



Personal Computer

Technical Reference

6936844



Personal Computer

Technical Reference

First Edition (Revised May 1983)

Use this publication only for the purpose stated in the Preface.

It is possible that this material may contain reference to, or information about, IBM products (machines or programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming, or services in your country.

Changes are periodically made to the information herein; these changes will be incorporated in new editions of this publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this manual at any time and without notice.

PREFACE

The IBM Personal Computer Technical Reference manual describes the hardware design and provides interface information for the IBM Personal Computer. This publication also has information about the basic input/output system (BIOS) and programming support.

The information in this publication is both introductory and for reference, and is intended for hardware and software designers, programmers, engineers, and interested persons who need to understand the design and operation of the computer.

You should be familiar with the use of the Personal Computer, and you should understand the concepts of computer architecture and programming.

This manual has two sections:

"Section 1: Hardware" describes each functional part of the system. This section also has specifications for power, timing, and interface. Programming considerations are supported by coding tables, command codes, and registers.

"Section 2: ROM BIOS and System Usage" describes the basic input/output system and its use. This section also contains the software interrupt listing, a BIOS memory map, descriptions of vectors with special meanings, and a set of low memory maps. In addition, keyboard encoding and usage is discussed.

The publication has six appendixes:

- Appendix A: ROM BIOS Listings
- Appendix B: 8088 Assembly Instruction Set Reference
- Appendix C: Of Characters, Keystrokes, and Color
- Appendix D: Logic Diagrams
- Appendix E: Specifications
- Appendix F: Communications

A glossary and bibliography are included.

Prerequisite Publication:

Guide to Operations for the IBM Personal Computer

Suggested Reading:

BASIC for the IBM Personal Computer

Disk Operating System (DOS) for the IBM Personal Computer

Hardware Maintenance and Service for the IBM Personal Computer

MACRO Assembler for the IBM Personal Computer

Related publications are listed in the bibliography.

TABLE OF CONTENTS

Section 1: Hardware

IBM Personal Computer System Unit	1-3
IBM Keyboard	1-29
IBM Expansion Unit	1-39
IBM 80 CPS Printer	1-53
IBM Printer Adapter	1-77
IBM Monochrome Display and Printer Adapter	1-83
IBM Monochrome Display	1-91
IBM Color/Graphics Monitor Adapter	1-93
IBM Color Display	1-115
IBM 5-1/4" Diskette Drive Adapter	1-117
IBM 5-1/4" Diskette Drive	1-141
Diskettes	1-143
IBM Fixed Disk Drive Adapter	1-145
IBM 10MB Fixed Disk Drive	1-161
IBM Memory Expansion Options	1-163
IBM Game Control Adapter	1-185
IBM Prototype Card	1-191
IBM Asynchronous Communications Adapter	1-197
IBM Synchronous Data Link Control (SDLC) Communications Adapter	1-227
IBM Communications Adapter Cable	1-257

Section 2: ROM BIOS and System Usage

ROM BIOS	2-2
Keyboard Encoding and Usage	2-11
BIOS Cassette Logic	2-21

Appendix A: ROM BIOS Listings	A-1
-------------------------------------	-----

Appendix B: 8088 Assembly Instruction Set Reference	B-1
--	-----

Appendix C: Of Characters, Keystrokes, and Colors	C-1
--	-----

Appendix D: Logic Diagrams	D-1
Appendix E: Specifications	E-1
Appendix F: Communications	F-1
Glossary	G-1
Bibliography	H-1
Index	I-1

