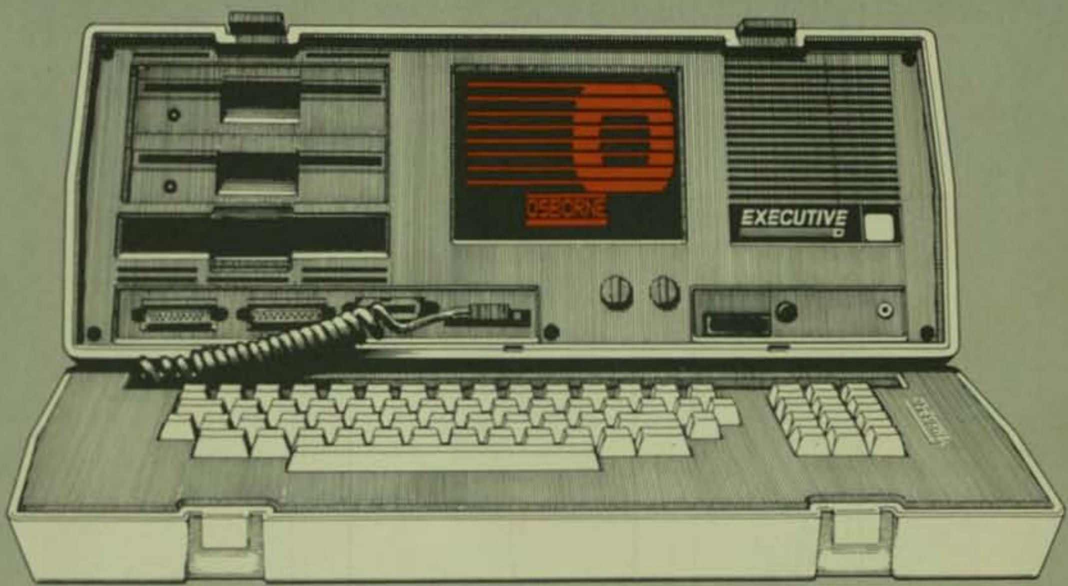
EXECUTIVE™

OSBORNE

Technical Manual



Volume A: Hardware and Software Fundamentals

OSBORNE™
COMPUTER CORPORATION

OSBORNE EXECUTIVE TECHNICAL MANUAL
VOLUME A: HARDWARE AND SOFTWARE FUNDAMENTALS

ABSTRACT

This manual summarizes the primary hardware and software elements of the Osborne Executive computer. The material is organized to provide a suitable foundation for understanding the remaining four volumes in this five-volume series.

OSBORNE EXECUTIVE TECHNICAL MANUAL
Volume A: Hardware and Software Fundamentals
Issue date: May 1984
Part Number: 2F00213-01

Volume A was written by Mike Iannamico. Roger Gottlieb offered guidance and sage advice for this entire series. Editorial assistance was provided by Stephanie North.

FEDERAL COMMUNICATIONS COMMISSION
RADIO FREQUENCY INTERFERENCE STATEMENT

WARNING: This equipment has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of Part 15 of FCC rules. Only peripherals (computer input/output devices, terminals, printer, etc.) certified to comply with the Class B limits may be attached to this computer. Operation with non-certified peripherals is likely to result in interference to radio and TV reception.

INSTRUCTIONS TO USER: This equipment generates and uses radio frequency energy and if not installed properly, i.e., in strict accordance with the operating instructions, reference manuals, and the service manual, may cause interference to radio or television reception. It has been tested and found to comply with the limits for a Class B computing device pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference when operated in a residential installation.

If this equipment does cause interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

o Reorient the receiving antenna.

o Relocate the equipment with respect to the receiver.

o Move the equipment away from the receiver.

o Plug the equipment into a different outlet so that equipment and receiver are on different branch circuits.

If necessary, consult your dealer service representative for additional suggestions.

The manufacturer is not responsible for any radio or TV interference caused by unauthorized modifications to this equipment. It is the responsibility of the user to correct such interference.

CAUTION: This product is equipped with a UL listed and CSA certified plug for the user's safety. It is to be used in conjunction with a properly grounded 115 Vac receptacle to avoid electrical shock.

PREFACE

The five volumes contained in the Osborne Executive Technical Manual examine the Executive computer from a variety of perspectives.

Volume A: Hardware and Software Fundamentals explains the operation of the computer by relating the hardware elements to the firmware and software routines. The information level is appropriate if you wish to gain an overall understanding of the major components of the system.

Volume B: Programmer's Guide describes the manner in which CP/M Plus is implemented on the Osborne Executive. Programmers can find the best means for using the powerful features available through CP/M Plus.

Volume C: Hardware Design and Theory of Operation explains the architecture and circuit operation of the Osborne Executive down to the chip level. Each of the circuit board assemblies contained within the cabinet is discussed. Timing diagrams and schematics are also included.

Volume D: Integrated Circuit Specifications describes the primary integrated circuits used in the Osborne Executive. Details of interest to both programmers and technicians are provided.

Volume E: ROM 1.21/BIOS 1.1 Source Code Listing is a complete listing of the routines contained within the Osborne Executive's ROM and the software BIOS routines. The comprehensive listing includes all the information necessary to understand the computer's functions at the lowest possible level.

Together these five volumes comprise one of the most complete technical references available for any microcomputer on the market today. We hope the material provided enables you to fully understand and utilize all the possibilities offered by your Osborne Executive.

VOLUME A: HARDWARE AND SOFTWARE FUNDAMENTALS

FIGURE LIST

FIGURE LIST (cont.)

TABLE LIST

SYSTEM OVERVIEW

The Osborne Executive is a powerful self-contained microcomputer designed around the Z80A microprocessor. The standard system has 128K bytes of main memory and is equipped with a built-in monitor, a keyboard, two disk drives, and a row of connectors for peripheral interface. Included with the computer are two operating systems, three application programs, and two variations of the BASIC programming language.

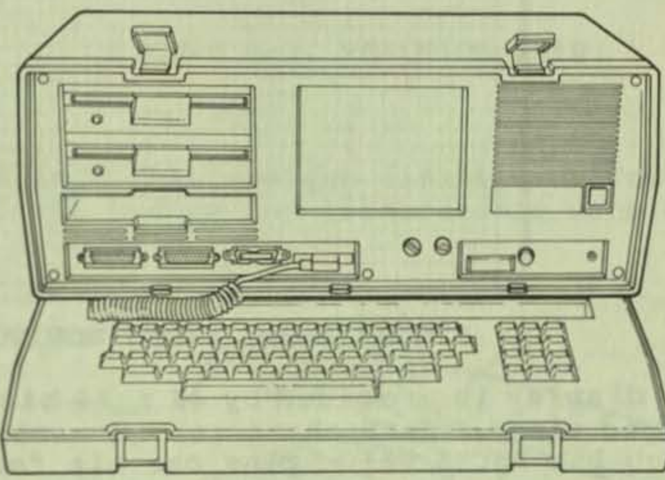


FIGURE A-1. OSBORNE EXECUTIVE

Central Processing Unit

The Z80A, an 8-bit NMOS microprocessor, fulfills the role as the system Central Processing Unit supplying the necessary control signals, addresses, and data manipulations to regulate and coordinate the operations of the Executive. The Z80A is a register-oriented CPU with internal registers that contain 208 bits of programmable read/write memory.

Memory

Main Memory for the Executive computer consists of 128K bytes of dynamic "Random Access Memory" (RAM), 8K bytes of "Read Only Memory" (ROM), and an additional 2K bytes of RAM providing memory space for system variables and "scratchpad use." Effective use of memory is accomplished though software bank switching. This allows the microprocessor to address more memory than would otherwise be possible.

Console

The Executive is equipped with a 7-inch amber monitor which has an 80-column by 24-row character display. Brightness and contrast are adjusted by controls located on the front panel.

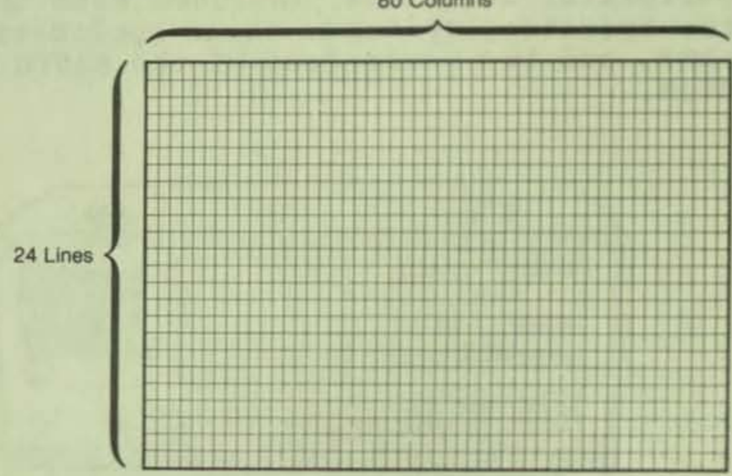


FIGURE A-2. VIDEO MONITOR

The display is provided by 4K x 12 bits of memory-mapped video. Each character is associated with a seven bit ASCII value plus one bit for each of five screen attributes. Two sets containing 128 characters each are stored in the character font and defined through the CHARGEN utility program.

The Executive console functions are patterned after the Televideo 912 and many of the same screen control characters are used. Configurable tables and pointers allow user defined keyboard and screen functions. The Executive is also capable of emulating the console functions of a wide range of popular terminals through universal terminal emulation software.

The detachable keyboard has 69 keys and a 12-key numeric keypad. The numeric and arrow keys also serve as programmable function keys which are software definable through the SETUP utility. Through alteration of the keyboard tables, all of the keys can be specifically defined by the software.

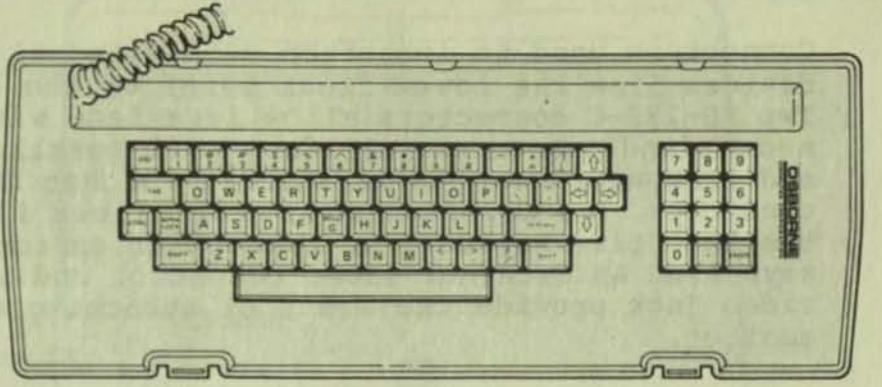


FIGURE A-3. FUNCTION KEYS

Disk Drives

Two 5 1/4 inch, half-height diskette drives provide an additional medium for transferable data storage.

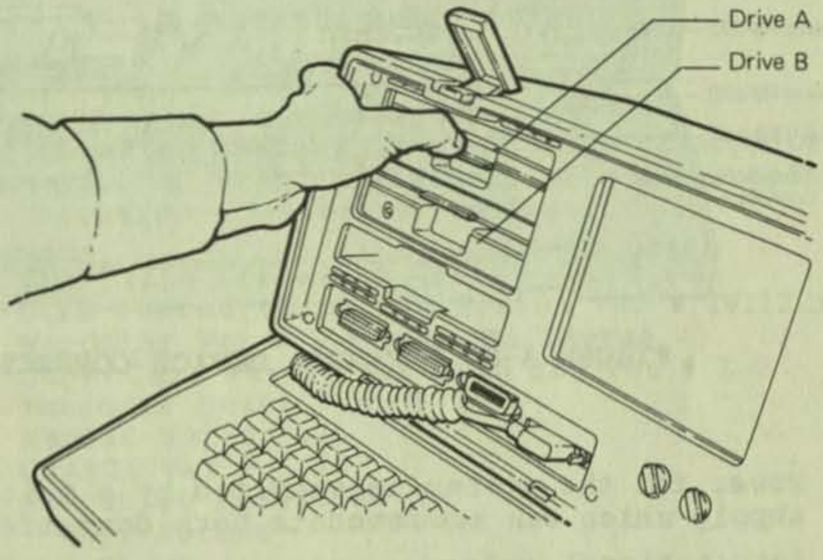


FIGURE A-4. DISKETTE DRIVES

These drives store and retrieve data from single-sided, double-density floppy diskettes which can handle up to 200 Kilobytes of information. The drives can also read the formats of a number of other computers. A compartment beneath the diskette drives can be used to store diskettes.

Connectors

Connectors used to interface with external peripheral devices line the lower front bezel of the Executive. Two RS-232-C connectors allow interface with serial modems and printers. Interface with parallel devices and IEEE equipment is maintained through the IEEE 488 connector. The keyboard connector allows input from the Executive keyboard or some other customized keyboard. An external video connector and a composite video jack provide two means of attaching an external monitor.

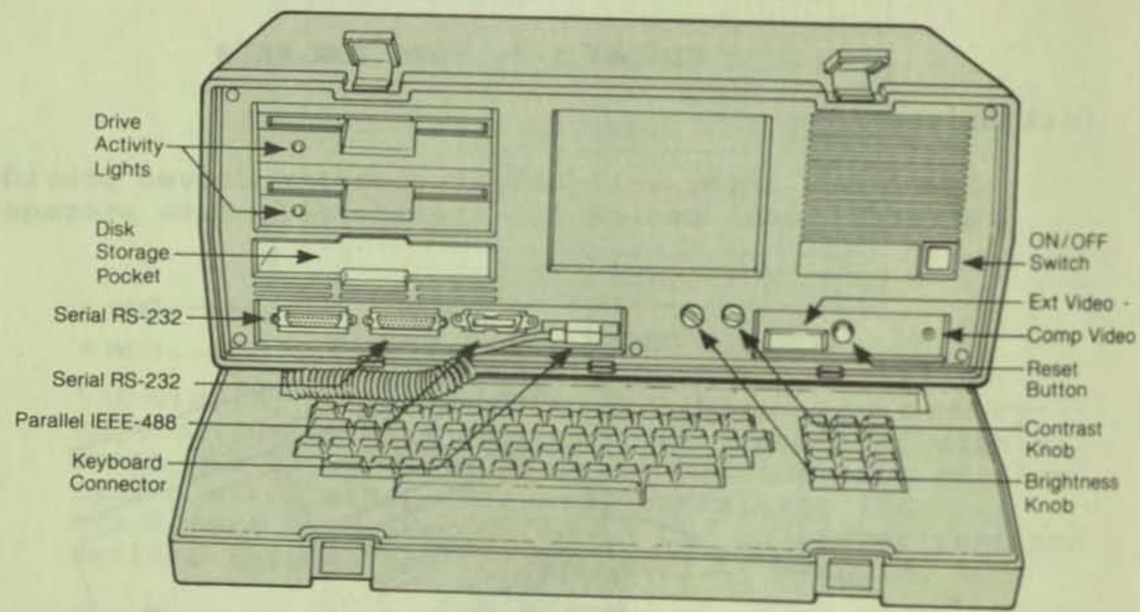


FIGURE A-5. EXTERNAL DEVICE CONNECTORS

Power

Power for the system is supplied by a switching power supply which can accommodate both domestic and international voltages.

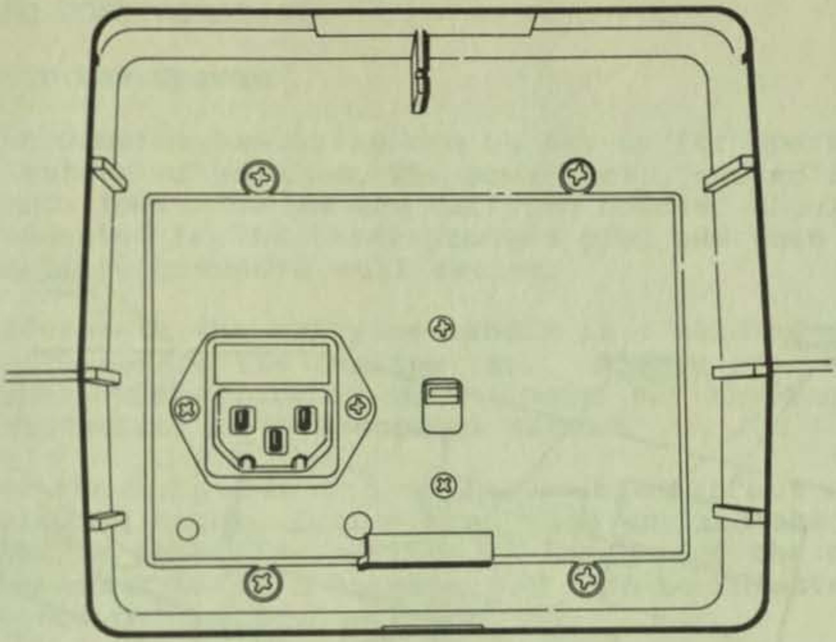


FIGURE A-6. POWER WELL

Software

Two operating systems, a wide range of application software, and preparatory utilities are supplied with the Executive computer as follows:

CP/M Plus Operating System Ver # 3.0
UCSD Pascal Operating System Ver # IV.12.B
WordStar Ver # 3.3 with MailMerge
SuperCalc Ver # 1.12 with SDI Ver # 1.0
Personal Pearl, Ver # 1.06
MBASIC Ver # 5.21
CBASIC Ver # 2.38
Executive Utilities
  Copy/Format
  Copysys
  Chargen
  Setup

PC Board Locations

Four circuit boards handle the majority of the circuit functions within the Osborne Executive 1: the power supply board, the monitor board, the dynamic RAM board, and the main logic board. The disk drive electronics are contained within two primary circuit boards for each drive, and two smaller specialized boards for each drive. The arrangement of these boards within the Executive's cabinet is illustrated in the following diagram.

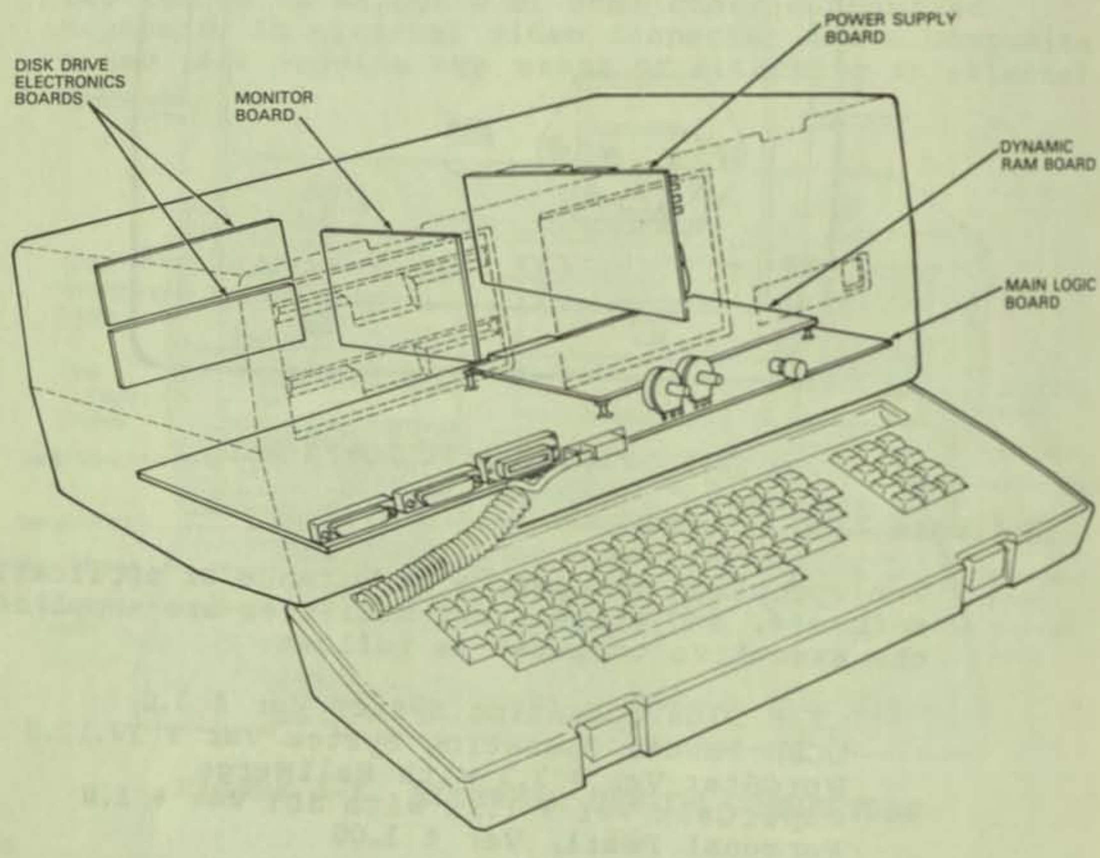


FIGURE A-7. LOCATION OF PC BOARDS WITHIN THE EXECUTIVE

OPERATING CONSIDERATIONS

Setting Up the System

The Osborne Executive can be set up for operation in a matter of minutes. The power cord, stored in a compartment beside the carrying handle, should be connected to the three-pronged plug and then into a properly grounded wall outlet.

