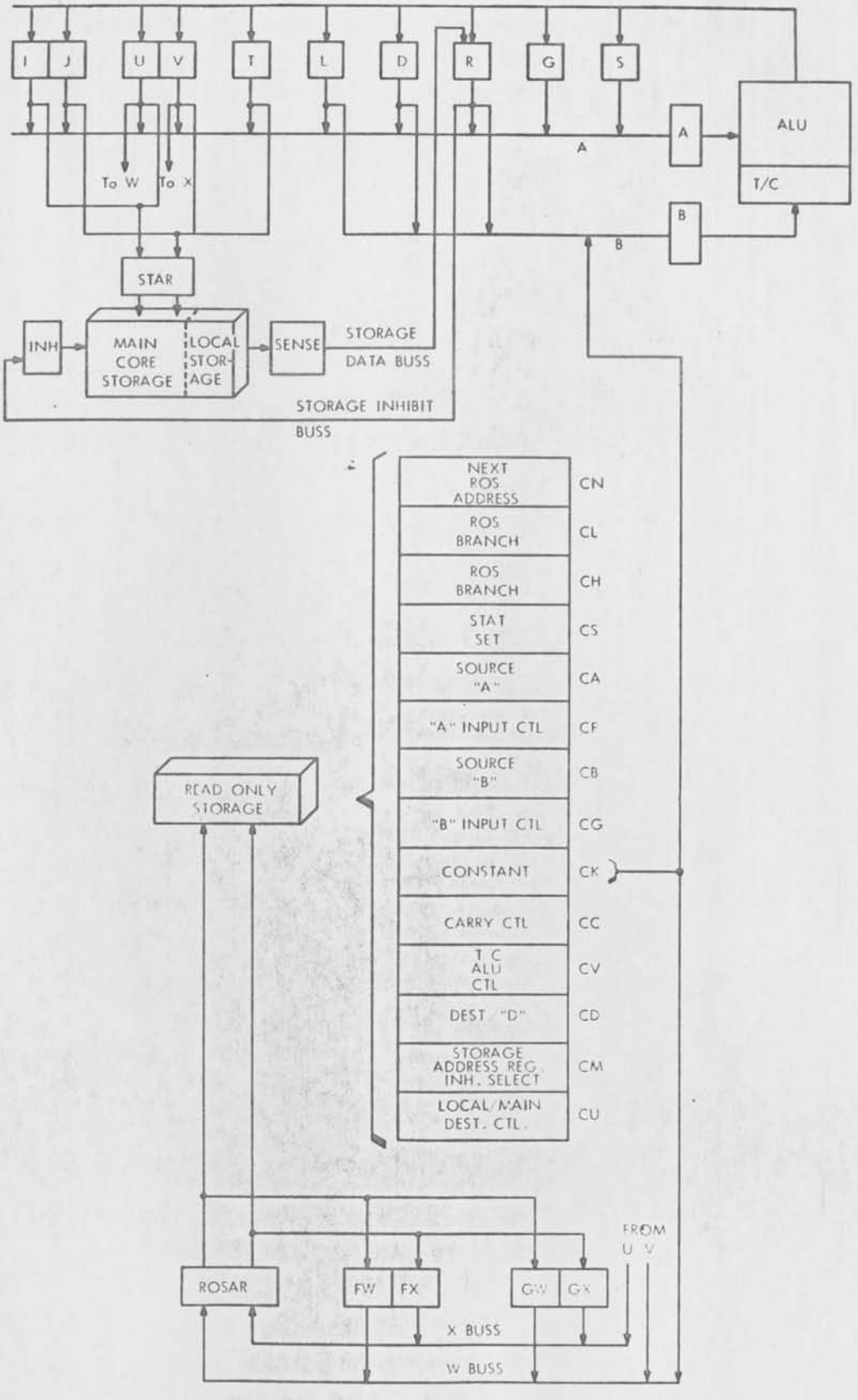