INDEX TAB LISTING

Section 1: Hardware

Section 2: ROM BIOS and System Usage

Appendix A: ROM BIOS Listings

Appendix B: 8088 Assembly Instruction
Set Reference

Appendix C: Of Characters, Keystrokes,
and Color

Appendix D: Logic Diagrams

Hardware

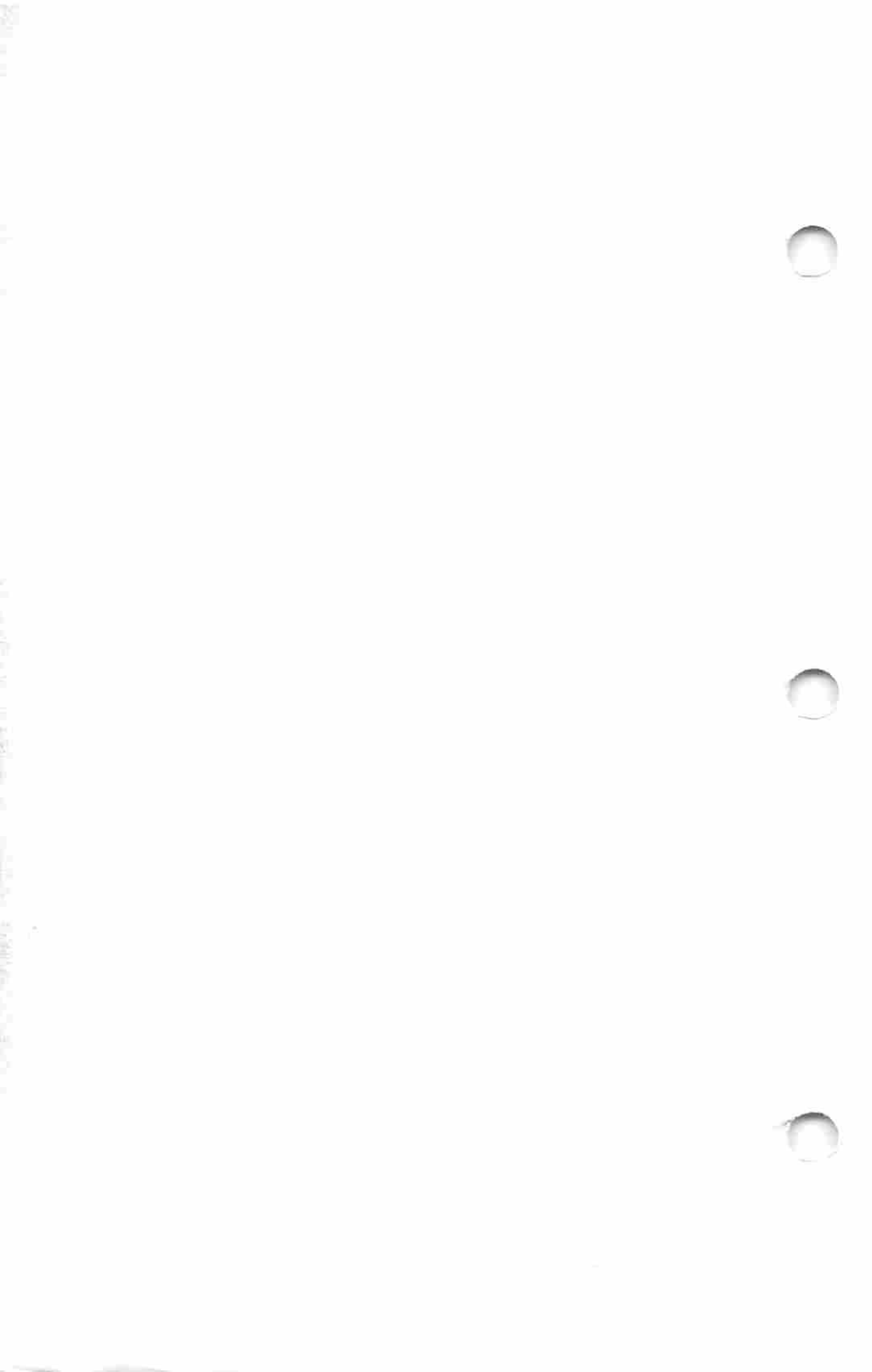
BIOS

Appendix A

Appendix B

Appendix C

Appendix D



Appendix E: Specifications	Appendix E
---	-------------------

Appendix F: Communications	Appendix F
---	-------------------

Glossary	Glossary
-----------------------	-----------------

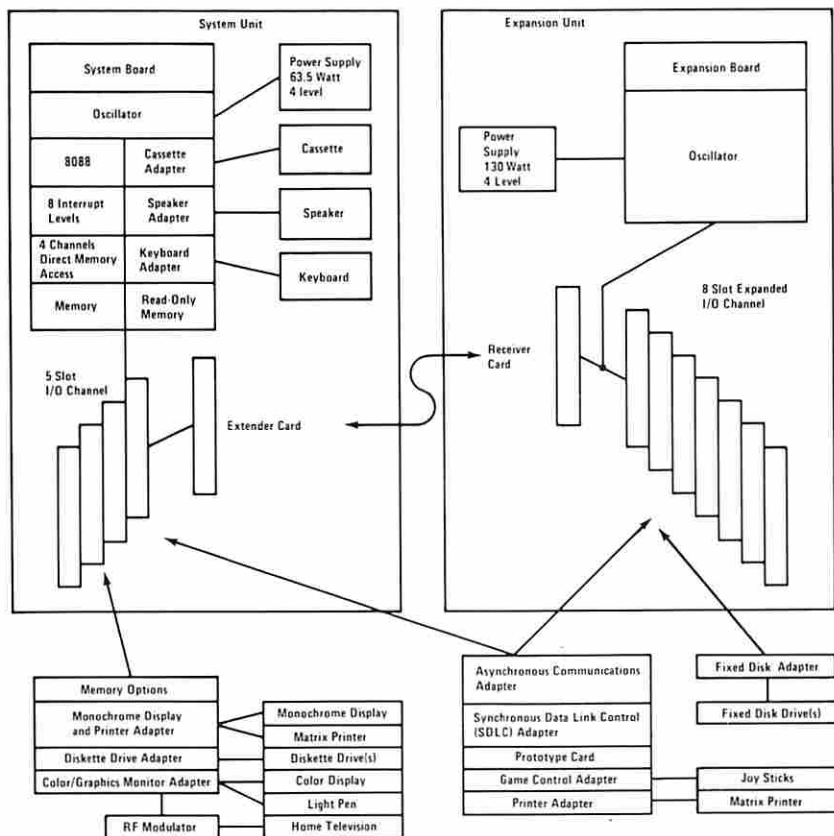
Bibliography	Bibliography
---------------------------	---------------------

Index	Index
--------------------	--------------

Notes:

SECTION 1: HARDWARE

IBM Personal Computer System Unit	1-3
IBM Keyboard	1-29
IBM Expansion Unit	1-39
IBM 80 CPS Printer	1-53
IBM Printer Adapter	1-77
IBM Monochrome Display and Printer Adapter	1-83
IBM Monochrome Display	1-91
IBM Color/Graphics Monitor Adapter	1-93
IBM Color Display	1-115
IBM 5-1/4" Diskette Drive Adapter	1-117
IBM 5-1/4" Diskette Drive	1-141
Diskettes	1-143
IBM Fixed Disk Drive Adapter	1-145
IBM 10MB Fixed Disk Drive	1-161
IBM Memory Expansion Options	1-163
IBM Game Control Adapter	1-185
IBM Prototype Card	1-191
IBM Asynchronous Communications Adapter	1-197
IBM Synchronous Data Link Control (SDLC) Communications Adapter	1-227
IBM Communications Adapter Cable	1-257



System Block Diagram

IBM Personal Computer System Unit

The system unit is the standalone tabletop unit that contains the power supply, the speaker, and the system board.

The system unit contains one of two system boards. One system board supports 16K to 64K of R/W memory. The other system board supports 64K to 256K of R/W memory. Both system boards are functionally identical.

The power supply provides dc voltage to the system board and the internal drive(s).

System Board

The system board fits horizontally in the base of the system unit and is approximately 8-1/2 by 12 inches. It is a multilayer, single-land-per-channel design with ground and internal planes provided. DC power and a signal from the power supply enter the board through two six-pin connectors. Other connectors on the board are for attaching the keyboard, audio cassette, and speaker. Five 62-pin card edge-sockets are also mounted on the board. The I/O channel is bussed across these five I/O slots.

Two dual-in-line package (DIP) switches (two eight-switch packs) are mounted on the board and can be read under program control. The DIP switches provide the system software with information about the installed options, how much storage the system board has, what type of display adapter is installed, what operation modes are desired when power is switched on (color or black-and-white, 80- or 40-character lines), and the number of diskette drives attached.