Underneath the carrying handle is a sliding panel which covers the cooling fan. Always open this panel before turning the computer on; the fan will not operate with the panel closed.

Set the computer on a clean, stable surface with the carrying handle facing away from you and the etched OSBORNE lettering visible on the top of the case. The two black plastic latches may then be unfastened, releasing the keyboard for use.

Ventilation

A recessed fan installed on the back panel ensures that the Executive maintains proper operating temperature. The fan will operate properly only when the panel beneath the Executive's carrying handle is in the open position.

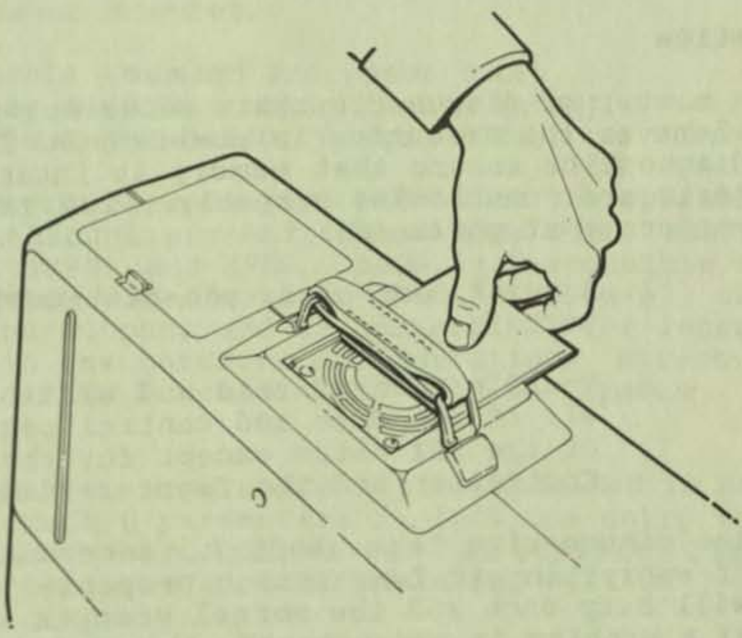


FIGURE A-8. OPENING FAN VENT

Excess heat may be generated within the Executive if any of the ventilation slots are blocked, or if the computer is placed into a cabinet that does not provide sufficient air flow. If you intend to purchase a custom cabinet for your computer or place it within any kind of homemade apparatus, ensure that ample room is allowed for air circulation within the cabinet to prevent excess heat from accumulating.

Voltage Selector Switch

A small red switch located within the power cord well allows the user to select the appropriate voltage input level for use of the Executive 1 in countries using other standards. For use domestically, always ensure that the switch is set to the 110 VAC range.

Power-On

All controls for the Osborne Executive are situated on the front panel. A blue ON-OFF switch is located on the right side of the front panel. The switch lights indicating that the system is receiving power. When powered-ON, the Executive will perform some self-test routines and then sound the operator alarm once. After a few seconds, instructions for starting the system will be displayed signifying that the computer is ready for use.

Diagnostics

A number of diagnostic tests occur automatically whenever the Executive is powered-ON. These diagnostics ensure that memory is intact and that the ports are functioning properly. Two tests are performed at power-ON:

1. A quick, one-pass, non-destructive memory test.

2. Verification of read and write operations to all of the data and control registers for all of the LSI chips except for the Floppy Disk Controller and the Counter-Timer Circuit.

The diagnostics take about 7.5 seconds to perform. If everything is functioning properly, the system will beep once and the normal prompts will appear. If a problem is encountered, the system will display the "Failed" message.

The memory test fills memory locations with a pattern of known characters, specifically AA's and 55's, then reads them back for verification. Next each address is hashed or XORed with the high- and low-order bytes of the address. The areas of memory that are tested appear in the following list:

Video RAM	(Bank 7)	hex C000 to CFFF	4K
ROM RAM	(Bank 8)	hex 2000 to 27FF	2K
Bank 0		hex 3000 to FFFF	48K
Bank 1		hex 4000 to EFFF	44K

The first area of memory tested is the video RAM which sometimes causes random characters to be visible on the screen. Next each of the remaining areas of memory are tested in turn.

After testing the available memory, the diagnostics perform tests on the LSI chips. These diagnostics consist of writing to the data and control registers of the LSI chips and then reading the data back for accuracy.

Initial Load (Cold Boot)

Before the system can perform any useful functions, an operating system needs to be supplied from a diskette. CP/M Plus is the standard operating system provided with the Osborne Executive. It consists of three major modules.

Console Command Processor (CCP)
Basic Disk Operating System (BDOS)
Basic Input Output System (BIOS)

The CCP provides an interface with the operating system through six utilities: DIR, DIRS, ERASE, RENAME, TYPE, and USER. BDOS is responsible for the file system organization. BIOS handles all of the input and output, and is responsible for interfacing BDOS with the hardware configuration. Direct system calls to BDOS functions are used for system interface.

Critical system parameters are contained in Bank 0 of memory. Bank 0 parameters include the entry to Warm Boot and to BDOS, as well as indirect links between programs and the operating system.

CP/M is loaded into memory in a three-step operation:

1. The ROM loads the CPMLDR program and stores it in Bank 0, then passes control to it.

2. CPMLDR reads the CPM3.SYS file, containing BDOS and BIOS, from the data area of the diskette into memory addresses in Bank 0 assigned by GENCPM. It then passes control to the Cold Boot system initialization routine in BIOS.

3. This BIOS routine (Cold Boot, also known as Function 0) performs the rest of the hardware initialization, displays the sign-on message, and reads the CCP from the system tracks into location hex 100. It then transfers control to the CCP or the "auto-start" program EXECST.COM if it is present. The EXECST.COM program may automatically load an application program.

   When the BOOT and WBOOT routines of BIOS get control, they initialize two system parameters in Bank 0 as shown below.

TABLE A-1. JUMP VECTORS

Location	Description
0, 1, 2	Set to JMP WBOOT (hex 0000: JMP BIOS+3). Locations 1 and 2 must contain the address of WBOOT in the jump vector.
5, 6, 7	Set to JMP BDOS, the primary entry point to CP/M 3 for other programs. The current address of BDOS is maintained in the System Control Block (SCB).

MAIN LOGIC BOARD

The Main Logic Board contains the Central Processing Unit, bus structure, a number of support chips, and additional circuitry with specialized functions. These components provide the intelligence and coordinate the operation of the video display, keyboard, diskette drives, and communications ports. RAM memory is located on a separate board and is described in a later section.

Hardware Design

The Osborne Executive hardware design is based on the Z80A microprocessor and is predicated on maintaining flexibility as well as reliable simplicity. An 8K Monitor ROM contains the boot loaders and other pre-programmed instructions; an additional 2K of support RAM resides in a jumper-configurable socket that will also accept another ROM chip. The video system has its own 8K of RAM memory while another 4K of RAM is used to store the primary and alternate character sets.

Serial interface for both the modem and printer port is provided by a Z80 SIO/2 Serial Interface chip while an 8253 Programmable Timer controls the baud rate for both serial ports. The parallel interface is provided by a 6821 Peripheral Interface Adaptor chip. An additional 6821 chip oversees input and output decoding to coordinate bus access for peripherals. Control over the disk drives is derived from a 1793 Floppy Disk Controller. Timing and control signals which ensure that events occur in an orderly fashion are provided by a 24 MHz Crystal Oscillator.

Z80A Central Processing Unit

The Z80A 8-bit NMOS microprocessor, a stable and reliable industry-standard, furnishes the intelligence and controls most on-board operations. The only circuit that can take precedence is video refresh; this is necessary to maintain an accurately updated video image.

The Z80A is a register-oriented CPU with internal registers that contain 208 bits of programmable read/write memory. There are two sets of six general purpose registers which may be used individually as 8-bit registers or as 16-bit register pairs. There are also two sets of accumulator and flag registers.

Additionally, the Z80 contains a stack pointer, program counter, two index registers, a refresh register (counter) and an interrupt register. Refer to Volume D for a complete description of the Z80A.

An internal refresh register and control circuit alternates with memory access cycles to provide RAM refresh. In the Executive computer the Z80A is operated at 4 MHz generated by the computer timing circuit based on a 24 MHz crystal oscillator.

The Z80A uses a 16-bit unidirectional address bus and an 8-bit bidirectional data bus. The CPU interfaces with the rest of the computer via an address bus, a data bus, and control lines relying on interrupts from the various components.

Control lines enable devices connected to the bus with the timing circuits exercising further control of events. The CPU essentially operates repetitiously performing standard CPU functions until an interrupt signal from a peripheral device notifies it of a requirement for service. When interrupted the CPU is instructed by preprogrammed instructions from the ROM.

The memory, disk system, video system, and the parallel and serial ports are tri-state devices or are interfaced to the bus via tri-state devices. The tri-state device, in addition to the normal high and low states, has a high impedance state which serves to reduce bus loading and simplifies addressing techniques. Significant parameters of the Z80A are:

Data Word Width:	8 bits
Instruction Width:	8 bits
Instruction Count:	158
External Address:	64K
IC Package	40 pin DIP
Power Required:	+5 V at 90 ma

SIO (Serial Input/Output)

The SIO supplies the interface capabilities for the two RS-232-C serial ports. The SIO is selected by the computer I/O decoder, which controls the selection of all I/O peripheral devices.

The actual selection of the serial ports is determined by the CPU via address lines 2, 3, 4, and 7. This produces the SIO Select signal, which is routed to the Z80 SIO chip. The chip then uses CPU address lines 0 and 1 to decode which of the two serial output ports is to be selected.

The SIO is polled by the Z80A whenever an interrupt request is detected. If the interrupt request flag is present in the status register, the Z80A determines the nature of the interrupt by reading an SIO internal register.

Note: A detailed discussion of the SIO interface is provided in Volume D. For additional information consult the Zilog Z80 SIO Technical Manual, and Zilog's Using the Z80 SIO in Asynchronous Communications.

PIA (Peripheral Interface Adapters)

There are two PIA's used in the design of the Executive computer. The System Select PIA monitors and gates input and output signals for system components and peripherals while the Communication Port PIA provides the parallel interface for connecting outside devices to the computer.

The System Select PIA has eight bank-select lines, eight input and output lines, and four control lines. Data lines PA0 through PA7 are used for bank-select logic. PB0 through PB5 are output lines. PB6 and PB7 are input lines. The four control lines are CA1, CA2, CB1, and CB2. All of the input/output and control lines are summarized in the following table:

TABLE A-2. SYSTEM SELECT PIA I/O AND CONTROL LINES

Output/Input Lines

PB0	Double density
PB1	Select Drive A
PB2	Select Drive B
PB3	Speaker output
PB4	Receive clock
PB5	Transmit clock
PB6	Modem ready
PB7	Modem ring

Control Lines

CA1	DMA interrupt request
CA2	Keyboard interrupt request
CA3	Real-time clock interrupt request
CA4	50/60 Hz select output

In addition to controlling peripheral device selection, the decoder also provides clock selection, determining whether the peripheral devices will be controlled by the computer's internal programmable clock generator or by the peripheral device.

The system's parallel interface is provided by the Communication Port PIA, the principal component for the IEEE 488 port. It coordinates the transfer of parallel data between the data bus and peripheral devices. The IEEE 488 internal ports are selected by CPU address lines 2, 3, 4, and 7 by signals routed through the System Select PIA.

The programmable nature of the 6821 PIA chips makes it possible to accommodate a variety of interface configurations. See Volume D of this manual for additional details and specifications of the PIA.

8253 Timer

The 8253 has three channels for counter timing. The first counter (Port 04) is for baud rate control on RS-232 Channel A. The second counter (Port 05) is used for baud rate control on RS-232 Channel B. The third (Port 06) is reserved to control disk-drive motor speed, but is not implemented at this time.

Disk Interface

The Floppy Disk Controller converts data and coordinates the activity of the disk drives. Interface with the Z80A is maintained over 8 bidirectional data lines and 6 control lines. The disk controller uses four physical ports (08 through 0B), but one port is used alternately for read and write operations, effectively giving the disk controller five logical ports. For a detailed description of how the controller works, consult the information of the 1793 FDC provided in Volume D of this manual.

The Floppy Disk Controller can be addressed directly, but it is generally easier to use the standard CP/M BDOS calls described in Volume B.

Port Selection

An I/O mapping technique is used so that the Z80A can coordinate the flow of data between system components and peripherals. IN and OUT instructions from the Z80A result in port addresses being generated on the lower address lines. Subsequently, the chips are selected by the I/O Decoder via the System Select PIA. Only one device can gain access to the bus at a given time.

Devices directly controlled by the decoder are:.

Real Time Clock
Keyboard
Parallel I/O
Serial I/O
Floppy Disk Controller
Counter-Timer Circuit

The I/O Decoder, via the Peripheral Interface Adapter (PIA), controls the following functions:

Selection of disk drive A or B
Priority control of the internal memory
RS-232 RI and DSR lines
TXCLK and RXCLK decode select
Double-density select
Audio Speaker
50/60 Hz monitor select

TABLE A-3. I/O PORT DESCRIPTIONS

PORT RANGE	DESCRIPTION
00-03	6821 System Select (PIA)
04-07	8253 Timer (baud rates)
08-0B	1793 Disk Controller
0C-0F	Z80-SIO/2 Serial Interface
10-13	6821 Parallel Interface (PIA)
14-17	Keyboard Read
18-1B	Real-Time Clock Read
1C-1F	Video Enable

The I/O decoder is enabled by the Input/Output Chip Select signal. Address lines 2, 3, 4, and 7 from the CPU are decoded to determine which peripheral device is to be selected. Devices controlled by the PIA require additional information from the CPU and computer timing circuits. Address lines 1 and 3 provide additional peripheral-device selection criteria after decoding by the PIA. The PIA must also be enabled by the computer timing circuits.

Data flow direction (Read or Write) is determined by the state of the Input/Output - Read/Write signal. This signal is governed through software by the Z80 port-addressed I/O conventions, whereby IN and OUT instructions contain address information in bits 2 through 4. Bits 0 and 1 are available for further decoding into four "functions" -- analogous to registers -- within the device. In many cases, these represent registers internal to the chip that is selected as a device.

Interrupt Structure

When the Z80 processor is interrupted by a peripheral device (through the SIO or a PIA), a one-byte value is placed on the data bus. The low-order byte of this value points to the address in memory where the CPU is to fetch an address to branch to. The high-order byte is retrieved from the CPU's I register, which is initialized at power-on or reset.

Before branching to the specified address, the CPU pushes the address of the next instruction to execute on the stack so that the interrupt handler can simply execute a RET instruction and return control to the interrupted routine.

The vector addresses are initialized by the ROM. There is one address for the PIA at hex FFFE, and eight for the SIO, from FFE0 to FFEF.

The SIO is allotted 8 vectors since it is configured in Status Affects Vector mode. This means that the SIO places a different value on the Bus depending on what condition has caused the interrupt (i.e., Channel A Rx, or Channel B Tx, etc.).

Although there are 9 vectors in all, they all point to the same routine, which is referred to as the "Global Interrupt Handler" (INT_HDL). This global interrupt handler saves the current state of the interrupted routine and establishes the state for interrupt handling (i.e., sets up the interrupt stack and interrupt bank). Next the global handler passes control to the "Interrupt Polling Routine" (INT_POLL). This routine determines the source of the interrupt. It first checks the SIO, then each channel (A and B) of both PIA's for a pending interrupt.

When the SIO generates an interrupt, the type of interrupt is determined by evaluating the vector placed on the bus. (The value can be retrieved from Read Register 2 on the SIO). Since the value is different for each interrupt condition, it is easy to ascertain what type of interrupt has occurred.

If neither the SIO nor the PIA's have generated the interrupt, then the interrupt is assumed to have been caused by the depression of the RESET button on the front panel of the Executive.

Once the source of the interrupt has been determined, the interrupt polling routine calls the "Interrupt Dispatching Routine" (INT_HD). This routine retrieves the bank mask and address for the appropriate interrupt handling routine.

Three bytes are retrieved from the "Interrupt Dispatching Table" (INT_TBL = hex 023F3). The table is stored in ROM, and read into location hex 23F3 of Bank 8 RAM. The Interrupt Table contains 3 bytes of information for each of the 14 possible devices; the first byte indicates which bank contains the appropriate servicing routine, and the next two indicate the address within that bank where the service routine can be found.
The Interrupt Table contains service routine locations for the following devices in order:

DMA
Parallel port
SIO transmit, Channel B
SIO external/status Channel B
SIO receive, Channel B
SIO receive, Channel B special condition
SIO transmit, Channel A
SIO external/status Channel A
SIO receive, Channel A
SIO receive, Channel A special condition
Intelligent keyboard
Real time clock (tick) interrupt
Floppy disk controller
System reset

The purpose of each interrupt handling routine is to service the interrupt. Once the interrupt has been serviced, the handler returns to the global interrupt handler which restores the state of the machine to the condition it was in before the interrupt occurred, and returns to the address pushed on the stack by the CPU when the interrupt was acknowledged.

Installing Different Interrupt Handlers

All that is required to install a customized interrupt handling routine is a modification of the interrupt dispatching table using BANK MOVE. (This should be done with the interrupts disabled). However, the customized interrupt handlers must meet certain criteria in order to function properly.

The routine must process the interrupt and then clear it. SIO interrupts are cleared with a IRET instruction and PIA interrupts are cleared by reading the appropriate channel. For an example of how the interrupt handling routine should be structured, consult the ROM listing, Volume E.

Since CP/M uses the interrupt handlers to communicate with the serial ports, customized routines should either retain the same link or establish a new one. Also, the Real Time Clock interrupt is used by the system to service the keyboard, maintain the system clock, and check for diskette changes. All of these functions must also be performed by the new handler.

To circumvent the need to include these functions in the routine, retrieve the bank mask and address of the ROM's interrupt handler from the interrupt dispatching table, save it, and replace it with the address of the customized routine. Then, when an interrupt occurs, reinstate the ROM's interrupt handler using BANK JUMP which will handle all of the necessary processing.

The interrupt handler must not re-enable interrupts. Interrupts are enabled after the global interrupt handler re-establishes the state of the machine before it was interrupted. If an interrupt were to occur in the midst of processing another interrupt, the state saved on the interrupt stack would be destroyed and the system would crash.

Testing of the new interrupt handler should be performed with some hardware debugging tools such as an ICE or a logic analyzer.

Character I/O Buffering

All character I/O except for the parallel interface is interrupt driven and processed through a set of buffering routines. These routines are:

FL_BF_ST--Determines if there is room in the buffer for another character.

RM_CH_ST--Determines if there is a character to remove from the buffer.

FL_BF_CH--Places a character in the buffer.

RM_CH_BF--Removes the next character from the buffer.

These routines implement a first-in/first-out queuing mechanism with dynamic buffer allocation. Each device has a list of buffers associated with it in which characters are added to the end of the list and removed from the beginning. These buffers are allocated as the need arises and then removed from the list when they no longer contain characters.

The buffering routines must always be called with interrupts disabled because they are called by the interrupt handlers. An interrupt issued during one of these routines could disrupt the data structure during its transition from one state to another.