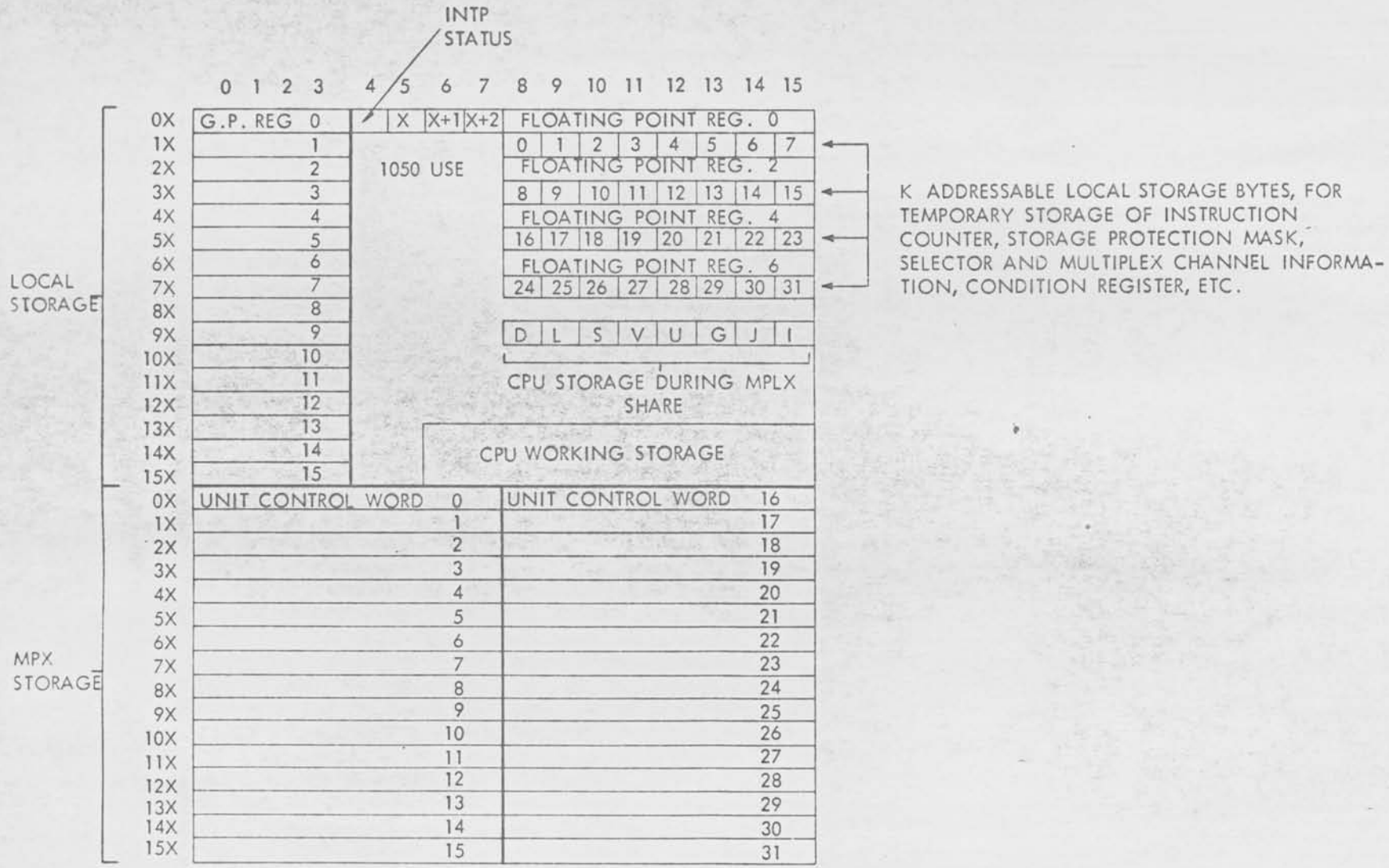