The system board consists of five functional areas: the processor subsystem and its support elements, the read-only memory (ROM) subsystem, the read/write (R/W) memory subsystem, integrated I/O adapters, and the I/O channel. All are described in this section.

The heart of the system board is the Intel 8088 microprocessor. This processor is an 8-bit external bus version of Intel's 16-bit 8086 processor, and is software-compatible with the 8086. Thus, the 8088 supports 16-bit operations, including multiply and divide, and supports 20 bits of addressing (1 megabyte of storage). It also operates in maximum mode, so a co-processor can be added as a feature. The processor operates at 4.77 MHz. This frequency, which is derived from a 14.31818-MHz crystal, is divided by 3 for the processor clock, and by 4 to obtain the 3.58-MHz color burst signal required for color televisions.

At the 4.77-MHz clock rate, the 8088 bus cycles are four clocks of 210 ns, or 840 ns. I/O cycles take five 210-ns clocks or 1.05 microseconds.

The processor is supported by a set of high-function support devices providing four channels of 20-bit direct-memory access (DMA), three 16-bit timer-counter channels, and eight prioritized interrupt levels.

Three of the four DMA channels are available on the I/O bus and support high-speed data transfers between I/O devices and memory without processor intervention. The fourth DMA channel is programmed to refresh the system dynamic memory. This is done by programming a channel of the timer-counter device to periodically request a dummy DMA transfer. This action creates a memory-read cycle, which is available to refresh dynamic storage both on the system board and in the system expansion slots. All DMA data transfers, except the refresh channel, take five processor clocks of 210 ns, or 1.05 μ s if the processor-ready line is not deactivated. Refresh DMA cycles take four clocks or 840 ns.

The three programmable timer/counters are used by the system as follows: Channel 0 is used as a general-purpose timer providing a constant time base for implementing a time-of-day clock; Channel 1 is used to time and request refresh cycles from the DMA channel; and Channel 2 is used to support the tone generation for the audio speaker. Each channel has a minimum timing resolution of 1.05 μ s.

Of the eight prioritized levels of interrupt, six are bussed to the system expansion slots for use by feature cards. Two levels are used on the system board. Level 0, the highest priority, is attached to Channel 0 of the timer/counter and provides a periodic interrupt for the time-of-day clock. Level 1 is attached to the keyboard adapter circuits and receives an interrupt for each scan code sent by the keyboard. The non-maskable interrupt (NMI) of the 8088 is used to report memory parity errors.

The system board supports both ROM and R/W memory. It has space for 48K x 8 of ROM or EPROM. Six module sockets are provided, each of which can accept an 8K by 8 byte device. Five of the sockets are populated with 40K bytes of ROM. This ROM contains the cassette BASIC interpreter, cassette operating system, power-on self-test, I/O drivers, dot patterns for 128 characters in graphics mode, and a diskette bootstrap loader. The ROM is packaged in 24-pin modules and has an access time of 250 ns and a cycle time of 375 ns.

The difference between the R/W memory on the two system boards is shown in the following chart.

System Board	Minimum Storage	Maximum Storage	Memory Modules	Soldered (Bank 0)	Pluggable (Banks 1-3)
16/64K	16K	64K	16K by 1 Bit	1 Bank of 9	3 Banks of 9
64/256K	64K	256K	64K by 1 Bit	1 Bank of 9	3 Banks of 9

Memory greater than either system board's maximum is obtained by adding memory cards in the expansion slots. All memory is parity-checked and consists of dynamic 16K by 1 bit or 64K by 1 bit modules with an access time of 250 ns and a cycle time of 410 ns.