Note: The device numbers and data structures are described in the ROM listing, Volume E. Register interfaces are described in the header for each routine.

MEMORY

Main Memory for the Executive computer consists of 128K bytes of dynamic "Random Access Memory" (RAM), 8K bytes of "Read Only Memory" (ROM), and an additional 2K bytes of RAM used in conjunction with the system ROM. The 128K bytes of RAM memory (of which 4K is unused), are provided by sixteen 4164 memory chips arranged in two 64K x 8-bit arrays on a separate PC board.

Memory for the monitor ROM is currently supplied by a 2764 (8K x 8) EPROM located on the main logic board. A 6116 static RAM chip (2K x 8) also located on the main logic board offers additional storage for program variables and tables necessary for system operation.

Effective use of memory is accomplished through software by means of a bank switching technique which allows the CPU to address more memory than would otherwise be possible with the available address lines.

Details concerning the main memory including the RAM, ROM, and Scratchpad RAM will be discussed here. Video memory is detailed in a later section.

RAM Functional Operation

RAM is treated as two contiguous 64K blocks of memory which are addressed by the Z80A through 8 input lines. Since only 8 address inputs are available for each 64K of RAM, 16-bit addressing is accomplished by alternately sharing the lines for an 8-bit row address and an 8-bit column address. The Z80A strobes the 8-bit row to determine the low order byte of the address and then strobes the 8-bit column for the high order byte. To maintain the dynamic RAM storage area in an accurate state it is continuously refreshed by means of the RAS and CAS signals and the CPU-generated refresh signal RFSH.

The row and column address lines are controlled by tri-state buffers. These buffers also perform the function of isolating RAM loading from the address bus. Row and Column address buffers are enabled by the Row Address Enable (RAE*) and Column Address Enable (CAE*) timing signals. The RAE* and CAE* timing signals are derived from the same flip-flop which prevents them from both being active simultaneously. This flip flop is clocked at 24 MHz to accomplish this quick sequencing.

Direction of data flow DATA IN to the RAM array (Read RD*) or DATA OUT of the RAM array (Write WR*) is determined by the state of the RD* signal originating from the CPU. The RD* signal is active when low which places the RAM array into the read mode. A high RD* signal allows data to be written to RAM memory.

Note: The Executive has five unused bank select enable lines available through the 6821 PIA so that future memory expansion can be accommodated.

RAM Connector Pinouts

All signal flow to and from the memory board is via the RAM connector. Pinouts for this RAM connector are listed below:

TABLE A-4. RAM CONNECTOR PINOUTS

Pin	Signal Description
1	Frame Ground
2	ROM Decoder
3	MREQ*
4	VRAM DECODE*
5	RFSH*
6	Not Used
7	RAS
8	Not Used
9	Signal Ground
10	Signal Ground
11	Signal Ground
12	Signal Ground
13	CAS*
14	Not Used
15	WR*
16	RD*
17	CAE*
18	Not Used
19	RAE*
20	Not Used
21	CAS
22	Not Used
23	Signal Ground
24	Signal Ground
25	Signal Ground
26	Signal Ground
27	ADR 15
28	ADR 14
29	ADR 13
30	ADR 12
31	ADR 11

TABLE A-4. RAM CONNECTOR PINOUTS (cont.)

Pin	Signal Description
32	Signal Ground
33	ADR 10
34	ADR 9
35	ADR 8
36	ADR 7
37	ADR 6
38	Signal Ground
39	ADR 5
40	ADR 4
41	ADR 3
42	ADR 2
43	ADR 1
44	ADR 0
45	Signal Ground
46	Signal Ground
47	DATA 7
48	DATA 6
49	DATA 5
50	DATA 4
51	DATA 3
52	Signal Ground
53	DATA 2
54	DATA 1
55	DATA 0
56	PRI 6
57	PRI 5
58	PRI 4
59	PRI 3
60	Signal Ground
61	PRI 1
62	PRI 2
63	12V
64	Signal Ground
65	+5 V
66	+5 V
67	+5 V
68	+5 V
69	+12 V
70	+12 V
71	+12 V
72	+12 V

ROM Functional Operation

There are two ROM slots located on the main logic board. Memory for the system firmware is currently supplied by a 2764 (8K x 8 bits) EPROM located in ROM socket 0. A 6116 RAM chip which provides additional memory storage for system operations occupies ROM socket 1. Both ROM sockets can accommodate a number of different memory chips. A set of jumper switches on the main logic board adapts the address and enable lines for the correct configuration.

The ROM enable signal (CE) is developed by the computer timing circuits, and the ROM output enable signal (OE*) is derived by a dual 2-to-4 line ROM Decoder. The upper half Y0* provides an active-low output enable for ROM 0 and Y1* provides OE* for ROM 1. The inputs to the ROM Decoder from the CPU are RD*, WR*, MREQ*, and ADR 13. ADR 13 selects the ROM to be accessed. MREQ* indicates the CPU has requested data.

Note: Jumpers J1, J2, J3, J4, J5, and J8 are set according to the type of chips used in ROM slots 0 and 1. ROM slot 0 can accommodate a 2732, 2764, or 27128. ROM slot 1 can accommodate all of the afore-mentioned chips as well a 6116 RAM. The current configuration for these jumpers is shown below:

J1: 1 - 2
J2: 3 - 4
J3: 1 - 2
J4: 2 - 3
J5: 1 - 2
J5: 1 - 2
J8: none

ROM Structure

The Monitor ROM consists of 11 assembly language source modules as described below:

Initialization--Routines contained in this module initialize the memory, SIO, PIA's, interrupts, and character font RAM at power-on or reset.

Power-On diagnostics--Diagnostic routines are run when the computer is powered-on to test the memory, parallel interface chip, the SIO serial chip, the clock timer and the ROM check sum.

Boot Loader--The boot module reads the first sector from track 0 of drive A to hex address 4000. The first byte of the sector should be a 0, the fifth byte should contain the product code (2 for the Osborne Executive). If these bytes are valid, the ROM will issue a jump to the beginning of the sector (hex address 4000).

Console Drivers--The console module performs two functions, it takes input from the keyboard and sends output to the screen.

The console input routine normally returns the ASCII values associated with a key being pressed. When a function key is pressed and the function key translation is enabled, the console input routine returns the translated value one character at a time. In other words, if function key 1 were defined as DIR (RET), the first character returned by the console input routine would be a D, then an I, and so on.

The console output routine displays ASCII characters on the screen. It also accepts control characters and ESC sequences to control console functions such as video attributes and cursor positioning.

Real Time Clock Driver--The Real Time Clock driver is accessed by the 60 Hz interrupt and updates the clock every second.

Keyboard Driver--The keyboard driver scans the keyboard 60 times a second. Valid keys are stored in a buffer where the console input routine can access them. When a key is held down for longer than half a second, the key will repeat and will be stored in the buffer. If three or more keys are pressed simultaneously, the keyboard driver will ignore them all. If only two keys are pressed, the driver will accept the last one that is pressed, ignoring the first one.

Parallel Port I/O--The parallel port driver performs input and output as well as checking status conditions for the parallel port. The driver can handle either IEEE 488 or Centronics protocol.

Serial Port I/O--This driver contains routines that set the baud rate, check status, and perform input and output for both serial ports.

Disk Drivers--The disk drivers contain all of the routines which communicate with the disk drives. They read a sector, write sectors, home the drive, or seek a track.

Interrupt Handlers--When an interrupt occurs, the handler saves the current state of the program being run. It saves the stack, the registers, the bank and address where the code was executing. Next the interrupt handler polls the interrupt to determine whether it is a 60 Hz, SIO, parallel port, or RESET button interrupt. Once the type of interrupt is determined, the appropriate handler is called. When the interrupt has been serviced, the program continues where it left off.

Character I/O Buffering--The character I/O buffering module is used to store characters waiting to be input or output. The character buffering routines are used for serial port I/O and keyboard input. There is enough room so that 640 characters can be buffered before any data is lost.

Note: A complete listing of the Executive ROM as well as the entry and exit points are provided in <u>Volume E.</u>

Scratchpad RAM

A 6116 static RAM chip (2K x 8), located on the main logic board, provides 2K of RAM scratchpad memory for system use. The following tables and pointers are located here:

CLK_PTR:--Points to the "seconds" byte of a 4-byte time stamp initialized by the BIOS which points to the clock area of the CP/M Plus System Control Block (SCB). The data structure for the clock area is the same as that described for the SCB which is fully described in <u>Volume B.</u> If this value is changed then the CP/M clock will not function properly.

MEDIA
A_DRIVE:--Points to the media change detect in SCB.
B_DRIVE:--Points to media disk change in the Disk Parameter Header; an FF indicates a disk change, a 00 indicates no disk change.

A_PROTCL:--Sets protocol for Channel A of the SIO.

B_PROTCL:--Sets protocol for channel B of the SIO.

0 -- no protocol
1 -- XON/XOFF protocol
2 -- ETX/ACK protocol

K_MSK_TBL:--Table of bit masks indicating which keys in the keyboard matrix are valid. A 1 indicates a valid key and 0 indicates an invalid key. This table is initialized by CP/M's cold boot process.

AL_KEY:, CT_KEY:, SH_KEY:--Lock and Shift keys in the keyboard matrix. Bits 0, 1, and 2 equal the column while bits 3, 4, and 5 equal the row in this data structure.

KEYS:--Starting here are 5 keyboard tables used to translate keystrokes into ASCII values. Each table is 64 bytes long. There is a one to one mapping between each keyboard table and the keyboard scan matrix.

INT_TBL:--This is the interrupt dispatching table.

Memory Organization

Memory for the Executive is organized into 9 banks shared by system routines and user programs. Bank 1 is for user programs, Banks 2 through 6 are not currently implemented, and Banks 0, 7 and 8 are reserved for system use.

Organizing memory into banks allows the CPU to address more memory than would otherwise be possible. Each bank represents a range of addresses in memory. A technique referred to as bank switching is used to enable certain segments of memory as they are needed.

The following memory map shows how the banks are organized for the current release (ROM V1.21 and BIOS V1.1) of the Executive computer:

```
+---------+  +---------+  +---------+  +---------+  +---------+
|         |  |xxxxxxxxx|  |xxxxxxxxx|  |xxxxxxxxx|  |xxxxxxxxx|
| CP/M    |  |xxxxxxxxx|  |xxxxxxxxx|  |xxxxxxxxx|  |xxxxxxxxx|
|  BIOS,  |  |xxxxxxxxx|  |xxxxxxxxx|  |xxxxxxxxx|  |xxxxxxxxx|
|  BDOS   |  |---------|  |---------|  |xxxxxxxxx|  |xxxxxxxxx|
|         |  |         |  |         |  |---------|  |xxxxxxxxx|
|         |  |         |  |         |  |4K (by 4)|  |xxxxxxxxx|
|         |  |         |  |         |  | ATTRIB  |  |xxxxxxxxx|
|         |  |         |  |         |  |---------|  |xxxxxxxxx|
|---------|  |         |  |         |  |4K (by 8)|  |xxxxxxxxx|
|         |  |         |  |         |  | VIDEO   |  |xxxxxxxxx|
|RESERVED |  |  60K    |  |  60K    |  |---------|  |xxxxxxxxx|
|  FOR    |  |  RAM    |  |  RAM    |  |xxxxxxxxx|  |xxxxxxxxx|
| SYSTEM  |  | (USER   |  | (USER   |  |xxxxxxxxx|  |xxxxxxxxx|
|  USE    |  |  AREA)  |  |  AREA)  |  |xxxxxxxxx|  |xxxxxxxxx|
|         |  |         |  |         |  |xxxxxxxxx|  |xxxxxxxxx|
|         |  |         |  |         |  |xxxxxxxxx|  |---------|
|         |  |         |  |         |  |xxxxxxxxx|  |         |
|         |  |         |  |         |  |xxxxxxxxx|  |   6K    |
|         |  |         |  |         |  |xxxxxxxxx|  | UNUSED  |
|         |  |         |  |         |  |xxxxxxxxx|  |---------|
|         |  |         |  |         |  |xxxxxxxxx|  | 2K RAM  |
|         |  |         |  |         |  |xxxxxxxxx|  |---------|
|         |  |         |  |         |  |xxxxxxxxx|  |         |
|         |  |---------|  |         |  |xxxxxxxxx|  |   8K    |
|         |  |  CP/M   |  |         |  |xxxxxxxxx|  |   ROM   |
+---------+  +---------+  +---------+  +---------+  +---------+
   RAM          USER      EXPANDABLE     VIDEO         ROM
                RAM          RAM
  BANK 0       BANK 1     BANKS 2-6      BANK 7       BANK 8
```

NOTE: xxxx means that address space is unused by this bank and another bank is seen here.

FIGURE A-9. OSBORNE EXECUTIVE MEMORY MAP

Memory in Bank 0 encompasses 64K bytes of RAM reserved for system routines. This area of memory contains BIOS, a portion of BDOS, and is also where the disk and interrupt buffers are located. The top 4K bytes of Bank 0 is used by the system to control bank switching and data handling, and thus is always enabled.

Bank 1 is composed of 60K RAM, with the area between hex 100 and EFFF available for user programs. The bottom 256-byte area, from hex 0 to 100, is reserved for various, critical system address pointers and is referred to as Page Zero. Consult Volume B for a detailed description of the contents of Page Zero. Banks 2 through 6 are not currently implemented.

However, by devising a different memory scheme, these additional banks or any number of banks up to a total of 32 can be implemented.

Bank 7 is composed of video memory. The lower portion of Bank 7 (hex C000 through CFFF) stores bits 0 through 7 for character generation and includes an area reserved for the Monitor and the DMA port. The upper portion (from hex D000 through DFFF) contains the attribute bits.

Bank 8 is 16K and occupies memory between hex addresses 0000 through 3FFF. The memory in Bank 8 is treated differently depending on whether the CPU is reading from or writing to it. When read by the CPU, Bank 8 contains 8K of ROM in the lower portion and 2K of RAM above it. The remaining upper 6K is unused.

When being written to, the bottom 4K is RAM and is where the two character sets (2K each) reside. Above the area reserved for the character set is 4K of unused address space. The next 2K of RAM is used for temporary storage (scratchpad RAM) and contains a pointer to the keyboard translation table. The top 6K of Bank 8 is unused, but will still overlay memory with the rest of the bank. The following illustration shows how Bank 8 appears differently in read and write modes:

```
        Read            Write
3FFFh +----------+   +----------+
      |          |   |          |
      |    6K    |   |    6K    |
      |  unused  |   |  unused  |
      |----------|   |----------|
      |  2K RAM  |   |  2K RAM  |
      |----------|   |----------|
1FFFh |          |   |          |
      |          |   |  4K ROM  |
      |          |   |          |
      1    8K    |   |----------|
      |   ROM    |   | 4K RAM   |
      |          |   | Character|
      |          |   |   Sets   |
      +----------+   +----------+
```

FIGURE A-10. BANK 8

Note: The memory in Bank 8 can be expanded by substituting other chips and using the appropriate jumpers to select them.

Memory Priority

Banks of memory can be thought of as pages that overlay each other as they are being used. A priority scheme determines which banks take precedence. Bank 0 is always enabled. Banks 1 through 8 are enabled by writing a data byte to port 0, with the bank(s) to be enabled identified by their corresponding bit being toggled ON. Here are the bits associated with each bank.

```
Bank      Bit

8         7
7         6
6\        5\
5 \       4 \
4  >      3  > not used
3 /       2 /
2/        1/
1         0
0         (Always enabled)
```

Those banks that correspond to higher-order bits (bits 6 and 7, for example) will have higher priority than banks corresponding to lower-order bits (bits 0 and 1). When two banks are selected that shadow each other, the bank with the higher priority takes precedence in that area. Here is an example:

```
          7   6   5   4   3   2   1   0   Bits
       +--------------------------------+
       |   |   |   |   |   |   |   |   |
       | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
       |   |   |   |   |   |   |   |   |
       +--------------------------------+
Banks     8   7   6   5   4   3   2   1   0
```

FIGURE A-11. DATA BYTE OUTPUT TO PORT 0

In the example above, Banks 0, 1, 7, and 8 are enabled. Bank 8 has the highest priority, so its 16K of ROM and RAM will overlay the CP/M and user areas of Bank 1 and part of the system area of Bank 0. Likewise, the 8K of video memory of Bank 7 will overlay the corresponding locations of Banks 0 and 1, leaving the area from hex 4000 to BFFF available for user programs. The following illustration shows what portions of memory would be enabled:

```
          +---------+ Source
          | CP/M,   |
          | BIOS,   | Bank 0
          | BDOS    |
    E000h |---------|
          |4K (by 4)| Bank 7
          | ATTRIB  |
    D000h |---------|
          |4K (by 8)| Bank 7
          | VIDEO   |
    C000h |---------|
          |         |
          |         |
          |         |
          |         |
          |         |
          |         |
          |         |
          | USER    |
          | AREA    | Bank 1
          |         |
          |         |
          |         |
          |         |
    4000h |---------|
          | UNUSED  | Bank 8
          |         |
    2800h |---------|
          | RAM     | Bank 8
    2000h |---------|
          |         |
          | ROM     | Bank 8
          |         |
          +---------+
```

FIGURE A-12. EFFECTIVE MEMORY AREA

As another example, suppose we output the following data byte to port 0.

```
         7   6   5   4   3   2   1   0
 Bits
       +-------------------------------+
       |   |   |   |   |   |   |   |   |
       | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
       |   |   |   |   |   |   |   |   |
       +-------------------------------+
         8   7   6   5   4   3   2   1   0
 Banks
```

FIGURE A-13. SAMPLE DATA BYTE

With this byte, we will have enabled Bank 1, and since Bank 0 is always enabled, the effective memory area will look like this.

```
        +---------+ Source
        | CP/M,   |
        | BIOS,   | Bank 0
        | BDOS    |
  F000  |---------|
        |         |
        |         |
        |         |
        | USER    |
        | AREA    | Bank 1
        |         |
        |         |
        |         |
        |         |
   100  |---------|
        | CP/M    | Bank 1 (256 bytes)
        +---------+
```

FIGURE A-14. NORMAL MEMORY ENABLING

The example above shows the memory that is enabled during normal use of the CP/M operating system.

Memory Allocation

The following table shows how the memory is allocated (all addresses are in hexadecimal notation):

TABLE A-5. ALLOCATION OF MEMORY

Range	Use
0000 - 0100	CP/M Page 0
0100 - F305	Temporary Program Area (TPA)
F306 - F8FF	Resident portion of BDOS in common memory
F9D0 - FCFF	Resident BIOS
FD00 - FD3F	Currently unused and reserved for user patches
FD40 - FD49	Pointers to common RAM routines installed by ROM
FD4D - FD41	Pointer to bank jump routine
FD42 - FD43	Pointer to bank call routine
FD44 - FD45	Pointer to bank move routine
FD46 - FD47	Pointer to DMARD routine
FD48 - FD49	Pointer to DMAWRT routine
FD4A - FEDB	ROM's Common RAM data area
FEDC - FEEF	Reserved
FEF0 - FFD1	Common RAM routines installed by ROM
FFD2 - FFDF	Reserved
FFE0 - FFEF	SIO hardware interrupt vectors
FFF0 - FFFD	Reserved
FFFE - FFFF	PIA hardware interrupt vectors

Using Memory in Bank 0

Although the 64K bytes of memory in Bank 0 is reserved for system use, memory allocated for LRU disk buffering can be used by an application program. The data structure for this disk caching is documented in Volume B.

Data buffers from one of the drive's list may be eliminated to provide this memory. However, at least one buffer must be reserved for disk data and directory data for each drive. Before the application program returns to CP/M it will need to restore all of the buffers. This is accomplished by recording the contents of the BCB (Buffer Control Block) data structure for each of the deallocated buffers as well as any other pertinent data structures that need to be restored.

In order for an application that resides in Bank 1 to access data in Bank 0, the BIOS function calls 25 (MOVE) and 29 (XMOVE) can be used. Note that a corresponding XMOVE must be called for each call to MOVE because certain values are reset. MOVE and XMOVE should be called directly, not through BDOS function 50.