Fig. 2 — Model 30 Data Flow



MODEL 30

Fig. 3

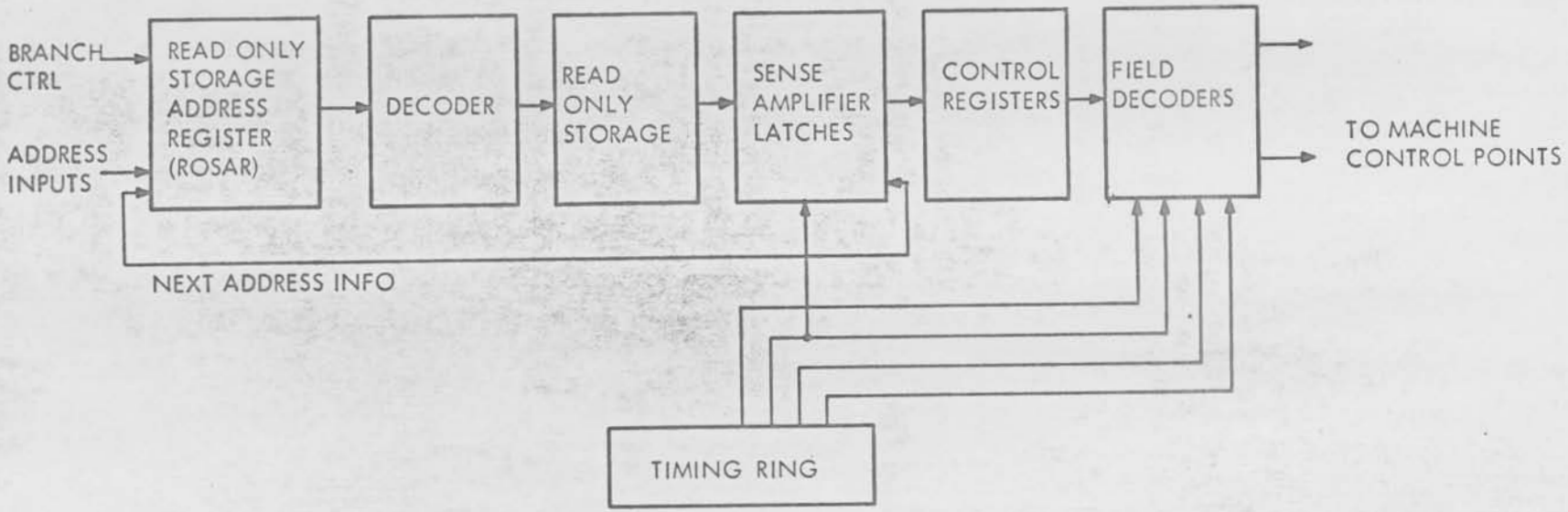