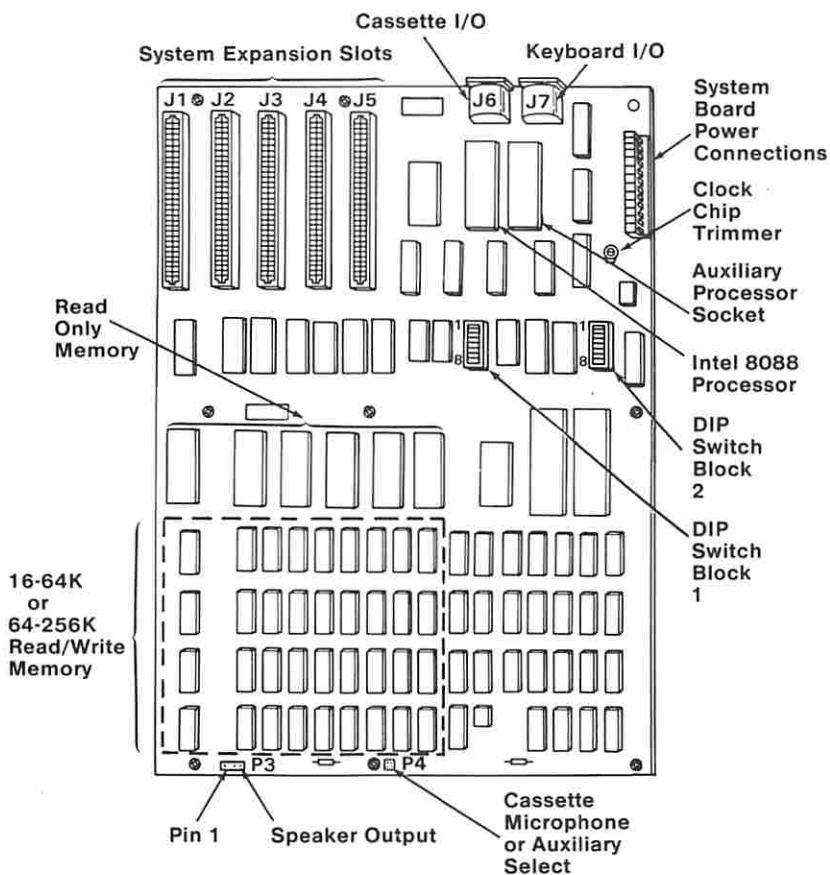
The system board contains circuits for attaching an audio cassette, the keyboard, and the speaker. The cassette adapter allows the attachment of any good quality audio cassette through the earphone output and either the microphone or auxiliary inputs. The system board has a jumper for either input. This interface also provides a cassette motor control line for transport starting and stopping under program control. This interface reads and writes the audio cassette at a data rate of between 1,000 and 2,000 baud. The baud rate is variable and dependent on data content, because a different bit-cell time is used for 0's and 1's. For diagnostic purposes, the tape interface can loop read to write for testing the system board's circuits. The ROM cassette software blocks cassette data and generates a cyclic redundancy check (CRC) to check this data.

The system board contains the adapter circuits for attaching the serial interface from the keyboard. These circuits generate an interrupt to the processor when a complete scan code is received. The interface can request execution of a diagnostic test in the keyboard.

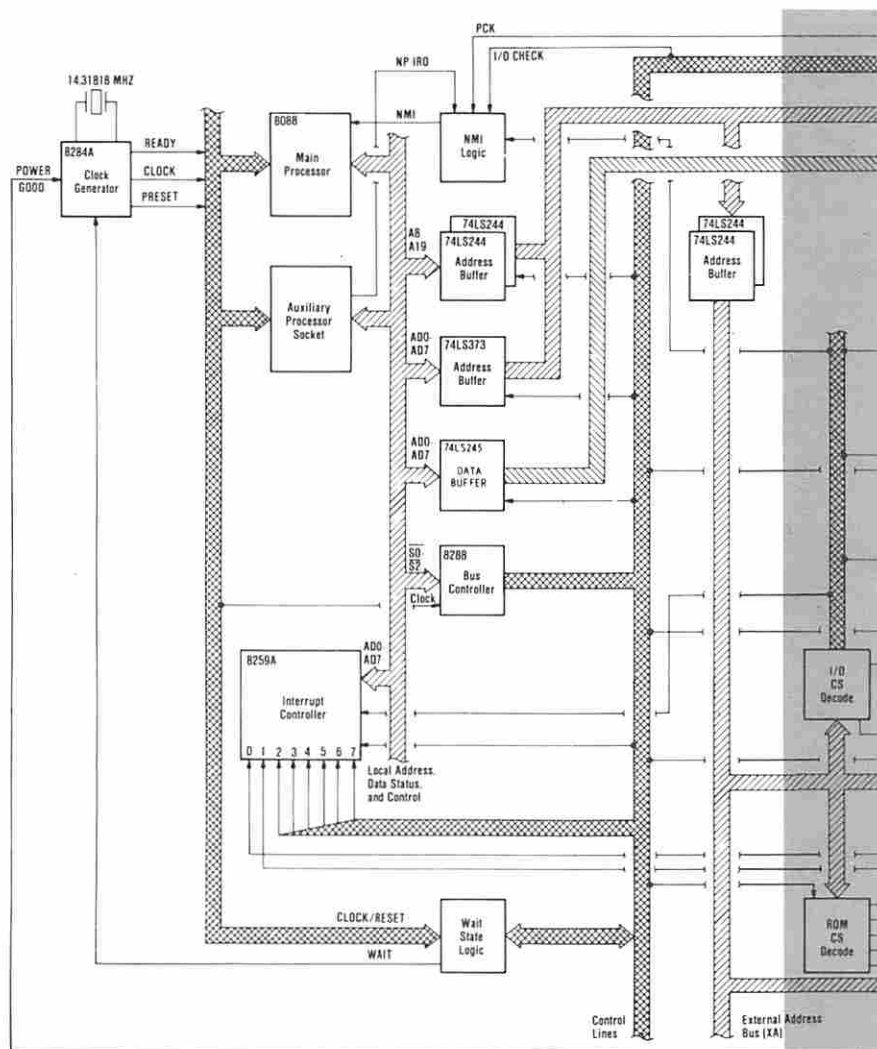
Both the keyboard and cassette interfaces are 5-pin DIN connectors on the system board that extend through the rear panel of the system unit.