Some problems may be encountered when using Bank 0 for application programs. If the program fails, the data structures will not be restored to their original state and that portion of memory will be unavailable until the system is reset or rebooted. Also, disk I/O performance will be decreased because of the reallocation of the buffers. To increase the performance, it may be wise to leave the LRU directory buffers intact as they are stored in memory.

CONSOLE DESIGN

The Executive console design encompasses the video system and the keyboard functions. Memory-mapped video consists of 8K x 12 bits of RAM. The 80 x 24 character display can either be output to the built-in monitor or to an external monitor. Each character placed on the screen is associated with a seven-bit ASCII value plus one bit for each of five screen attributes. There are two sets of 128 characters each stored in a 4K x 8-bit character font RAM.

The detachable keyboard has 69 keys and a 12-key numeric keypad. The numeric and arrow keys also serve as programmable function keys while configurable tables and pointers allow further keyboard functions to be defined.

The Executive console functions are patterned after the Televideo 912. The Executive is also capable of emulating the console functions of a wide range of popular terminals through software.

Video System

The video system is centered around a three-port RAM which maintains a mirror image of the symbols (in ASCII format) that are intended to be displayed on the video screen. Each screen location has an equivalent memory location in RAM; the contents of these locations are changed by either the Z80A or an external device attached to the DMA connector. The Video RAM is accessed automatically by the video display circuitry which supplies the necessary addresses to walk through all screen positions while maintaining synchronism with the vertical and horizontal drive signals to the video monitor.

The encoded ASCII symbol which is outputted from the Video RAM's data lines becomes an address input to the FONT RAM. The FONT RAM contains the actual dot patterns to be routed to the video monitor as directed by the timing and control signals.

During a write operation to the Video RAM, a 12-bit address is introduced from the Z80 address bus to the video system via the Video Address Mux. It is the function of the Address Mux to coordinate the reading and writing to the video system. At the appropriate window period, the first 8 bits which contain the ASCII value of the character are written into Low Video RAM. The remaining 4 bits, which represent the attributes associated with the display, are read into High Video RAM during a subsequent write cycle.

Memory for the video is provided by two 6116 static RAM chips. Display memory is composed of a 128 x 32 matrix of addresses from hex C000 to D000 in bank 7. An area 80 columns by 24 lines (1920 character locations) of the memory is used for display with the remainder reserved or unused in the current implementation.

The 4K x 8 bits of Low Video RAM (between hex C000 and CFFF) is used to store the 7 low-order bits associated with ASCII values plus 1 attribute bit for reverse video. The 4K x 4 bits (from hex D000 through DFFF) of the High Video RAM is used to store the 4 attributes associated with each character position on the screen.

The character symbols are subsequently sent to the character FONT RAM, a 4K x 8-bit 6116 static RAM accessed through Bank 8. A serial sequence of bits is outputted for each line of the displayed character. A dot clock driving a serial shift register then generates the signal which illuminates the appropriate dot pattern on the video display line-by-line.

Parallel data in the shift register is outputted one bit at a time. Each dot clock cycle allows one bit of data to be displayed. One full character cycle provided by the character clock allows one full character to be displayed. Scan and Text counters keep track of the rows, lines, and the number of characters per line.

The video monitor display must constantly be refreshed in order to maintain the display. Each monitor display frame (one complete display) is generated 50 or 60 times per second depending on the currently selected monitor frequency. The monitor frequency can be changed through the SETUP utility.

Internal Video Monitor

The built-in, 7-inch (measured diagonally) amber video monitor has an 80-character by 24-line display. Each raster is generated by an electron beam sweeping across the phosphor-coated screen which emits light when struck by the beam. Whether or not the screen emits light is determined by the intensity of the electron beam. The beam intensity is modulated by highs and lows where "1's" cause the screen to emit light and the "0's" result in no light.

The beam starts in the upper left-hand corner and sweeps across the screen from left to right to form row 1 which takes 52 microseconds. Ten rows are required to form one text line since character cells are formed in an 8 x 10 dot matrix. The beam retraces in 12 microseconds while horizontal blanking is in effect, which prevents the retrace from being visible. This process is repeated until the entire screen is filled. Filling the entire screen (1 frame) is accomplished synchronously with the AC power line input frequency and takes place about 50 or 60 times per second. After the beam fills the screen, it moves back (during vertical retrace) to the starting point while vertical blanking is in effect.

Video Monitor Interface

There are two video-monitor connectors located on the lower front panel of the Osborne Executive computer. An "External video connector" and a "Composite video connector" provide alternative methods for connecting external monitors.

Video from the shift register, along with the vertical and horizontal blanking and sync signals, character clock, and video attributes, are input to the video output circuit. These signals are combined to form the composite video output and are also routed to the 10-pin connector. The real-time clock overflow maintains the real-time clock count in the event the CPU is busy (such as during disk data transfer which cannot be interrupted while in progress).

The Composite Video connector is for direct plug-in of RS-170 standard video monitors with a single coax cable. The composite signal supplied to this jack contains all attributes as well as horizontal and vertical synch signals. However, unlike the External Monitor Connector, power is not supplied thru this connector.

Note: It is recommended that only those monitors that meet the EIA RS-170 standard be used with the Executive computer. Monitors that do not meet the requirements defined by the RS-170 standard may not display data correctly.

The 10-pin External Video Connector provides another means of attaching an external video monitor. Video signals are either directed to the built-in monitor when a shunt plug is in place or to an external monitor when attached. When the shunt plug is attached, video signals are routed from the bottom of the PC board to the monitor connections on the top edge of the board. The following PC board pinouts apply:

TABLE A-6. VIDEO EDGE CONNECTOR PINOUTS

Pin	Signal Name
2	Ground
4	Brightness high
6	Brightness low
8	Brightness arm
10	Ground
12	Horizontal synch
14	+12 Volts
16	Video out
18	Vertical synch
20	Ground

```
   19  17  15  13  11  9  7  5  3  1
  //  //  //  //  // // // // // //
 //  //  //  //  // // // // // //
20  18  16  14  12  10 8  6  4  2
```

FIGURE A-15. EXTERNAL VIDEO EDGE CONNECTOR

Note: Do not attach anything to this connector while the Executive computer is powered-on; doing so may cause damage to the computer. Also, do not lose the shunt plug, since the built-in monitor will not operate without it.

Normally, the shunt plug connects each of the above signals to the top side of the PC board connector. In addition, a single in-line connector on the main logic board becomes active when the shunt is in place. The in-line connector provides the signals and power to the internal video monitor. The pinouts for this internal connector are listed in the following table:

TABLE A-7. INTERNAL VIDEO CONNECTOR

PIN	SIGNAL NAME
1	Signal Ground
2	Brightness (This is one leg of a pot)
3	Brightness (This is the other end of the pot)
4	Brightness (This is the wiper arm)
5	Chassis Ground
6	Horizontal Synch
7	+12V (Supplied to the external monitor)
8	Video
9	Vertical Synch
10	Signal Ground

Memory Mapped Video

Video memory resides in Bank 7 from hex C000 to DFFF. Memory between C000 to D000 provides a 128 by 32 memory mapped matrix of which an 80 by 24 portion is used for the display. Memory from D000 through DFFF is used to store the attributes associated with the display.

Memory-mapped video is laid out in the following manner:

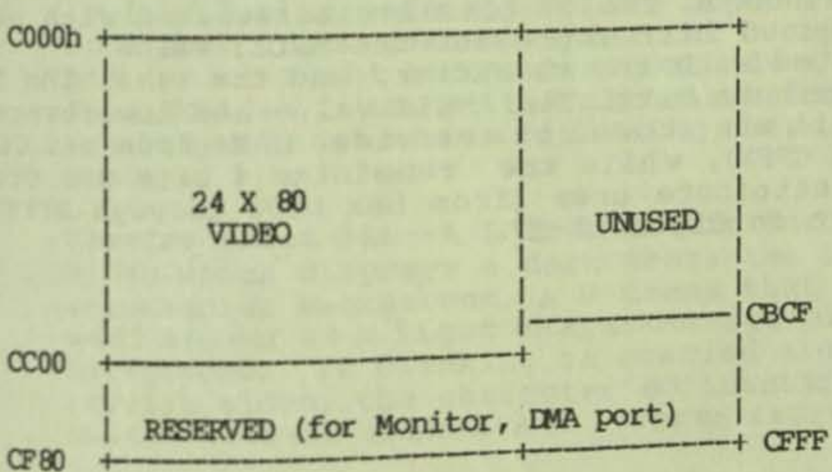


FIGURE A-16. MEMORY-MAPPED VIDEO

Note: In the above diagram, location C000 corresponds to the upper left corner of the screen; location CBCF corresponds to the lower right corner.

A proprietary feature of the Executive allows 12-bit block moves. When moving data in the video memory (C000 - CFFF) with LDIR and LDDR instructions, both the character and attribute bits are moved as a unit. Moves outside the Video RAM area are accomplished in 8-bit segments.

The Executive utilizes software scrolling so that the top line of the screen display is always located at hex address C000. Lines 25 through 32 (decimal) are allocated in video RAM for DMA and monitor functions, and columns 81 through 128 of each line are currently unused.

The normal method of enabling the video bank for a user-defined video driver is shown below:

```
IN   0          ;Get current banks enabled
STA  TEMP       ;Save in temp location
OR   A,40H      ;Enable video bank
OUT  0          ;
.
.               ;User video driver here
.
LDA  TEMP       ;Restore banks
OUT  0
```

Video Attributes

Characters for the display are stored in the character FONT RAM, accessed from hex address 0000 thru FFFF in Bank 8. Twelve bits are associated with each character; 7 bits represent the ASCII value associated with the character, and the remaining 5 are attribute bits. The ASCII value and the reverse video bit are stored in the video RAM (from hex C000 through CFFF), while the remaining 4 bits are stored in the attribute area (from hex D000 through DFFF), as shown in Figure A-17.

```
                     D000 - DFFF        C000 - CFFF
                   +--+--+--+--+--+--+--+--+--+--+--+--+
                   |7 | 6| 5| 4| 7| 6| 5| 4| 3| 2| 1| 0|
                   +--+--+--+--+--+--+--+--+--+--+--+--+
                    |  |  |  |  |  |  |  |  |  |  |  |
   Full intensity <+  |  |  |  |  |  |  |  |  |  |  |
   Underline <--------+  |  |  |  |  |  |  |  |  |  |
   Blink <---------------+  |  |  |  |  |  |  |  |  |
   Alternate font <---------+  |  |  |  |  |  |  |  |
                               |  |  |  |  |  |  1  1
   Reverse video <-------------   |  |  |  |  |  |  |
 7 bit ASCII character code <-----+--+--+--+--+--+--+
```

FIGURE A-17. VIDEO ATTRIBUTE BITS

Full intensity -- A value of 1 in this bit indicates full intensity; a 0 indicates that the character will be half-intensity.

Underline bit--A value of 1 in this bit indicates the character will be underlined; a 0 value means no underline.

Blink bit--A value of 1 turns blinking ON while a 0 turns blinking OFF. The associated character will blink about twice a second.

Alternate font bit--This bit is used to access either the standard or alternate character set stored in the Font RAM. If this bit is 0, the standard set in the lower 2K of the Font RAM is used. If this bit is 1, the alternate set in the upper 2K is used. Note that a 256-character set can occupy that 4K of RAM (which might be the case for a language other than English). Use of the alternate font bit can be used as an index when addressing these two areas of memory.

Reverse video bit--A 1 in this bit enables reverse video which displays a dark character in a light rectangular background. A 0 means that the character will appear as a light character against a dark background. If blinking is enabled along with reverse video, the character so identified will switch between normal and reverse representation.

Direct Screen Manipulation

The video memory is actually two paired sections of Bank 7 that allow each character to have 7 character bits and 5 attribute bits.

Remember that each character is actually considered to be 12 bits wide: 8 bits, consisting of 7 ASCII character bits and one attribute bit, are stored in the video RAM area of Bank 7 (from C000 through CFFF); while the remaining 4 attribute bits are stored in the attribute area of the same bank (from D000 through DFFF).

By directly manipulating the relevant bits for each character, individual memory locations can be changed without having to use the output sequences demonstrated earlier. To get to the attribute bits, use the assembly language code shown below.

```
IN      0
STA     TEMP     ;save in temp location
ORI     40h      ;shadow in video memory
OUT     0
```

Next read the memory location you wish to change, and set or clear the bits you wish to change. Note that 1=on and 0=off for all attributes.

```
LDA     TEMP     ;shadow out video
OUT     0
```

By addressing the video attributes in this manner you can create a "template" of the screen areas that are to be affected. Once set, these areas will remain so affected (e.g., blinking, etc.) as new characters are written into them. These areas should be reset to their default values before the template program is exited.

Character Generation

A character set is a collection of characters in a particular style. The Executive is capable of storing two character sets of up to 128 characters each. The default character set is composed of Standard English (ASCII) and resides in the lower 2K of Bank 8. The alternate character set currently consists of 128 graphic characters. Any set of characters can be defined through the CHARGEN program described in Volume 1 of the Executive Guides. Reading and writing to this area involves different types of memory. Reading accesses the ROM, and writing accesses the character FONT RAM.

The alternate character set can be enabled for a particular character by toggling a bit in the lower 4K of Bank 8 from hex 0000 through 0FFF.

Each character set is selected by the "alternate set bit" (bit 4 of the Attribute RAM in Bank 7); a 0 in this bit directs the system to use the character set stored in the lower 2K; a 1 in this bit directs the system to use the alternate character set in the upper 2K. The character font is enabled by setting the high bit on port 0. An example of a subroutine that enables the font RAM is shown below:

```
IN    0              ;GET CURRENT BANK STATUS
STO   A,TEMP
OR    100000000b     ;SET FONT (and ROM) BANK BIT
OUT   0              ;ENABLE BANK 8
.
.                    ;USER CODE
.
LD    A,TEMP
OUT   0              ;RESTORE PREVIOUS BANK STATUS
RET
```

Because Bank 8 has a higher priority than Bank 1 (where the user TPA is located), any subroutine like the example presented must be located above hex 3FFF.

The 4K of FONT RAM is divided into two sections, one for each of the character sets. The first half from 0 to hex 07FF is reserved for the main character set. The top half from hex 0800 to 0FFF is reserved for the alternate character set. Each character set is 128 character definitions of 16 bytes each. Because the character size is only 8 x 10 pixels, 6 of the bytes are not used.

Keyboard Pinouts

The pinouts for the keyboard connector are listed below:

TABLE A-8. KEYBOARD CONNECTOR PINOUTS

Pin	Row/Column	Description
1	—	Chassis Ground
2	Row 4	Address 12
3	Row 0	Address 8
4	Row 3	Address 11
5	Row 6	Address 14
6	Row 2	Address 10
7	Row 5	Address 13
8	Row 1	Address 9
9	Row 7	Address 15
10	Column 0	Data 0
11	Column 1	Data 1
12	Column 2	Data 2
13	Column 3	Data 3
14	Column 4	Data 4
15	Column 5	Data 5
16	Column 6	Data 6
17	Column 7	Data 7
18 *	—	+12 V
19 *	—	+12 V
20	—	Signal Ground

* These pins are enabled only by dealer-installed jumpers for those keyboards requiring a +12 V output.

The physical configuration of the connector as seen from the front of the Executive is shown below.

19	17	15	13	11	9	7	5	3	1
20	18	16	14	12	10	8	6	4	2

FIGURE A-18. KEYBOARD CONNECTOR PIN CONFIGURATION

Note: Since the Executive uses a non-encoded keyboard, encoded keyboards will not operate properly.

Keyboard Interface

The Executive keyboard provides all of the standard alphanumeric and symbol keys, as well as four arrow keys, a ten key numeric keypad and the following special purpose keys:

[SHIFT]: — When depressed, provides the uppercase equivalent or character shown on the upper portion of the key being pressed.

[TAB] — Advances to the next tab stop; tabs are set every 5 spaces.

[ALPHA LOCK] — When locked down, indicates that all subsequent letters should be uppercase. The ALPHA LOCK key affects only the letter keys.

[RETURN] — This rectangular key advances the cursor to the next line on the screen or signals a command sequence.

[ESC] — The "ESCAPE" key functions differently depending on which program you are using. It is usually used to "escape" from a situation by cancelling whatever process is in progress.

[CTRL] — The CTRL key is used in conjunction with one of the other keys to issue commands sequences.

Besides the keys visible on the keyboard, there are four hidden characters which exists but are not shown on the keycaps. These characters can be displayed by holding down the CTRL key and simultaneously typing the indicated character:

CTRL / -- tilde (~)

CTRL < -- left bracket ({)

CTRL > -- Right bracket (})

CTRL = -- Open single quote (')

Note: An alternate character set or any character you desire can be defined by using the CHARGEN program described in Volume 1, Chapter 2, of the Executive User's Guide.

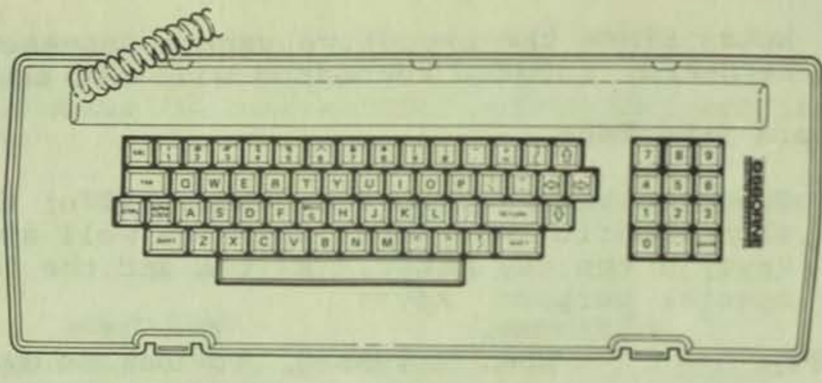


FIGURE A-19. KEYBOARD LAYOUT

The keyboard supplied with the Executive basically consists of a 8-column x 8 row matrix of keyswitches. Pressing a key causes a switch closure. The Z80 monitors the input/output lines to decode the switch closure and converts the signals into data and addresses which are subsequently converted into ASCII code by the monitor ROM. The keyboard matrix as it appears to the software decode function is shown below:

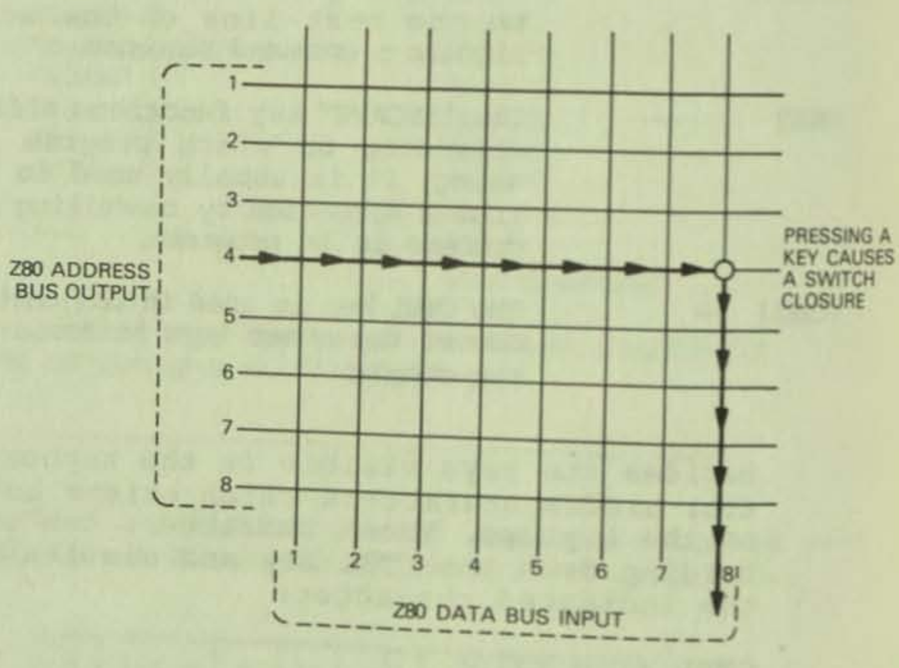


FIGURE A-20. KEYBOARD MATRIX

The keyboard port is a read-only port. It is tested with an IN instruction for port 14, which scans the 8 x 8 matrix. The 8 bits of Register A are set to test any of the eight rows in the matrix. When the IN instruction is performed, the byte returned flags the column being tested. A positive value returned in Register A signifies the column of the key being pressed. For instance, in the example below row 3 is flagged for testing.