Fig. 4 — Read-Only Storage Schematic

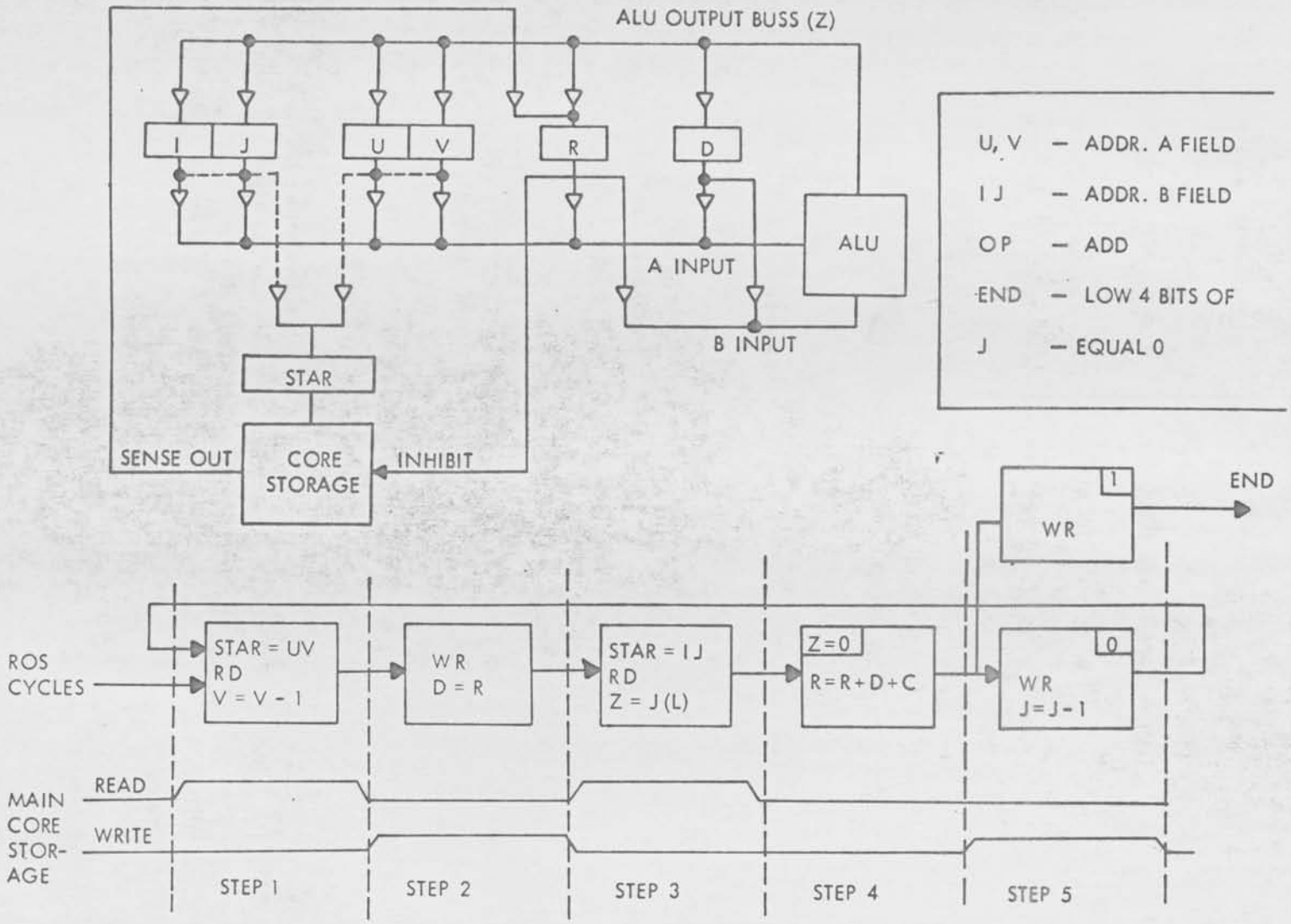
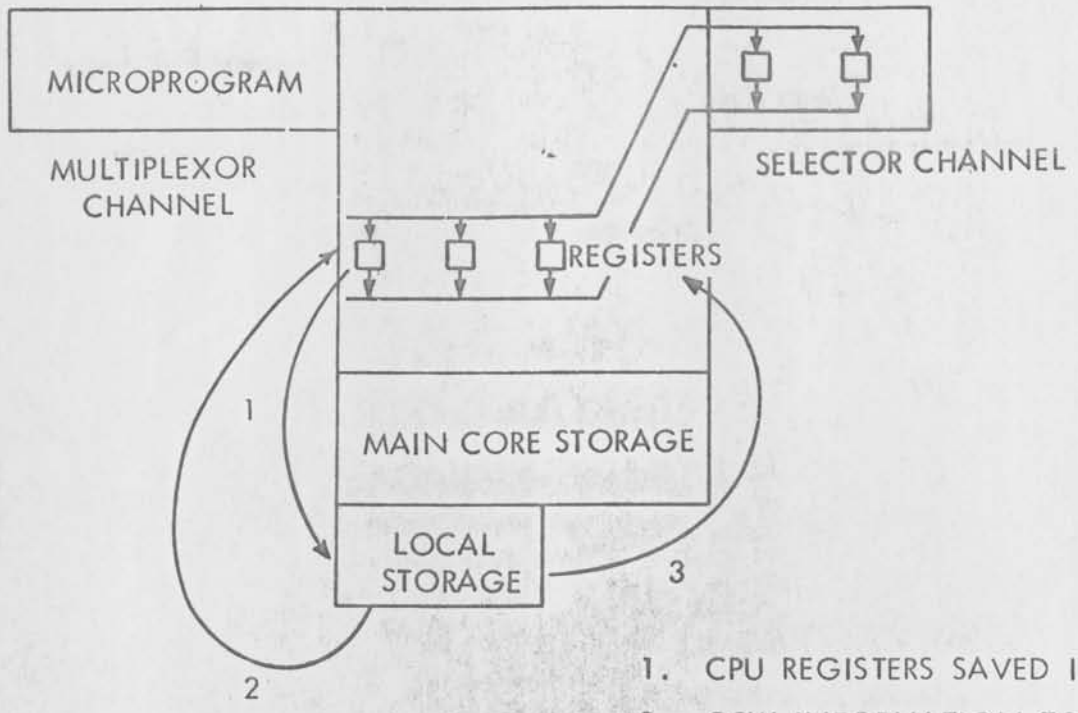