Register A

0	0	0	0	1	0	0	0	
7	6	5	4	3	2	1	0	Bits

If a 0 value is returned for bit 2 in the A Register:

1	1	1	1	1	0	1	1	
7	6	5	4	3	2	1	0	Bits

the decode tables in Bank 8 will interpret the key pressed as an A.

Configurable Keyboard Tables and Pointers

The following section describes five tables and pointers that may be modified for user-specific keyboards and console definitions.

There are five pointers to the keyboard translation tables starting at hex address 2000 in Bank 8. Included are the key code table, the shift table, the control key table, the alpha lock table, and the control shift table.

TABLE A-9. KEY CODE TABLE
(starting location = KYCDTB)

KYCD-TB	ESC	TAB	ignored	ignored
+ 4	ignored	<cr>	'	[
+ 8	1	2	3	4
+ C	5	6	7	8
+ 10h	q	w	e	r
+ 14	t	y	u	i
+ 18	a	s	d	f
+ 1C	g	h	j	k
+ 20	z	x	c	v
+ 24	b	n	m	,
+ 28	8Ah	8Dh	0	{space>
+ 2C	.	p	o	9
+ 30	8Bh	8Ch	-	/
+ 34	;	\	l	=
+ 38	ignored	ignored	ignored	ignored
+ 3C	<cr>	ignored	ignored	ignored

TABLE A-10. SHIFT KEY TABLE
(used when shift key is down)

SHFT-TB + 40	ESC	TAB	ignored	ignored
+ 44	ignored	<cr>	"	]
+ 48	!	@	#	$
+ 4C	%	^	&	*
+ 50	Q	W	E	R
+ 54	T	Y	U	I
+ 58	A	S	D	F
+ 5C	G	H	J	K
+ 60	Z	X	C	V
+ 64	B	N	M	<
+ 68	8Ah	8Dh	)	<space>
+ 6C	>	P	O	(
+ 70	8Bh	8Ch	_	?
+ 74	:			L	+
+ 78	ignored	ignored	ignored	ignored
+ 7C	<cr>	ignored	ignored	ignored

TABLE A-11. CONTROL KEY TABLE
(used when control key is down)

CT-TB	+ 80h	ESC	TAB	ignored	ignored
	+ 84	ignored	<cr>	ignored	^[
	+ 88	81h	82h	83h	84h
	+ 8C	85h	86h	87h	88h
	+ 90	^Q	^W	^E	^R
	+ 94	^T	^Y	^U	^I
	+ 98	^A	^S	^D	^F
	+ 9C	^G	^H	^J	^K
	+ A0	^Z	^X	^C	^V
	+ A4	^B	^N	^M	{
	+ A8	8Ah	7Fh	80h	note: ^BackArrow=DEL
	+ AC	}	^P	^1O	89h
	+ B0	8Bh	8Ch	^	~
	+ B4	ignored	^\	^L	`
	+ B8	ignored	ignored	ignored	ignored
	+ BC	<cr>	ignored	ignored	ignored

TABLE A-12. ALPHA LOCK TABLE
(used when Alpha Lock key is down)

AL-TB	+ C0	ESC	TAB	ignored	ignored
	+ C4	ignored	<cr>	'	[
	+ C8	1	2	3	4
	+ CC	5	6	7	8
	+ D0	Q	W	E	R
	+ D4	T	Y	U	I
	+ D8	A	S	D	F
	+ DC	G	H	J	K
	+ E0	Z	X	C	V
	+ E4	B	N	M	,
	+ E8	8Ah	8Dh	0	
	+ EC	.	P	O	9
	+ F0	8Bh	8Ch	-	/
	+ F4	;	\	L	=
	+ F8	ignored	ignored	ignored	ignored
	+ FC	<cr>	ignored	ignored	ignored

TABLE A-13. CONTROL/SHIFT TABLE
(Used when Control and Shift keys
are pressed simultaneously)

CS-TB	+ 100	ESC	TAB	ignored	ignored
	+ 104	ignored	<cr>	ignored	^]
	+ 108	81h	^@	ignored	84h
	+ 10C	ignored	^^	87h	88h
	+ 110	^Q	^W	ignored	^R
	+ 114	ignored	^Y	^U	^I
	+ 118	^A	^S	ignored	^F
	+ 11C	ignored	^H	^J	^K
	+ 120	^Z	^X	ignored	^V
	+ 124	ignored	^N	^M	{
	+ 128	8Ah	7Fh	ignored	space
	+ 12C	}	^P	^D	{
	+ 130	8Bh	8Ch	^_	7Fh
	+ 134	^_	^\	^L	`
	+ 138	ignored	ignored	ignored	ignored
	+ 13C	<cr>	ignored	ignored	ignored

The function-key translate pointers and tables are in the same area. The pointers include FKEYFLAG (= 0 if the program is to use the ROM table, = 1 if the application tables should be used, or = 2 if neither is to be used); ROMFKEY and APPFKEY which point, respectively, to the ROM and application function key tables.

A number of control pointers are stored in the same area:

CURPOS--Points to the current cursor position.

BACKGND--Is the background attribute flag (bit 7 = reverse video, bit 6 = full intensity, bit 5 = underline, and bit 4 = blink).

Programmable Keys

In addition to the normal key interpretations, 14 programmable keys are available. The 0 thru 9 keys and the four arrow keys can be converted by a routine in BIOS to any series of up to 236 keystrokes. The function keys are programmed through the SETUP program located on the System diskette and described in <u>Volume 1</u> of the <u>Executive Guides.</u>

Function key definitions are stored sequentially in the Scratchpad RAM. Each function key definition consists of a length byte followed by the actual function key translation.

Applications programs can define the function keys by directly modifying their location in the RAM area of ROM, but this approach is not recommended as it entails bank switching and references locations which are subject to change.

The preferred method of defining function key values from within an application program is to disable the function key translations. This will cause the function key code rather than the translation to be passed to the program. The codes are hex 80 - 89 for function keys 0 - 9, and 8A - 8D for the arrow keys. The function key translations are disabled by sending the escape sequence ESC f to the console, and re-enabled with ESC e. Programs which define their own function key values should re-enable the translations before being terminated.

Console Commands

Cursor positioning is user-programmable. In order to position the cursor in column x of line y, you would send the following command sequence to the console:

<ESC> = <row y> <column x>

where <row y> is the number of the row where the cursor is to appear, and <column x> is the column number. Both numbers should be offset by 32. In BASIC, the sequence would be coded like this:

```
10 PRINT CHR$(27)+"="+CHR$(row+32)+CHR$(col+32);
```

Alternatively, to place the cursor in a desired location, an assembly language routine such as the following would reposition the screen.

```
         BDOS    EQU  0005h
         ESC     EQU  27
         PLACE   EQU  yyxx        ;you fill in
SETCUR:  MVI     E,ESC
         CALL    OUTCH            ;send ESCAPE
         MVI     E,'='
         CALL    OUTCH            ;send SETCUR
         LXI     H,PLACE
         LXI     D,2020h          ;bias for YX
         DAD     D                ;add bias to position
         PUSH    H                ;save column s
         MOV     E,H              ;prepare row y
         CALL    OUTCH            ;send row y
         POP     D                ;prepare column x
         CALL    OUTCH 'RET       ;send column x and return
OUTCH:   MVI     C,6              ;CP/M write to console
                                  ;function
         CALL    BDOS             ;call CP/M
         RET
```

Other video attributes can be controlled by issuing the appropriate sequence of characters. The Executive Monitor is patterned after the Televideo 912, and thus uses many of the same screen control characters as listed in the following table.

TABLE A-14. SCREEN CONTROL CHARACTERS

Keyboard Sequence	Decimal Sequence (BASIC)	Hex Sequence (Assembly)	Action
CTRL G	07	07	Beep
CTRL H	08	08	Cursor left, no erasure
CTRL J	10	0A	Cursor down. Line feed
CTRL K	11	0B	Cursor up
CTRL L	12	0C	Cursor right
CTRL V	22	16	Cursor down. No line feed
CTRL ^	30	1E	Home cursor
CTRL _	31	1F	New Line
CTRL I	09	09	Tab
CTRL Z	26	1A	Clear screen
CTRL TAB			Toggle Alternate/Standard Character Set
ESC)	27 41	1B 29	Start Dim mode
ESC (	27 40	1B 28	End Dim mode
ESC Z	27 90	1B 5A	Clear screen
ESC "	27 34	1B 22	Keyboard enable
ESC #	27 35	1B 23	Keyboard disable
ESC =	27 61	1B 3D	Load cursor
ESC W	27 87	1B 57	Character delete
ESC E	27 69	1B 45	Line insert
ESC R	27 82	1B 52	Line delete
ESC T	27 84	1B 54	Clear to end of line
ESC Y	27 89	1B 59	Clear to end of page
ESC ^	27 94	1B 5E	Start blink
ESC g	27 103	1B 67	End blink/blank
ESC j	27 106	1B 6A	Start inverse
ESC k	27 107	1B 6B	End inverse
ESC l	27 108	1B 6C	Start underline
ESC m	27 109	1B 6D	End underline
ESC <	27 60	1B 3C	Key click on
ESC >	27 62	1B 3F	Key click off
ESC a	27 96	1B 60	Start alternate char
ESC A	27 65	1B 41	End alternate char
ESC e	27 101	1B 65	Key translate enable
ESC f	27 102	1B 66	Key translate disable
ESC w	27 119	1B 77	Define window
ESC s	27 116	1B 74	Set window
ESC .	27 46	1B 2E	Define cursor type
ESC o	27 111	1B 6F	Set X/Y offset
ESC b	27 98	1B 62	Set reverse video
ESC d	27 100	1B 64	Set normal video
ESC x	27 120	1B 78	Define background

To clear the screen in a BASIC program, use the following "PRINT" statement.

```
PRINT CHR$(&H1A)
```

In assembly language, place a 1Ah in the E register and use CALL OUTCH to clear the screen.

Because of the similarities between the Executive and the Televideo 912's video functions, many user programs that work correctly with the Televideo terminal should also work correctly with an Executive.

DISK DRIVE SYSTEM

The complete disk system consists of a Floppy Disk Controller, the disk drive electronics, and two 5-1/4 slim-line disk drives, all of which are directed by firmware routines in ROM. The disk drive controller provides the interface between the Z80 and the disk drives. The disk-drive electronics board receives data and control signals from the controller and processes them for direct use by the drive stepping motor, the drive head, and the drive motor. The controller is directed by low-level disk drivers in the ROM through high-level drivers which are accessed via the BIOS. The disk drives access data from 5-1/4 inch, single-sided diskettes according to a number of popular formats.

Controller

The Floppy Disk Controller is essentially a bi-directional Large Scale Integration (LSI) chip, located on the main logic board that provides a serial/parallel interface between the computer and the disk drive system. The controller interface also includes a write precompensator and a raw data separator for synchronizing write clock signals and for separating read clock signals.

The disk drives are interfaced to bus and control circuits by the disk controller and the disk-drive electronics board. The disk controller performs the following functions:

1. Provides parallel-to-serial and serial-to parallel conversions on data being transferred between the data bus and the disk drives.

2. Uses software via the data bus to provide the following disk drive control:
   - o Selection of the disk drive (A or B)
   - o Direction of head travel (In or Out)
   - o Head stepping commands
   - o Direction of data transfer (Read or Write)
   - o Selection of the side of the disk to be used (if using a double-sided disk drive)

3. Accepts status information from the disk drives consisting of the following:
   - o Indicates when the head has reached disk track zero (for software reference).
   - o Indicates when the disk being used has a write-protect tab installed (if the user is attempting to perform an operation requiring writing to the disk).
   - o Indicates when the disk index marker (beginning of sector) has been detected.

The primary components that work in tandem with the disk controller are the Peripheral Interface Adapter (PIA), a Write Precompensator, Read Data Separator, and a Dual Density Hybrid (DDEN) chip.

Control signals transmitted to the disk drives through software consist of STEP, DIR, WR GATE, and WR DATA. STEP controls the disk-drive stepping motor, DIR controls the direction of movement (in or out) of the disk-drive stepping motor, WR GATE controls whether data is being read or written onto the disk, and WR DATA consists of the actual data being transferred to disk.

DRV SEL 1 and DRV SEL 2 are drive select signals provided by software via I/O Decoder circuits. The I/O Decoder circuits are responsible for selection of the I/O peripheral gaining bus access in addition to providing the drive select signal. The I/O Decoder circuits are discussed in detail in Volume C.

The Disk Controller is selected by the I/O Decoder circuits which controls selection of all I/O devices. The decoder is enabled by an Input/Output Chip Select signal. The actual selection of the Disk Drive Controller is determined by CPU address lines 2, 3, 4, and 7. This produces the I/O select signal which is routed to the Controller. The I/O Decoder circuit also controls selection of Disk Drive A or Disk Drive B and enables the dual-density mode of operation.

Disk Drive Electronics

The disk drive electronics are contained on three circuit boards which process signals from the Controller. These electronics assist in converting data signals from parallel to serial and vice versa, control the servo motors, and detect the index and write-protect status.

The 1 MHz clock is supplied by the computer timing circuits for operation in the single-density mode. When dual density is selected by software via the I/O Decoder circuits, the DDEN Hybrid is enabled. This chip uses the 4 MHz clock signal.

The Drive Select signal from the disk controller serves to select either disk drive A or disk drive B for the particular operation currently being performed. The Read Amplifier or the Write Amplifier is selected by the state of the WR GATE signal. The Disk Drive Electronics board also contains circuits to maintain a constant drive motor speed.

Disk Drives

The disk drive on the top is drive A and the one on the bottom is drive B. The drives use 5-1/4 inch, single-sided diskettes as the storage media.

The drives supplied with the Executive comply with ANSI (American National Standards Institute) standards; however, other non-ANSI drives are also supported. The primary differences here involve supplying +12 V (versus ground) from pins 11, 13, 15, and 17; supplying +5 V (versus ground) from pins 21, 23, and 25; and supplying the +5 V MOTOR ON from pin 16.

The head is normally positioned in a retracted position away from the Read/Write slot. Data is input to memory and displayed on the video monitor without being written to disk. The disk drive is enabled only when data is saved to or retrieved from the disk.

When the disk rotates, the index hole is optically detected and that information (INDEX) is sent back to the computer so the position of the head can be monitored. With INDEX and the TRACK 00 as a reference, along with its own control information (step direction and number of tracks to step), software can maintain control and account for the head position at all times.

The disk drives, when operating, rotate at 300 rpm which results in a 200 millisecond period of rotation. The head moves across the 40 tracks taking approximately 20-milliseconds-per-track step time. Serial data is transferred between the disk drives and the controller. The following specifications describe the characteristics of the disk drives:

- o Ready Line is always active.
- o Head load line is always active and tied to the drive select.
- o Motor-on is tied to the drive select (internally).
- o Step rate: 20 milliseconds
- o Head settle time: 20 milliseconds
- o Head load delay: 35 milliseconds
- o Motor spin-up time: 750 milliseconds
- o Rotational speed: 200 msec =/-1%
- o Instantaneous speed variation: +/0 - 5%
- o Erase turn-off delay: 1.1 msec
- o Index pulse length: 4 msec +/ -2%
- o Transfer rate: 250,000 bits per second (MFM)

The following signals are provided by the pinouts on the disk drive connector:

TABLE A-15. DISK DRIVE CONNECTOR PINOUTS

Pin	Signal Name
1	Ground
2	Spindle *
3	Ground
4	Not Used
5	Ground
6	Not Used
7	Ground
8	Ground
9	Ground
10	Drive Select 1 *
11	Ground
12	Drive Select 2 *
13	Ground
14	Not Used
15	Ground
16	Motor On *
17	Ground
18	DIR *
19	Ground
20	Step *
21	Ground
22	Write Data *
23	Ground
24	Write Gate *
25	Ground
26	TR00 *
27	Ground
28	Write Protect
29	Ground
30	Read Data *
31	Ground
32	Side Select 1*/ 0
33	Ground
34	Not Used

For a more detailed discussion on these signals consult Volume C.

Disk Format

The disk media used is single-sided, single-density, soft-sectored 5-1/4 inch diskettes. Data is stored on the diskette in 40 tracks of five 1024-byte sectors each, resulting in approximately 186K bytes of data storage per diskette.

Even though information is stored on the diskette in 5 sectors of 1024 bytes each, to CP/M there are 40 sectors of 128 bytes each on the diskette. In other words, if you are using the Executive disk routines as documented, you'll be working with 5 physical sectors of 1024 bytes; but if you're working with CP/M BIOS or BDOS routines, you'll be dealing with 40 logical sectors of 128 bytes each.

The disk byte capacity, sectors per track, and tracks per disk depend on the particular disk format being used. The disk drives are capable of supporting the following disk formats:

TABLE A-16. DISKETTE FORMATS SUPPORTED

	Osborne Single Density	Osborne Double Density	Xerox Cromemco	IBM CP/M 86	DEC
Tracks per disk	40	40	40	40	40
Physical sectors per track	10	5	18	8	9
Logical sectors per track	20	40	18	32	36
Bytes per sector	256	1024	128	512	512
Total disk capacity	100K	200K	90K	160K	180K
Reserved tracks	3	3	3	1	2
Directory tracks	4	4	3	1	2
Data storage capacity	90K	183K	83K	156K	171K
Sector skew factor	2	1	5	1	2

The first three tracks of the Executive program diskettes contain System information. Track 0 contains the initialization program in sector 1 while the CP/M Plus loader program occupies sectors 2 through 5. Sectors 1 through 4 on track 1 contain the CCP.COM program and sector 5 contains the BIOS overrides. Sectors 1 through 4 of track 2 hold the Character generation tables while sector 5 contains the BIOS overrides.

Disk-Drive ROM Routines

High- and low-level disk drive routines stored in the ROM provide the interface with the Floppy Disk Controller. The high-level routines are accessed through the BIOS and interface with the more specialized function of the low-level drivers which in turn direct the Controller. The low-level routines parallel the commands used by the Floppy Disk Controller described in <u>Volume D.</u>

High Level Drivers

There are four high-level disk drivers located in the ROM. Each of the high-level drivers is called through the ROM jump table by passing the appropriate parameters to determine such things as track, sector, and DMA address. Prior to entering the ROM directly, the stack must be located above hex 4000 and there must be sufficient stack space allocated for the ROM to perform its functions. Here are the four high-level routines:

RDRV--This routine positions the head in the home position. First the variable SDISK is checked to determine which drive is active (0 = A, 1 = B). Then the proper drive is selected (SELDRV), a STEPIN is performed, and the drive is homed (HOME). If the HOME routine returns an error, then the entire routine is performed again. No registers are set with this routine.

RSEC/WSEC--These routines read or write a sector of the diskette respectively. Register B contains the number of sectors to read and write, which must be equal to or less than the maximum number of sectors on the selected track. The variables SDISK, DMADR, SAVTYP, SAVTRK, and SAVSEC must be set appropriately.

The Read and Write Sector routines set their respective flags then jump to the common read/write routine R_WSEC. The drive is selected (SELDRV) and, if necessary, a read address (RADR) is performed followed by a seek (SEEK). R_WSEC calls RD_WRT which actually performs the read/write operation beginning with the sector specified by SAVSEC. If an error is returned by either RADR, SEEK, or RD_WRT, then up to ten retries are performed. On the fifth retry (or 1/2 the total retries), a call to RDRV is performed to recalibrate the head. The retries do not reselect the drive, but start by reading the address (RADR).

The H and L registers return the last DMA address plus 1 except where no bytes are actually transferred in which case they will return whatever the DMA was previously set to. This feature can be used to determine in which sector the error occurred. For instance, if five sectors were transferred and the value returned in the H and L registers was equal to the starting DMA address plus 1024, then only one sector (1024 bytes) was transferred and the error occurred in reading the second sector.

SENDEN--This routine senses the density and number of sectors per track of a given diskette. The "Sense Density" routine determines whether the diskette is formatted in single or double density and sets the appropriate bit in the SAVTYP variable. The SDISK variable specifies which drive is currently active. No registers need to be set prior to entering this routine.

The SAVTYP density bit is initialized at power-up to 0, or is set with the density last read for the drive which is the density that is checked first. SENDEN assumes that there is no physical interleave of sectors and that the first sector is 1 (not 0).

The drive is selected (SELDRV) and a read address (RADR) is performed while the head remains positioned on the current track. The headers for any sector on the current track are read to determine the side number, track number, sector number, sector size in bytes, and the density.

If the read address returns an error, then the density bit is toggled and the select and read address is attempted again for the alternate density. If an error is still encountered, then the drive is homed and the process is retried ten times. If the home is unsucessful, then the SENDEN routine exits with an error return. A home error is usually caused by an improperly seated or unformatted diskette.

The density and number of bytes per sector saved in SAVTYP is used by SENDEN to calculate the number of sectors per track. This calculation is accomplished by reading sequential addresses and comparing their sector numbers. In other words, if the present sector number is greater than the previous one, then the present sector is saved and the code loops. This algorithm is repeated until the last sector is located and SENDEN sets the number of sectors per track in Register B.

When a high-level routine returns, the zero flag is set if the operation was successful or is reset if an error was encountered. In addition, Register A is reset to 0 if the operation was successful or to 1 if it was not. If an error occurs during any of these high-level routines, up to 10 retries will be performed.

Low Level Drivers

The low-level drivers provide the intermediate interface between the high-level drivers and the floppy disk controller. These low-level drivers are used mostly by system utilities and diagnostics to direct the floppy disk controller without having to worry about timing considerations. Thus, these routines use the same registers and almost directly parallel the commands used by the disk controller.

The command/status register, port 08, on the Executive is used to issue a command when a byte is output or give status when a byte is input. Track sector and data registers are ports 09, 0A, and 0B, respectively.

Timing is especially important for read/write functions. The tracks and sector registers as well as the command/status registers should not be written to after the disk controller is active or busy. However, data can be read from and written to the data registers while the controller is busy. These registers can handle forced interrupts, but the program should wait for the busy signal to drop before doing so. The following functions are performed by the low-level disk drivers:

Read/Write Routines

The Read/Write routines require that the drive be rotating, the number of sectors be specified in Register B, and that the track and sector parameters (SAVTRK and SAVSEC) be set. Also, the buffer referenced by DMADR must be filled or ready to be filled. No retries are performed by this group of routines except during initial setup.

Read/Write Sector (RD_WRT)--Each routine sets the R_WCOM flag to designate the appropriate function and execution and then continues with the common code at RD_WRT.

RD_WRT is executed both by the high- and low-level read and write routines. Register B contains the number of sectors to transfer, and R_WCOM is set to the desired read or write function. First the sector number is output to the controller, then the value derived from SAVTYP is used to calculate the number of bytes per sector. The R_WCOM flag is loaded into Register A, and if multiple sectors are to be transferred this too is noted in Register A and subsequently becomes the value for FDSK.

The total number of bytes to be transferred is calculated and stored in the H and L registers. Interrupts are disabled and the command is issued to the controller (FDSK). If FDSK returns an error, interrupts are re-enabled and the routine returns without any retries being performed. If the function is successful, the low-level support routines DMARD and DMAWRT transfer the bytes in or out of the data register, one at a time, based on the values derived from the controller's status register.

The status of the controller is checked following the data transfer. If the status is busy and multiple sectors are being transferred, the busy is cleared; during single-sector transfers the routine waits until the busy drops. The status is checked once again for errors, the return condition is set, interrupts are enabled, and the routine returns.

Read Address (RADR)--is a simple routine which attempts to read the header of the sector wherever the head happens to be positioned. This routine does not check SAVSEC or SAVTRK, but does require that it be called with the proper density bit set in the SAVTYP variable.

The read address routine is different from the others in that it has a built-in timeout. This means that if the first byte of the sector header transferred by the controller cannot be read, it times out for a period before a retry is attempted by the calling routine; RDADR performs no retries of its own. It also checks for errors and sets the carry flag appropriately. This routine can be used to set the track register and is used by the SEDEN routine to determine density.

Read Track (RDTRK)--This routine simply reads every byte on a given track and works exactly as does the equivalent command used by the controller.

Format Track (FMTTRK)--This routine formats a track of a given diskette. The density bit in the SAVTYP variable is used to determine the value for filling a track while the DMAWRT routine executes the command. The B registers are used to specify the track length, and DMADR points to the first byte of the address in the buffer used to hold the fill characters. The fill byte for single density is FF and the fill byte for double density is 4E. A full track is written following the index pulse. If an error is encountered during this routine, the carry flag is set.

Head Movement Routines

The head movement routines rely on the SELDSK variable to specify head movement speed so that different brands of disk drives can be accommodated. These routines also specify the amount of delay to allow sufficient time to carry out instructions from the controller.

SEEK, STEPIN, and STEPOUT set Register B with the appropriate seek command and then jump to the common code, PSEKC. The seek command is ORed with the value in SEKDEL, then FDSK is called to perform the function. If there are no errors, the routine waits until the busy (WBUSY) drops, a head settle delay of 40 milliseconds takes place, and error checking is performed.

SEEK--This routine executes the controller's seek instruction sequence. A seek will move the head any number of tracks as specified by SAVTRK. By contrast, Step, Step In, and Step Out only cause the head to move a single track relative to the current position.

The Seek routine assumes that the value in the controller's track register corresponds to the current head position. If the value from the track register is not the same as that in the SAVTRK variable, then the specified track number is placed in the data register before the seek is executed. Since no error checking takes place, unreported problems can occur if a seek is performed for a number larger than the number of tracks referenced by the routine.

STEPIN causes the controller to move the head one track toward the center of the diskette while STEPOUT moves it one track towards the outside. STEP causes the head to move one or more tracks in the same direction. Neither of these routines perform error checking.

HOME--This routine directs the controller to issue a RESTORE command which positions the head at track 0. This routine assumes that SDISK has previously been set to select the appropriate disk drive. The function is performed by FDSK using the speed and verify information stored in SEKDEL and followed by a wait (WBUSY). The status is subsequently checked to insure that the operation was successful. If the verify bit in SEKDEL is set, seek and CRC errors are checked. When an error condition is detected, the carry bit is set.

Disk Drive Support Routines

FDSK--The FDSK routine makes sure that the referenced drive is not busy, then sends out the command in Register A. A set period of time is allotted for the command to be carried out (56 microseconds for single density and 28 microseconds for double density). Finally, the controller is checked to insure that the command is being processed, the DACTVE counter is reset to indicate which drive was accessed, and the calling routine is returned. If a timeout which indicates a problem with the controller is detected, then FDDK reports an error condition by setting the carry flag.

SELDRV--This routine activates the disk drive specified by SDISK. Before activating the drive, the density bit of the controller is set to that specified in the SAVTYP variable. If the drive is already active, then the DACTVE is set to FFH and a return is performed. DACTVE is used by the interrupt processor (INTHNDL) to turn the drive off after three seconds to give any other routine time to access the drive while it is turning. SELDRV does not report errors, but does return conditions. For instance, the zero bit is set if the drive was already selected or is reset if the drive was not selected.

WBUSY--This routine monitors the busy bit to determine when it is cleared. If the busy bit remains set for a prolonged period of time, then this routine checks for a timeout error. If a timeout error is detected then the carry flag is set.

DMARD--This routine transfers data from the controller to memory. The number of bytes to be transferred is specified in the BC register and the start of the buffer in the HL register. This routine is moved to common memory (High RAM) on power-up to avoid the need for transfer buffers.

DMAWRT--Transfers data from memory to the controller.

DELAY--This routine waits until the number of milliseconds specified in Register A has elapsed.

FORINT--(Force Interrupt): This routine waits until busy drops, then interrupts the controller.

Variables

The disk transfer variables SEKDEL, DMADR, SAVTYP, SAVSEC, SAVTRK, SDISK, and NRETRY are referenced at one point or another by the low-level drivers. The following variables are used in the high-level disk operations:

DMADR--The first byte address of the DMA buffer that will be read or written on this particular call. Buffer size will be sufficient to hold the maximum number of sectors transferred on this call.

SAVTYP--This is the byte set by the SENDEN routine to indicate number of bytes per sector and single or double density:

```
                  7    6    5    4    3    2    1    0
               +----+----+----+----+----+----+----+----+
               :    :    :    :    :    :    : 0  :    :
               +____+____+____+____+____+____+____+____+
                  :    :    :    :    :    :    :    :
Reserved;    <----+----+----+----+    :    :    :    :
  all zeroes                          :    :    :    :
Number of bytes <---------------------+----+    :    :
per sector 00 = 128                             :    :
           01 = 256                             :    :
           10 = 512                             :    :
           11 = 1024                            :    :
Side; always 0 <--------------------------------+    :
Density <--------------------------------------------+
           0 = double
           1 = single
```

FIGURE A-21. DISK FORMAT BYTE

SDISK--This byte indicates on which drive the read or write operation is to take place (0 = drive A, 1 = drive B). This byte must be set before the disk driver is invoked and is not modified by the driver.

SAVTRK--This byte indicates the track on which the read or write operation is to take place. This byte must be set before entering the driver and is not modified by the driver.

SAVSEC--This byte indicates the sector where the read or write is to take place, or the starting point in the case of multiple-sector operations. This byte must be set before entering the driver and is not modified by the driver.

SEKDEL--Unlike the preceding variables which serve as reference values, this byte is actually used to modify the instructions to the 1793.

```
          7   6   5   4   3   2   1   0
        +---+---+---+---+---+---+---+---+
        :   :   :   :   :   :   :   :   :
        +---+---+---+---+---+---+---+---+
                      :       :   :   :
update <--------------+       :   :   :
verify <----------------------+   :   :
speed  <--------------------------+---+
```

FIGURE A-22. 1793 INSTRUCTION BYTE

This "seek delay" variable specifies the amount of time the head waits on a track between move commands issued by the disk controller. This allows the controller to meet its wait state and allows the hardware to complete the required operation. SEKDEL is ORed into the instruction given to the disk controller to specify the duration of that delay for movement of the head from one track to the other.

The update bit is used to modify the track register of the controller according to the operation being performed. The verify bit specifies the address that should be read from the controller following an operation. The speed bit is tied to number of milliseconds it takes for the head to move from one track to the other as follows:

0 = 6 milliseconds
1 = 12 milliseconds
2 = 20 milliseconds
3 = 30 milliseconds

NRETRY--This byte indicates the number of retries to perform on power-up and is initialized to ten.

PERIPHERAL INTERFACE DESIGN

The Executive computer can interface with a wide range of peripheral devices through connectors on the lower front panel. The following interfaces are provided:

o RS-232-C Serial Modem Port (DCE)
o RS-232-C Serial Printer Port (DTE)
o IEEE 488 Parallel Port

This section presents general information pertaining to the interfaces accessible through connectors on the front panel of the Executive computer. Interface for the disk drives, memory board, and DMA are not discussed here. For information on these interfaces you will need to consult the appropriate section in this volume and in Volume C.

Because equipment manufacturers do not always fully comply with the interface standards, it is recommended that you consult the information presented here before attempting to connect a peripheral device. The term "compatible" often means that some minor hardware or software modifications may be required.

RS-232-C Serial Interface

The RS-232-C is a widely used standard for serial binary data transfer published by the Electronic Industries Association. The Executive RS-232-C serial interface consists primarily of a Z80 SIO/2 (Serial Input/Output) chip which communicates with peripherals through two DB-25S connectors.

The Executive computer uses Channel A of the SIO chip for the modem port and Channel B for the printer port. Software-selectable baud rates up to 38,400 are provided by a 8253 Programmable Clock Generator. The SETUP utility described in Volume 1 of the Executive Guides is used to select the Baud rate, protocol, device assignment, and initialization string for both serial ports. The Z80 SIO/2 and 8253 are discussed in Volume C and information necessary for programming each chip is located in Volume D.

Modem Interface

The modem interface uses a standard RS-232 DB-25S connector which is also compatible with the ISO 2113 standard. This interface is configured for DTE (Data Terminal Equipment). Any RS-232 compatible device may be connected here as long as the pinouts are arranged correctly. If the connector on the peripheral device is also configured for DTE, then a null modem cable should be used.

Modem Port Pinouts And Signals

Here are the pin designations for the modem port and the signals associated with each:

TABLE A-17. MODEM PORT PINOUTS

Pin	Signal	Input or Output	
1	Chassis Ground		
2	Transmitted Data	(TXD)	O
3	Receive Data	(RXD)	I
4	Request To Send	(RTS)	O
5	Clear To Send	(CTS)	I
6	Data Set Ready	(DSR)	I
7	Signal Ground		
8	Data Carrier Detect	(DCD)	I
11	+V	(+12 V)	O
15	Transmit Clock (Modem Source)	(TXCLK)	O
17	Receive Clock	(RXCLK)	I
20	Data Terminal Ready	(DTR)	O
22	Transmit Clock (Executive Source)	(TXCLK)	O

Note: All other pins (9, 10, 12, 13, 14, 16, 18, 19, 21, 25) are not connected.

The pin arrangement for the DB25-S connector as seen from the front of the Executive is shown below:

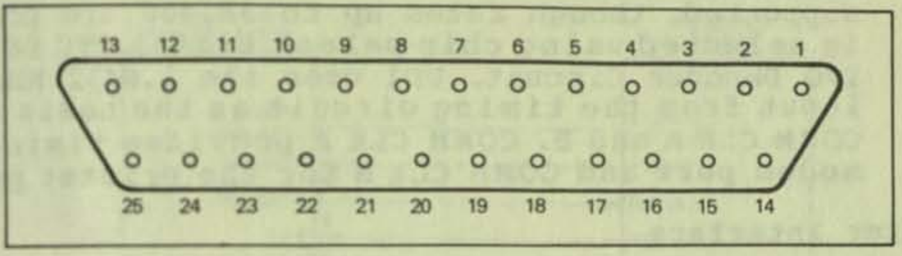


FIGURE A-23. MODEM PORT CONNECTOR CONFIGURATION

Here, the signal direction for each of the active pins is illustrated:

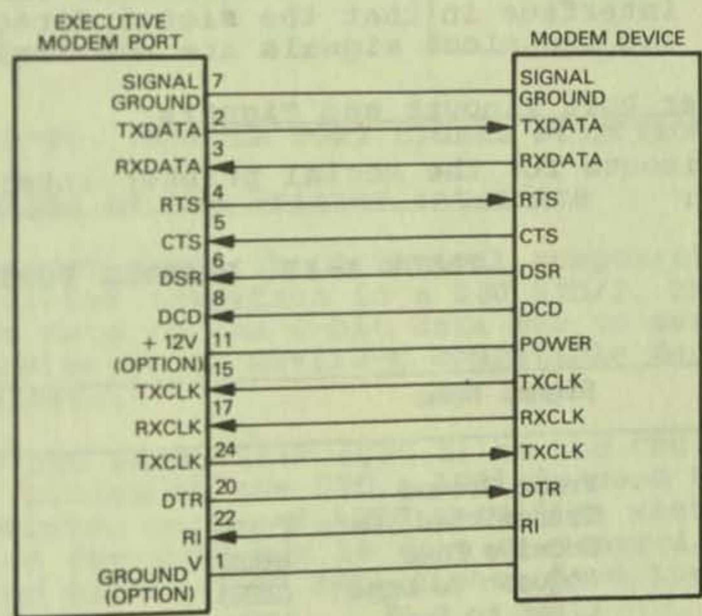


FIGURE A-24. MODEM PORT SIGNAL DIRECTIONS

Functional Description of the Modem Interface

The modem interface uses port A of the Z80 SIO in conjunction with the Programmable Clock Generator UD1. When the Data Set Ready (DSR) and Ring Indicator (RI) signals are received from a peripheral device, they are inverted by UF1 and UF2 then routed to the CPU via the 6821 PIA RS232 receiver UD12. Refer to Volume D for a complete description of the 6821 PIA. Transmit Clock Select (TXC SEL) and Receive Clock Select (RXC SEL) from the PIA determine the clock to be used by the SIO for transmitting and receiving data.

The Programmable Clock Generator UD1 allows software, using ADR0 and ADR1 to control the port baud rate. Software-selectable Baud rates up to 19,200 are fully supported, though rates up to 38,400 are possible. UD1 is selected using chip-select signal CTC SEL from the I/O Decoder Circuit. UD1 uses the 1.8432 MHz clock input from the timing circuit as the basis for the COMM CLK A and B. COMM CLK A provides timing for the modem port and COMM CLK B for the printer port.

Printer Interface

The printer port is configured for Data Circuit Equipment (DTE). Note that the handshaking system used to control the transfer of data consists of "Request To Send" (RTS) and "Data Set Ready" (DSR) signals. The printer interface differs from the modem interface in that the signal directions are reversed and the clock signals are not implemented.

Printer Port Pinouts and Signals

The pinouts for the serial printer interface are shown below:

TABLE A-18. PRINTER PORT PINOUTS

Pin	Signal Name
1	Frame Ground
2	Transmitted Data (TXD)
3	Receive Data (RXD)
4	Request to Send (RTS)
5	Clear to Send (CTS)
6	Data Set Ready (DSR)
7	Signal Ground
8	Data Carrier Detect (DCD)
20	Data Terminal Ready (DTR)

The pin arrangement for the DB25-S connector as seen from the front of the Executive is shown below:

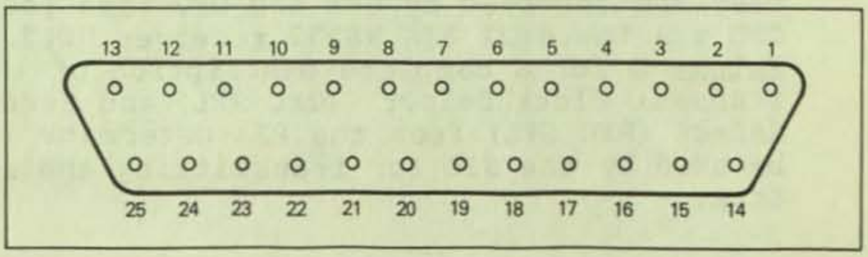


FIGURE A-25. PRINTER PORT CONNECTOR CONFIGURATION

Here, the signal direction for each of the active pins is illustrated:

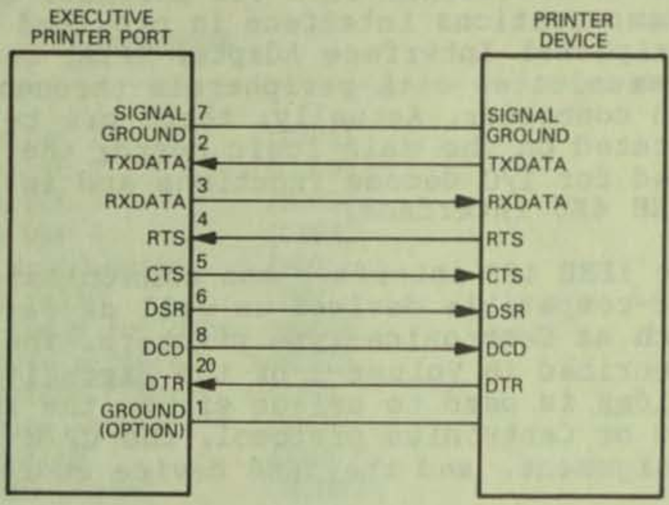


FIGURE A-26. PRINTER PORT SIGNAL DIRECTIONS

Functional Description of the Printer Interface

As with the modem port, the principal component of the serial printer interface is a Z80 SIO/2. The SIO chip converts data on the 8-bit data bus to serial data for transfer to an RS-232-C compatible device such as a printer.

The SIO uses the same clock (CPU I) as the CPU for the internal timing of the SIO. ADR1 is used to select the printer port and ADR0 specifies whether information on the data bus is data or control. The SIO is enabled by the (SIO SEL) signal from the I/O Decoder circuit.