Fig. 5 — Example of ROS Control Sequence



1. CPU REGISTERS SAVED IN LOCAL STORAGE
2. CCW INFORMATION TO CPU REGISTERS
3. RESTORE CPU REGISTERS

Fig. 6 — Illustration of Channel Functions

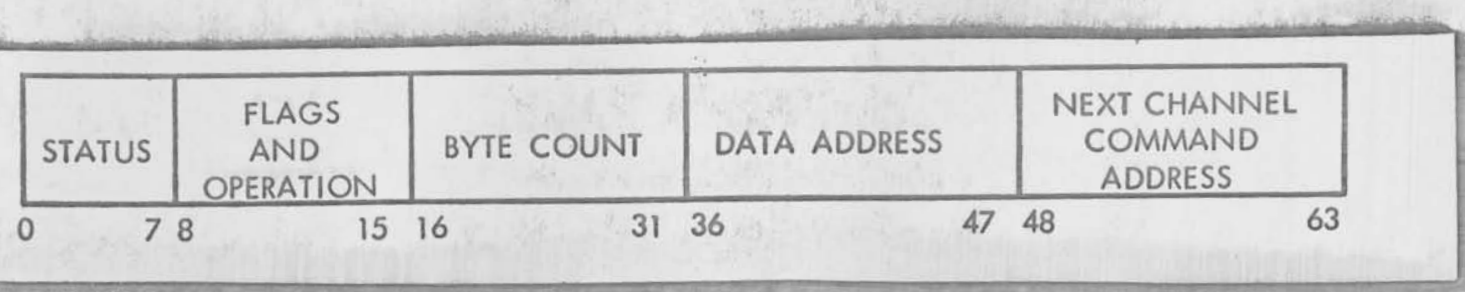

STATUS
FLAGS AND OPERATION
BYTE COUNT
DATA ADDRESS
NEXT CHANNEL COMMAND ADDRESS
0
7 8
15 16
31 36
47 48
63

Fig 7

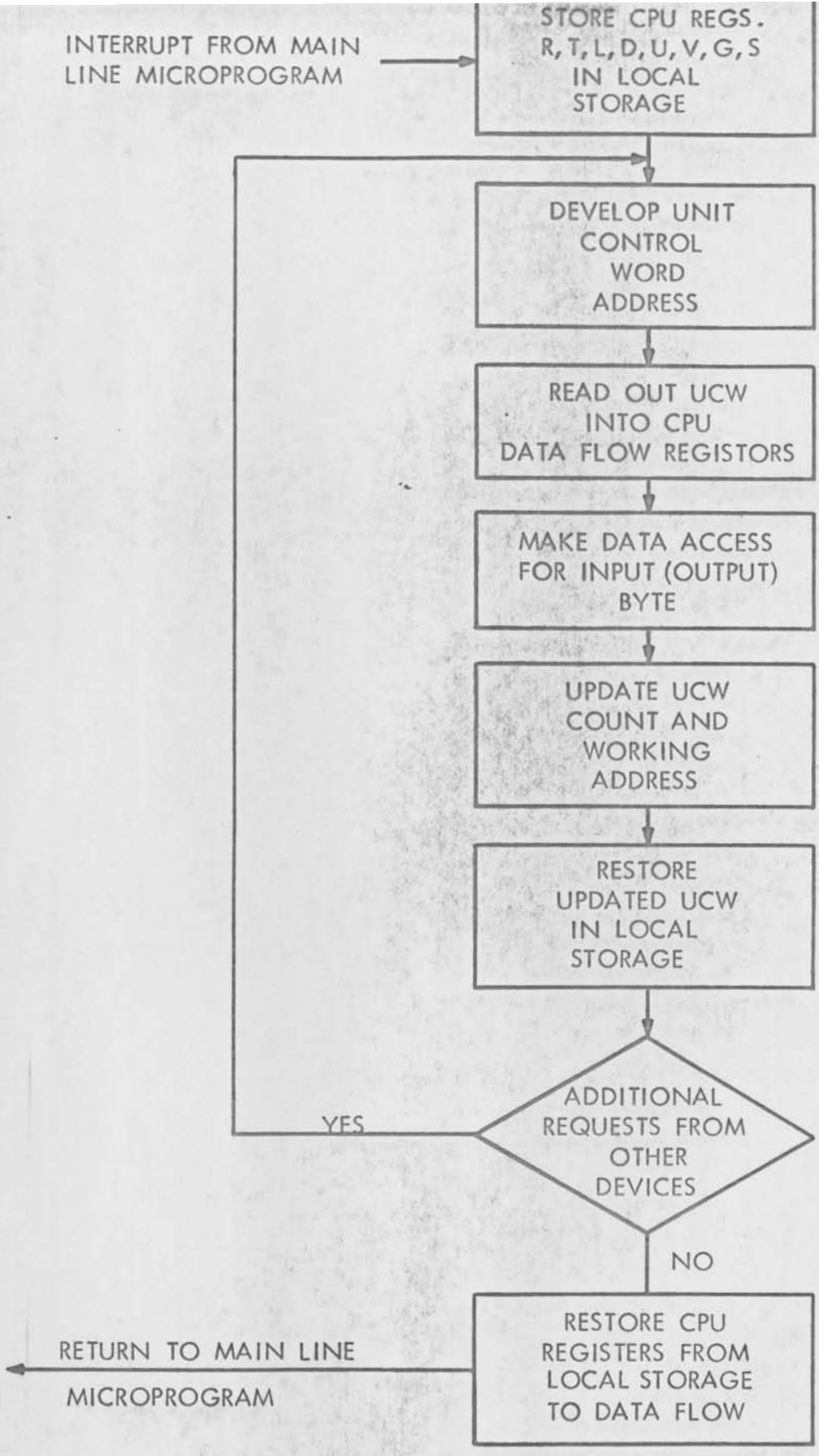


Fig. 8 — Flow Diagram of Multiplexor Share Routine