Serial data is transmitted to the peripheral device over the Transmit Data line (TXD) using the Ready To Send (RTS), and Data Terminal Ready (DTR) control signals routed through an Inverter/Buffer. Signals from the peripheral device consist of the Receive Data (RXD), Data Carrier Detect (DCD), and Clear To Send (CTS) signals routed through another Inverter/Buffer.

IEEE 488 Parallel Interface

The bidirectional IEEE 488 parallel data communications interface is provided by a 6821 Peripheral Interface Adapter (PIA) chip which communicates with peripherals through a 24-pin IEEE 488 connector. Actually, there are two 6821 PIA chips located on the main logic board; the other 6821 is used for I/O decode functions and is not part of the IEEE 488 interface.

The IEEE 488 interface can communicate with most IEEE 488-compatible devices as well as parallel devices such as Centronics-type printers. The SETUP utility described in Volume 1 of the Executive Guides is used to select either the IEEE 488 or Centronics protocol, the CP/M device assignment, and the IEEE device address.

IEEE 488 Parallel Port Pinouts and Signals

The only hardware consideration when connecting a peripheral device to the IEEE 488 connector is compatiblity of the pinouts. In some cases, this means a cable must be fabricated. Here are the pin assignments for the IEEE 488 connector:

TABLE A-19. IEEE 488 PINOUTS

PIN	SIGNAL NAME	
1	Data bit 1	(DI01)
2	Data bit 2	(DI02)
3	Data bit 3	(DI03)
4	Data bit 4	(DI04)
5	End or identify	(EOI)
6	Data valid	(DAV)
7	Not ready for data	(NRFD)
8	No Data Accepted	(NDAC)
9	Interface clear	(IFC)
10	Service request	(SRQ)
11	Attention	(ATN)
12	Shield ground	(SHIELD)
13	Data bit 5	(DI05)
14	Data bit 6	(DI06)
15	Data bit 7	(DI07)
16	Data bit 8	(DI08)
17	Remote enable	(REN)
18	DAV signal ground	(DAVGND)
19	NFRD signal ground	(NFRDGND)
20	NDAC signal ground	(NDACGND)
21	IFC signal ground	(IFCGND)
22	SRQ signal ground	(SRQGND)
23	ATN signal ground	(ATNGND)
24	Logic ground	(LOGICGND)

The pin connections on the Osborne Executive are counted from the top right to the lower left. The physical configuration of the 24-pin connector and the signals associated with each pin are illustrated below:

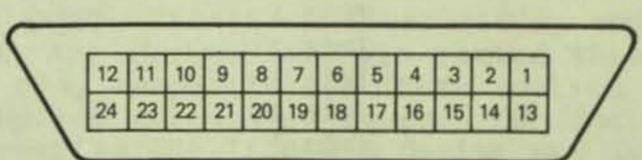


FIGURE A-27. IEEE 488 CONNECTOR CONFIGURATION

Here are the signal directions for the IEEE 488 pinouts:

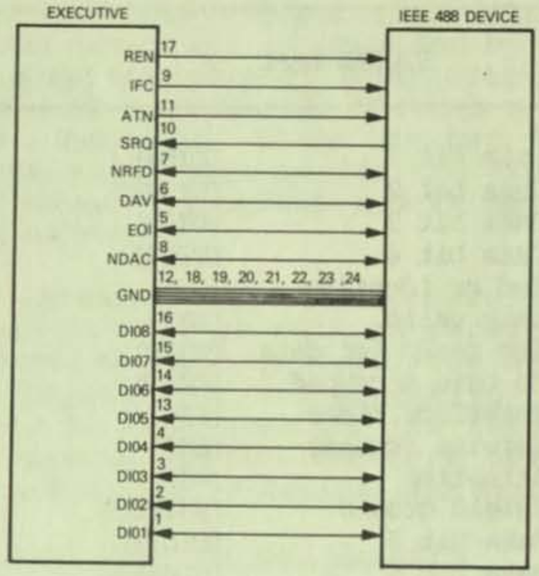


FIGURE A-28. IEEE 488 SIGNAL DIRECTIONS

Centronics Protocol

Firmware within the ROM allows Centronics compatible printers can be run through the parallel interface. The following illustration compares the IEEE 488 edge connector with the standard Centronics connector.

TABLE A-20. IEEE 488 AND CENTRONICS COMPARISON

24 Pin IEEE Edge Connector	Signal Name	36 Pin Centronics Connector
1	Data 1	2
2	Data 2	3
3	Data 3	4
4	Data 4	5
13	Data 5	6
14	Data 6	7
15	Data 7	8
16	Data 8	9
6	Strobe (data valid)	1
8	Busy (no data accepted)	11
10	Select (service request)	13
12	Signal Ground	17
19	Signal Ground	19
20	Signal Ground	20
21	Signal Ground	29

Generally speaking, the preceding signal definitions apply to most parallel printers, although the connector on the printer end may be entirely different. One area that has provided confusion is the Centronics protocol definition. A number of printer manufacturers, Centronics included, do not use the full set of ground hookups defined, and in some cases, printer manufacturers who claim Centronics compatibility do not always attain it.

Functional Description of the Parallel Interface

The principal component of the IEEE 488 parallel interface is a 6821 PIA (Peripheral Interface Adapter). The PIA is a tri-state device that coordinates the transfer of parallel data between the data bus and peripheral devices. The IEEE 488 ports are selected by CPU address lines 2, 3, 4, and 7 by signals routed through the PIA. Specifications for the 6821 PIA and its instruction set are located in Volume D for those who wish to make direct use of the PIA for controlling the IEEE 488 port.

There are five signal paths between the PIA and the IEEE 488 connector. The uppermost path allows bi-directional data to pass between the data bus and the device on lines D0 through D7.

The direction of data flow is determined by an inverter driven by the PIA. This invertor is controlled by the signal IOR/W* when the timing signal ENABLE is present and the PIA has been selected (488 SELECT) by the I/O Decoder. The PIA transfers data just as it is presented, meaning that if the data bus line is high, the output to the device will be high. If the input is low, the output will be low. This is true of data transfers in both directions.

Remote Enable (REN), Interface Clear (IFC), and Attention (ATN) are control signals routed through CB2, CA2, and PB4, respectively. These control signals are applied through an Inverter/Buffer circuit. REN enables the IEEE 488 device and is controlled by software via PIA ADR0 and ADR1 input lines which also determine the status of IFC. The ATN signal will follow data bus line 4 when PIA output port B is selected.

The next two signal paths are formed by signals SRQ and FDCIRQ which are routed through the PIA input pins CA1 and CB2. These interrupt signals originate from the disk system and the peripheral device causing IRQ* to be sent from the 6821 to the CPU. FDCIRQ is not a part of the IEEE 488 interface but does use the 6821 input pin CB1 as a means of generating a CPU interrupt.

Service Request (SRQ) also causes the 6821 PIA to generate an interrupt to the CPU.

NRFD, NDAC, DAV, and EOI are bidirectional communications protocol signals. The direction of data flow is controlled by PIA outputs.

IEEE Handshaking Techniques

The IEEE 488 bus transfers data and commands between the Executive and peripheral devices on 16 signal lines. The interface functions for each device is performed within the device itself so only passive cabling is required. The cables connect all instruments, controllers, and other components of the system in parallel to the signal line.

Eight of the lines (D101 thru D108) are reserved for transferring data and messages in a byte-serial, bit parallel manner. Data and message transfers are asynchronous, coordinated by three handshake lines (DAV, NRFD, NDAC). The other five lines are for control of Bus activity.

Up to 15 devices may be addressed on the bus using up to 32 different addresses. An IEEE 488 device address is assigned to the device by the manufacturer and may sometimes be changed through switches built into the device or by jumpers on their PC board. By default, the address in BIOS is set to address device number 4, but can be changed through the SETUP utility.

Devices connected to the Bus may be either talkers, listeners, or controllers. The Executive is the controller and dictates the role of all the other devices by setting the Attention (ATN) line true and sending a talk or listen address on the data lines (DIO1 - DIO8).

While the ATN line is false, only devices that have been addressed will actively send or receive data. All others ignore the data lines. Several listeners can be active simultaneously, but only one device can be a talker at any one time. Whenever a talker address is put on the bus (while ATN is true), all other talkers will be automatically unaddressed.

Information is transmitted on the data line under sequential control of the three handshake lines. No step in the sequence can be initiated until the previous step has been completed. Information transfer can proceed as fast as the devices can respond, but no faster than the slowest device presently addressed as active. This permits several devices to receive the same message byte concurrently.

The ATN line is one of five control lines. When ATN is true, address and universal commands are transmitted on only seven of the data lines using the ASCII code. When ATN is false, any code of 8 bits or less understood by both talker and listener may be used.

The other control lines are IFC, REN, SRQ, EOI. Interface Clear (IFC) places the interface system in a known quiescent state. Remote Enable (REN) is used with other coded messages to select either local or remote control of each device.

Any active device can set the Service Request (SRQ) line true. This indicates to the controller that some device on the bus wants attention. An example of this would be a device that has completed a time-interval measurement and wants to transmit the reading to a printer.

The End Or Identify (EOI) is used by a device to indicate the end of a multiple-byte transfer sequence. When a controller sets both the ATN and EOI line true, each device capable of a parallel poll indicates its current status on the DIO line assigned to it.

It is not necessary for every device on the bus to be capable of responding to all of the lines. Each can be designed to respond only to those lines that are pertinent to its functions on the bus.

Executive IEEE 488 Implementation

The Osborne Executive's programming includes eight low-level IEEE 488 drivers located in the ROM. These routines provide a simplified method of communicating with IEEE 488 devices. Here are the eight primary commands and their entry points to the ROM:

TABLE A-21. IEEE 488 EXIT AND ENTRY POINTS

hex	IEEE ROM Routine	Entry State	Exit State
0127	Control Out	Any	Source, ATN High
012A	Status In	Source	Source
012D	Go To Standby	Source	Source, ATN High
0130	Take Control	Source, ATN High	Source, ATN Low
0133	Output Interface Message	Any	Source, ATN Low
0136	Output Device Message	Source	Source, ATN High
0139	Input Device Message	Any	Acceptor, ATN High
013C	Input Parallel Poll message	Source	Source, ATN Low

These routines reside in the ROM and may be accessed through the BIOS. The easiest way to access the ROM is through the user function routine TOROM located in the BIOS jump table. To avoid bank switching and referencing a location that may change in future releases of the BIOS, access the TOROM routine through BDOS function call 50. BDOS 50 provides access to each of the 32 entry points for the BIOS as described in <u>Volume B.</u>

The IEEE 488 commands are defined at a low level and must be called in an intelligent sequence with the appropriate parameters passed through preassigned registers. The commands are general-purpose and can be used by different hardware implementations. There are, however, some restrictions that impose limits which are not part of the IEEE standard, but which can be overcome by modifying and adding to the software. The SRQ input can generate an interrupt, but commands assume the interrupt is not used. It would require a sophisticated software modification to allow for SRQ interrupts. A background scan of SRQ should satisfy most applications.

IEEE 488 command routines use no RAM other than the stack. Routines determine status by reading the current state of the PIA.

The software was made considerably more complicated by the use of bi-directional signals from the PIA. To reduce the overhead of determining the current direction of all signals, the PIA is always left in one of two modes: the source handshake mode, or the acceptor handshake mode. Several of the commands require that the PIA be in the source handshake mode when called. The PIA is normally in the source handshake mode following the completion of an IEEE bus transfer, so this is not a major restriction. Both Status In and Parallel Poll commands require that the PIA be in the source mode. This means that detection of device requests using either serial poll or parallel poll can be done only when the interface is idle.

To send data to a device on the IEEE bus, the controller makes the device a LISTENER, then assumes the role of a TALKER and sources the data. To receive data from a device, the controller makes the device a TALKER, then assumes the role of a LISTENER and accepts the data. It is also possible for the controller to enable both a TALKER and a LISTENER, then go to standby and allow the two devices to talk at their own rate. The problem is how to regain control.

The controller can take control asynchronously by setting ATN true. If a device-dependent message is true when ATN becomes true, the interrupted byte could be misinterpreted by other devices as an interface message and produce unintended state transitions. The problem can normally be avoided by taking control synchronously. If high speed transfer of data between the devices is not required, and if the computer can be tied up during the transfer, it is better to have the controller LISTEN (acceptor handshake) to transfer while discarding the data. This allows the controller to count transfers, look for EOI signals, or timeout the TALKER before regaining control.

The "Go To Standby" command has three uses. If the controller is going to allow an unmonitored data transfer between two other devices on the bus (without listening, as described above), this command is used to relinquish bus handshake control to the devices. Secondly, this command can be used after the completion of a bus transfer, causing the controller to float all signals (except REN) so that a second controller, also on the bus, can take control. This command is also used to initialize the interface following certain device handshake errors.

There is no formal transfer of control as defined in the IEEE Specification, but it does allow an operator the ability to share IEEE devices on the bus.

Commands having loops that might cause the interface to "hangup" when a device on the bus is either busy or malfunctioning have error exits. Therefore, the programmer can define separate timeout limits, report the error, or recall the command and ignore the error.

The timeouts are very short, some as short as 100 microseconds, and are meant to occur often. Some IEEE peripherals use this timeout to indicate that they are not ready to receive data. When outputting to a slow device, e.g., a 300 baud printer, the printer will continuously not be ready for the next character. The program could be written to wait as long as required, even allowing for a printer that is off-line. While the program is waiting for the printer, it is possible for the computer to perform background functions following each timeout. If the command times out, the device can be UNLISTENED. This capability is necessary for devices such as modems and keyboard printers operating in full-duplex mode.

CP/M BIOS calls receive parameters in Registers C or BC, parameters are returned to A or HL. Only registers used to return results and status are modified by command execution.

Control Out

7	6	5	4	3	2	1	0	Bit No.
					ENR	REN	IFC	C Register

IFC: If "1", the interface logic is initialized and IFC is driven low for 100 microseconds.

ENR	REN	
0	X	No action
1	0	REN set high
1	1	REN set low

When the PIA is reset, all internal registers are cleared. All I/O pins except REN become inputs and are pulled high by a pull-up register. REN becomes an input and floats but has no pull-up register. Signals IFC, ATN, and REN power up till the active low state is achieved, but all other signals float. If IFC is set, the PIA is initialized to the source handshake mode in the standby state (ATN high), then the IFC signal is pulsed low to initialize all devices on the bus. The signal REN can be set or cleared independently.

Status In

No input parameters

7	6	5	4	3	2	1	0	Bit No.
							SRQ	A Register

SRQ: Set to "1" if one or more devices are requesting service by driving SRQ low.

This command reads the service request signal. This signal is used by a device requiring attention and also to request that the current sequence of events be interrupted. If SRQ is "1" (low on the IEEE bus) then a device is requesting service. This signal is connected to PIA input CA1 and could be used to generate an interrupt. However, the software has not been written to support interrupts; to detect a request the application program must periodically call Status In to read SRQ.

Since the CA1 input of the PIA can only sense signal transitions, not signal levels, a trick must be used to read the SRQ signal. An open-collector driver controlled by the PIA output PB2 (ENABLE EOI/DAV) is ORed to the output of the SRQ input buffer. If the SRQ signal is low, the buffer presents a high to the PIA. When signal PB2 is driven low, and input CA1 is also driven low, a transition is simulated that the PIA will detect. Conversely, if the SRQ signal is high, the buffer presents a low to the PIA. When signal PB2 is driven low, input CA1 remains low and there is no transition to detect.

This command can be called only when the interface is in the source handshake mode. To use the SRQ interrupt, all command routines would have to be modified to protect the interrupt state during their execution. Interrupts would have to be disabled during command execution.

Go To Standby

No input or output parameters

This command causes the controller to enter the controller standby state where all signals are allowed to float. This command can be used as follows:

1. After completion of a bus transaction to free the bus for use by another controller.

2. To pass control to a device that the controller has previously made a TALKER.

3. To clear an interface error condition.

This command can only be called when the interface is in the source handshake mode. The only action performed by GO TO STANDBY is to clear the ATN signal and float the data bus.

Take Control

7	6	5	4	3	2	1	0	Bit No.
							TCA	C Register

TCA: If "1", the controller will regain control of the bus asynchronously.

If "0", it will regain control synchronously.

7	6	5	4	3	2	1	0	Bit No.
ERR							ATO	A Register

ATO: Set if DAV remains active fo longer than 100 microseconds during take control synchronously.

ERR: Set if any error bit is set.

This command should only be used following a Go To Standby that allows independent transfer of data between devices on the bus. The asynchronous take control can be used safely only when there is no activity on the bus. Normally it is used to take control from a device that has malfunctioned and will not give up control synchronously. Use the synchronous Take Control unless there is a special circumstance that requires asynchronous take over.

If the data valid timeout error is returned (ATO), it means that a LISTENER is slow in accepting data, or the TALKER is slow removing data. This command can be called repeatedly, until the take over is performed successfully.

Output Interface Message

Message byte in the C Register.

7	6	5	4	3	2	1	0	Bit No.
ERR					ATO	RTO	NDP	A Register

NDP: No device present
RTO: Device not ready fo data timeout
ATO: Data not accepted timeout
ERR: Set if any error bit is set

This command outputs the contents of Register C as an interface message (with the ATN signal active). If ATN is not active when called, ATN is first driven low and is left low following the data transfer.

The command first checks the ENABLE DATA OUT bit (PBO) to determine if the interface is in the source handshake mode or the acceptor handshake mode. If this bit is set, then the interface is in the acceptor mode and must be switched to the source mode. Before starting the source handshake, the routine checks if the DAV signal is being driven low by the interface. If it is being driven low, then the command is being re-entered following a ATO timeout error and the first portion of the source handshake is branched over.

The handshake begins with the command placing Register C contents on the bus. The routine then checks the NRFD signal to determine if the devices on the bus are all ready for data. If this signal remains low for more than 100 microseconds, the command returns the RTO timeout error. This does not mean that a hardware error has occurred, only that the devices on the bus are not all currently ready for data.

When the NRFD signal becomes high, the routine checks the NDAC signal. If this signal is also high, it means that there is no active device on the bus and the NDP error is returned. If the NDAC signal is low, DAV is driven low to indicate that there is valid data on the bus. The command then checks the NDAC signal to determine if all the devices on the bus have accepted data. If this signal remains low for more than 1000 microseconds, the ATO timeout error is returned. Again, this does not mean that a hardware error has occurred, but it should be less common than the RTO timeout errors, since few IEEE 488 devices use this portion of the handshake to control their input rate. When the NDAC signal becomes high the DAV signal and data is removed, the A register is cleared, and the command then ends.

The NDP error occurs when there are no devices on the IEEE 488 bus or power is off at all devices. This error leaves data on the bus and should be followed by a Go To Standby command in order to float the bus.

The RTO error occurs when a device does not become ready for data within 100 microseconds. Most IEEE 488 devices can accept interface control messages at or near the full rate of the interface, but there are exceptions. This error can be handled in any of several ways. It can be ignored by continuously re-calling the command until all devices become ready, or the command can be called a fixed number of times (effectively lengthening the timeout) before responding to the error, or the error can be reported immediately. This error also leaves data on the bus and should be followed by a Go To Standby command to float the bus.

The ATO error occurs when a device does not accept data that it has already agreed to accept. If this error persists, a Go To Standby command followed by an UNLISTEN may clear the error, or it may be necessary to command an IFC and initialize the interface. It cannot be determined whether valid data was transferred if the handshake is aborted here.

Output Device Message

Data byte in C Register

7	6	5	4	3	2	1	0	Bit No.
							EOI	B Register

EOI: If "1" the EOI signal will be driven low when the data is on the bus.

7	6	5	4	3	2	1	0	Bit No.
ERR					ATO	RTO	NDP	A Register

NDP: No device present
RTO: Device not ready for data timeout
ATO: Data not accepted timeout
ERR: Set if any error bit is set

This command outputs the byte in Register C as device dependent data (with ATN SIGNAL high). If ATN is active when called, ATN is first driven high and is left high following the data transfer.

This command is called only when in source handshake mode. If bit 0 of Register B is "1", then the EOI signal is driven low. Before starting the source handshake the command checks whether the DAV signal is being driven low, by the interface. If it is being driven low then the command is being re-entered following an ATO timeout error and the first portion of the source handshake is branched over. The handshake begins with the interface replacing the data byte on the bus. The command then checks the NRFD signal to determine whether all LISTENERS are ready for data. If this signal remains low for more than 100 microseconds then the routine returns the RTO timeout error. This does not mean that a hardware error has occurred, but only that a device on the bus is not currently ready for data.

When the NRFD signal becomes high, the command checks the NDAC signal. If this signal is also high at this time it means that there is no active LISTENER on the bus so the routine returns the NDP error. If the NDAC signal is low, the command responds by driving DAV low to identify valid data on the bus. The command then checks the NDAC signal to determine whether all the LISTENERS have accepted the data. If this signal remains low for more than 1000 microseconds, the command returns the ATO timeout error. Again, this does not mean that a hardware error has occurred, but this error should be less common than the RTO timeout error since few IEEE 488 devices use this portion of the handshake to control their input rate. When the NDAC signal becomes high, the command removes the DAV signal and the data, clears Register A, and returns to the calling program.

The NDP error occurs when none of the devices on the bus have been made a LISTENER. This could be caused by using a nonexistent IEEE bus address, or the device addressed not being powered up. This error leaves data on the bus and should be followed by a Go To Standby command to float the bus.

The RTO error occurs when a device does not become ready for data within 100 microseconds. Most IEEE 488 devices cannot accept device-dependent data at the full rate of the interface. This error can be handled in one of several ways. It can be ignored by continually recalling the command until all LISTENER devices become ready, or the command can be called a fixed number of times (effectively lengthening the timeout) before responding to the error, or the error can be reported immediately. As an example, a printer may only be able to accept data at a rate of 30 CPS. If the delay between transfers is much greater than this, it could mean that the printer is off-line or that the paper has jammed. It is possible to check a device and determine whether it is ready for data by attempting to transfer a data byte. If the command returns a timeout error, the device can be issued an UNLISTEN and the interface is then free for other transactions. Since this error leaves data on the bus it should be followed by an UNLISTEN command to free the bus.

The ATO error occurs when a device does not accept data that it has already agreed to accept. If this error persists, a Go To Standby command followed by an UNLISTEN may clear the error, or it may be necessary to command an IFC to initialize the interface. It cannot be determined whether valid data had been transferred if the handshake is aborted here.

INPUT DEVICE MESSAGE

No input parameters

Data bytes in A and H Registers

7	6	5	4	3	2	1	0	Bit No.
ERR					NTO	VTO	EOI	L Register

EOI: Set to "1" if the EOI signal was low when the data was read
VTO: Data valid timeout
NTO: Data not valid timeout
ERR: Set if any error bit is set

This command inputs a device-dependent data byte (with ATN high), reads the EO signal, then returns the results in Registers A, H, and L.

The command first saves the contents of Register HL in Register DE. Register HL will contain input data if the routine is being re-entered following a NTO timeout error. The ENABLE DATA OUT bit (PB0) is checked to determine if the interface is in the source handshake mode or the acceptor handshake mode. If this bit is clear, then the interface is in the source mode and is switched to the acceptor mode. When the interface is in the source handshake mode, the ATN signal will also be low. Signals NRFD and NDAC are driven low before the ATN signal is set high.

Before starting the acceptor handshake the routine checks if the NDAC signal has been set high by the interface. If it is high, then the command is being re-entered following an NTO timeout and the first portion of the acceptor handshake is branched over. The handshake sequence is initiated with the command setting signal NRFD high to indicate that it is ready for data. The command then checks the DAV signal to determine whether the TALKER has placed data on the bus. If this signal remains high for more than 100 microseconds, then the command returns the VTO timeout error. This does not mean that a hardware error has occurred, rather that the TALKER does not currently have data available for transfer.

When the DAV signal becomes low, the command drives signal NRFD high to indicate that it is no longer ready for data. It then reads data and the EOI signal. Next the command sets signal NDAC high to indicate that it has accepted data. The command then checks the DAV signal to determine whether the TALKER has removed its data. If this signal remains low for more than 1000 microseconds, the command returns the NTO timeout error. Again, this does not mean that a hardware error has occurred, but it should be less common than the VTO timeout error since few IEEE 488 devices use this portion of the handshake to control their output rate. When the DAV signal becomes high, the routine drives NDAC low and returns.

The VTO error occurs when a device does not make data available within 100 microseconds. Most IEEE devices will not supply data at this rate. This error can be handled in several ways. It can be ignored by continuously recalling the command until data becomes available, or the command can be called a fixed number of times (effectively lengthening the timeout) before responding to the error, or the error can be reported immediately.

As an example, a keyboard may transfer a character only when a key is pressed. The time separating key entries could be very long. To exit from this error state, it is necessary to UNTALK the slow-responding device. There is a possibility of losing a character from the device when this is done, so if the device supports another method to check for data available, it should be used. Also, if the TALKER places the data on the bus after the error was detected, but before the UNTALK begins, then the ATN will occur on the bus with the data and could be misinterpreted by other devices as an interface message; this will produce unintended state transitions.

The NTO error occurs when the TALKER does not remove the data after that data has been accepted by the controller. This can occur if the controller has enabled other LISTENERS on the bus and not all devices have accepted the data, or if the TALKER is slow in removing the data. If this error persists, an UNTALK to the device must be sent. This could also cause unintended state transitions since the TALKER does have data on the bus. If the UNTALK is not accepted, an IFC should be used to clear this error state. The data returned when the error was reported is valid.

Input Parallel Poll Messages

No input parameters

A = Parallel poll response

This command can only be called in the source handshake mode. The controller may be in the Standby state, but ATN is left low following this command. The routine drives both the ATN and EOI signals low. All devices on the bus with parallel poll capability respond by driving their previously assigned data bit low. The command floats the internal data bus, then reads the parallel poll data. The command restores the source handshake, then ends. The Go To Standby command should be called following this command to float the bus.

IEEE 488 Sample Programs

The following assembly language code contains routines which access each of the IEEE 488 commands within the ROM. This code must be present in any program attempting to make use of the examples presented in this section. Before calling any of these routines, the proper register must have been loaded with the requested parameter. When using the TOROM routine, the high order byte of the address in Register D is always assumed to be 1. Only the low-order byte of the address in the ROM needs to be placed in Register E.

```
;
; Control Out
CO:
        CALL    SETUP     ;setup all the reg. except DE
        LD      L,27H
        LD      (DEREG),HL
        CALL    GOROM
        RET
;
; Status In
SI:
        CALL    SETUP     ;setup all the reg. except DE
        LD      L,2AH
        LD      (DEREG),HL
        CALL    GOROM
        RET
;
; Go To Standby
GTS:
        CALL    SETUP     ;setup all the reg. except DE
        LD      L,2DH
        LD      (DEREG),HL
        CALL    GOROM
        RET

;
; Take Control
TC:
        CALL    SETUP     ;setup all the reg. except DE
        LD      L,30H
        LD      (DEREG),HL
        CALL    GOROM
        RET
;
; Output Interface Message
OIM:
        CALL    SETUP     ;setup all the reg. except DE
        LD      L,33H
        LD      (DEREG),HL
        CALL    GOROM
        RET
;
; Output Device Message
ODM:
        CALL    SETUP     ;setup all the reg. except DE
        LD      L,36H
        LD      (DEREG),HL
        CALL    GOROM
        RET
```

```
;
; Input Device Message
IDM:
        CALL    SETUP      ;setup all the reg. except DE
        LD      L,39H
        LD      (DEREG),HL
        CALL    GOROM
        RET

;
; Parallel Poll
PP:
        CALL    SETUP      ;setup all the reg. except DE
        LD      L,3CH
        LD      (DEREG),HL
        CALL    GOROM
        RET
;
;

;
; This routine will set up the required parameter to perform
; the IEEE-488 functions
; Exit:
;  AREG  = the content of the A register
;  BCREG = the content of the BC registers
;  HLREG = "    "    "   "  HL    "
;

SETUP:
        LD      (AREG),A    ;save the A reg.
        LD      (HLREG),HL  ;save the HL reg.
        LD      (BCREG),BC  ;save the BC reg.
        RET

;
; This routine will pass control to the ROM, via BDOS call
;  50
; Entry:
; AREG, BCREG, DEREG, HLREG must contain the proper
;   parameter
;
GOROM:
        LD      A,30        ;the function number, USERF
        LD      (FUNC),A
        LD      C,32H       ;indirect accessing the BIOS
        LD      DE,BIOSPB   ; point to the BIOS Parameter
                            ;Block
        CALL    5
        RET
```

```
;
;
; Data area
;
BIOSPB:
FUNC:           DB    00
AREG:           DB    00
BCREG:          DW    0000
DEREG:          DW    0000
HLREG:          DW    0000
```

The following sample programs ignore error returns from the IEEE 488 BIOS calls except to re-enter the routine if any error condition exists. A real I/O driver should analyze errors and determine the correct action. Timeout errors might be ignored, but the "no device present" error must generate an I/O error.

Make Listener

There are two alternative versions of the L interface function: one with, and one without address extension. The normal L function uses a 1 byte address. The L function with address extension (the LE function) uses a 2 byte address. This example implements the LE function.

```
;INPUT PARAMETERS:  C    PRIMARY ADDRESS (0-31)
;                   B    SECONDARY ADDRESS (0-31)

ML:     LD      A,C                ;FORM PRIMARY
                                   ;ADDRESS MESSAGE
        ADD     00100000B
        LD      C,A
ML10:   CALL    OIM                ;OUTPUT INTERFACE
                                   ;MESSAGE
        OR      A
        JR      NZ,ML10            ;RE-ENTER ON ERROR
        LD      A,B                ;FORM SECONDARY
                                   ;ADDRESS MESSAGE
        ADD     01100000B
        LD      C,A
ML20:   CALL    OIM                ;OUTPUT INTERFACE
                                   ;MESSAGE
        OR      A
        JR      NZ,ML20            ;RE-ENTER ON ERROR
```

Make Talker

There are two alternative versions of the T interface function: one with and one without address extension. The normal T function uses a 1 byte address. The T function with address extension (the TE function) uses a 2 byte address. This example implements the TE function.

```
;INPUT PARAMETERS   C    PRIMARY ADDRESS (0-31)
;                   B    SECONDARY ADDRESS (0-31)

MT:     LD      A,C                     ;FORM PRIMARY
                                        ;ADDRESS MESSAGE
        ADD     01000000B
        LD      C,A
MT10:   CALL    OIM                     ;OUTPUT INTERFACE
                                        ;MESSAGE
        OR      A
        JR      NZ,MT10                 ;RE-ENTER ON ERROR
        LD      A,B                     ;FORM SECONDARY
                                        ;ADDRESS MESSAGE
        ADD     01100000B
        LD      C,A
```

Second Level Software Description

```
MT20    CALL    OIM                     ;OUTPUT INTERFACE
                                        ;MESSAGE
        OR      A
        JR      NZ,MT20
```

UNLISTEN

```
UL:     LD      C,00111111B             ;UNLISTEN MESSAGE
UL10:   CALL    OIM                     ;OUTPUT INTERFACE
                                        ;MESSAGE
        OR      A
        JR      NZ,UL10                 ;RE-ENTER ON ERROR
```

UNTALK

```
UT:     LD      C,01011111B             ;UNTALK MESSAGE
UT10:   CALL    OIM                     ;OUTPUT INTERFACE
                                        ;MESSAGE
        OR      A
        JR      NZ,UT10                 ;RE-ENTER ON ERROR
```

OUTPUT DATA

```
;INPUT PARAMETERS  C    DATA BYTE
;                  B    EOI

OD:   CALL   ODM                  ;OUTPUT DEVICE
                                  ;MESSAGE
      OR     A
      JR     NZ,OD                ;RE-ENTER ON ERROR
```

INPUT DATA

```
;OUTPUT PARAMETERS  A,H  DATA BYTE
;                   L    EOI

ID:   CALL   IDM                  ;INPUT DEVICE
                                  ;MESSAGE
      BIT    7,L
      JR     NZ,ID                ;RE-ENTER ON ERROR
```

Higher Level Functions

The following examples use the above routines to perform the standard interface functions. This is not meant to be a complete description of the capabilities of the interface.

OUTPUT DATA TO A DEVICE

```
      LD     BC, DEVICE ADDRESS
      CALL   ML                   ;MAKE LISTENER
      LD     BC,DATA              ;DATA AND EOI
      CALL   DO
             ;ANY NUMBER OF DATA BYTES MAY BE SENT
      CALL   UL                   ;UNLISTEN
      CALL   GTS                  ;GO TO STANDBY
```

INPUT DATA FROM A DEVICE

```
      LD     BC,DEVICE ADDRESS
      CALL   MT                   ;MAKE TALKER
      CALL   ID                   ;INPUT DATA AND EOI
      ;ANY NUMBER OF DATA BYTES MAY BE RECEIVED
      CALL   UT                   ;UNTALK
      CALL   GTS                  ;GO TO STANDBY
```

Data Transfer Between Devices (Monitored)

This will make one device a TALKER and another device a LISTENER then let the TALKER control the bus. The controller will monitor the bus and take control when it detects an EOI.

```
       LD     BC,TALKER ADDRESS
       CALL   MT
       LD     BC,LISTEN ADDRESS
       CALL   ML
LOOP:  CALL   ID              ;INPUT DATA AND EOI
       BIT    O,L
       JR     Z,LOOP          ;LOOP UNTIL EOI LOW
       CALL   UT              ;UNTALK
       CALL   UL              ;UNLISTEN
       CALL   GTS             ;GO TO STANDBY
```

Data Transfer Between Devices (Un-monitored)

This will make one device a TALKER and another device a LISTENER then let the TALKER control the bus. The controller will wait a fixed delay then take control synchronously.

Second Level Software Description

```
       LD     BC,TALKER ADDRESS
       CALL   MT
       LD     BC,LISTEN ADDRESS
       CALL   ML
       CALL   GTS             ;GO TO STANDBY
       ;FIXED DELAY WHILE DATA IS TRANSFERED
       LD     C,O
LOOP:  CALL   TC              ;TAKE CONTROL
                              ;SYNCHRONOUSLY
       OR     A
       JR     NZ,LOOP         ;WAIT FOR CONTROL
       CALL   UT              ;UNTALK
       CALL   UL              ;UNLISTEN
       CALL   GTS             ;GO TO STANDBY
```

Serial Poll

This is an example of a serial poll. It assumes that the detection of the service request was performed separately. It is important that every device capable of generating the service request is polled to assure that two devices were not simultaneously driving the service request signal:

```
        LD      C,00011000B
LOOP1:  CALL    OIM             ;SERIAL POLL ENABLE
        OR      A
        JR      NZ,LOOP1        ;RE-ENTER ON ERROR
        LD      BC,TALKER ADDRESS
        CALL    MT              ;MAKE FIRST DEVICE
                                ;A TALKER
        CALL    ID              ;INPUT STATUS BYTE
        CALL    UT              ;UNTALK

        ;ACT ON STATUS
        LD      BC,TALKER ADDRESS
        CALL    MT              ;MAKE SECOND
                                ;DEVICE A TALKER
        CALL    ID              ;INPUT STATUS
        CALL    UT              ;UNTALK
        ;ACT ON STATUS
        LD      C,00011001B
LOOP2:  CALL    OIM             ;SERIAL POLL DISABLE
        OR      A
        JR      NZ,LOOP2        ;RE-ENTER ON ERROR
        CALL    GTS             ;GO TO STANDBY
```

SEND LOCAL LOCK OUT (UNIVERSAL)

```
        LD      C,00000110B
        CALL    CO              ;SET REN LOW
        LD      C,00010001B
LOOP:   CALL    OIM             ;SEND LOCAL LOCK
                                ;OUT
        OR      A
        JR      NZ,LOOP         ;RE-ENTER ON ERROR
        LD      BC,LISTEN ADDRESS
```

Second Level Software Description

```
        CALL    ML              ;FIRST DEVICE GO
                                ;TO REMOTE
        LD      BC,LISTEN ADDRESS
        CALL    ML              ;SECOND DEVICE GO
                                ;TO REMOTE
        CALL    UL              ;UNLISTEN
```

GO TO LOCAL (ADDRESSED)

```
        LD      BC,DEVICE ADDRESS
        CALL    ML                          ;MAKE FIRST DEVICE
                                            ;A LISTENER
        LD      BC,DEVICE ADDRESS
        CALL    ML                          ;MAKE SECOND
                                            ;DEVICE A LISTENER
        LD      C,00000001B
LOOP:   CALL    OIM                         ;GO TO LOCAL
        OR      A
        JR      NZ,LOOP                     ;RE-ENTER ON ERROR
        CALL    UL                          ;UNLISTEN
        CALL    GTS                         ;GO TO STANDBY
```

REMOVE LOCAL LOCK OUT (UNIVERSAL)

```
        LD      C,00000100B
        CALL    CO                          ;SET REN HIGH
```

PARALLEL POLL CONFIGURE

```
        LD      BC,LISTEN ADDRESS
        CALL    ML                          ;MAKE LISTENER
        LD      C,00000101B
LOOP1:  CALL    OIM                         ;PARALLEL POLL
                                            ;CONFIGURE
        OR      A
        JR      NZ,LOOP1                    ;RE-ENTER ON ERROR
        LD      C,0110SPPP
LOOP2:  CALL    OIM                         ;PARALLEL POOL
                                            ;ENABLE
        OR      A
        JR      NZ,LOOP2                    ;RE-ENTER ON ERROR
        CALL    UL                          ;UNLISTEN
        CALL    GTS                         ;GO TO STANDBY
```

PARALLEL POLL UNCONFIGURE (UNIVERSAL)

```
        LD      C,00010101B
LOOP:   CALL    OIM                         ;PARALLEL POLL
                                            ;UNCONFIGURE
        OR      A
        JR      NZ,LOOP                     ;RE-ENTER ON ERROR
        CALL    GTS                         ;GO TO STANDBY
```

PARALLEL POLL DISABLE (ADDRESSED)

Second Level Software Description

```
        LD      BC,LISTEN ADDRESS
        CALL    ML                      ;MAKE LISTENER
        LD      C,00000101B
LOOP1:  CALL    OIM                     ;PARALLEL POLL
                                        ;CONFIGURE
        OR      A
        JR      NZ,LOOP1                ;RE-ENTER ON ERROR
        LD      C,01110000B
LOOP2:  CALL    OIM                     ;PARALLEL POLL
                                        ;DISABLE
        OR      A
        JR      NZ,LOOP2                ;RE-ENTER ON ERROR
        CALL    UL                      ;UNLISTEN
        CALL    GTS                     ;GO TO STANDBY
```

DEVICE CLEAR (UNIVERSAL)

```
        LD      C,00010100B
LOOP:   CALL    OIM                     ;DEVICE CLEAR
        OR      A
        JR      NZ,LOOP                 ;RE-ENTER ON ERROR
        CALL    GTS                     ;GO TO STANDBY
```

DEVICE CLEAR (ADDRESSED)

```
        LD      BC,DEVICE ADDRESS
        CALL    ML                      ;MAKE FIRST DEVICE
                                        ;A LISTENER
        LD      BC,DEVICE ADDRESS
        CALL    ML                      ;MAKE SECOND
                                        ;DEVICE A LISTENER
        LD      C,00000100B
LOOP:   CALL    OIM                     ;SELECTED DEVICE
                                        ;CLEAR
        OR      A
        JR      NZ,LOOP                 ;RE-ENTER ON ERROR
        CALL    UL                      ;UNLISTEN
        CALL    GTS                     ;GO TO STANDBY
```

DEVICE TRIGGER

```
        LD      BC,DEVICE ADDRESS
        CALL    ML                      ;MAKE FIRST DEVICE
                                        ;A LISTENER
        LD      BC,DEVICE ADDRESS
        CALL    ML                      ;MAKE SECOND
                                        ;DEVICE A LISTENER
        LD      C,00001000B
LOOP:   CALL    OIM                     ;GROUP EXECUTE
                                        ;TRIGGER
        OR      A
        JR      NZ,LOOP                 ;RE-ENTER ON ERROR
        CALL    UL                      ;UNLISTEN
        CALL    GTS                     ;GO TO STANDBY
```

OSBORNE
COMPUTER CORPORATION

2F00213-01