The system unit has a 2-1/4 inch audio speaker. The speaker's control circuits and driver are on the system board. The speaker connects through a 2-wire interface that attaches to a 3-pin connector on the system board.

The speaker drive circuit is capable of approximately 1/2 watt of power. The control circuits allow the speaker to be driven three different ways: 1.) a direct program control register bit may be toggled to generate a pulse train; 2.) the output from Channel 2 of the timer counter may be programmed to generate a waveform to the speaker; 3.) the clock input to the timer counter can be modulated with a program-controlled I/O register bit. All three methods may be performed simultaneously.



System Board Component Diagram



System Board Data Flow (Part 1 of 2)

Hex Port Number 0060	I	PA0	+Keyboard Scan Code	0	Or	IPL 5-1/4 Diskette Drive	(SW1-1)																
	N	1		1		Reserved	(SW1-2)																
	P	2		2		System Board Read/Write Memory Size	*(SW1-3)																
	U	3		3		System Board Read/Write Memory Size	*(SW1-4)																
	T	4		4		+Display Type 1	** (SW1-5)																
		5		5		+Display Type 2	** (SW1-6)																
		6		6		No. of 5-1/4 Drives	*** (SW1-7)																
	7		7	No. of 5-1/4 Drives	*** (SW1-8)																		
0061	O	PB0	+Timer 2 Gate Speaker																				
		1	+Speaker Data																				
	U	2	+(Read Read/Write Memory Size) or (Read Spare Key)																				
	T	3	+Cassette Motor Off																				
	P	4	-Enable Read/Write Memory																				
	U	5	-Enable I/O Channel Check																				
	T	6	-Hold Keyboard Clock Low																				
	7	-(Enable Keyboard) or + (Clear Keyboard and Enable Sense Switches)																					
0062	I	PC0	I/O Read/Write Memory (Sw2-1)		Binary Value X 32K	Or	I/O Read/ Write Memory (Sw2-5)																
	N	1	I/O Read/Write Memory (Sw2-2)																				
	P	2	I/O Read/Write Memory (Sw2-3)																				
	U	3	I/O Read/Write Memory (Sw2-4)																				
		4	+Cassette Data In																				
		5	+Timer Channel 2 Out																				
		6	+I/O Channel Check																				
	7	+Read/Write Memory Parity Check																					
0063 Command/Mode Register																							
Mode Register Value			<table> <tr><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>1</td></tr> </table>					7	6	5	4	3	2	1	0	1	0	0	1	1	0	0	1
7	6	5	4	3	2	1	0																
1	0	0	1	1	0	0	1																

Hex 99

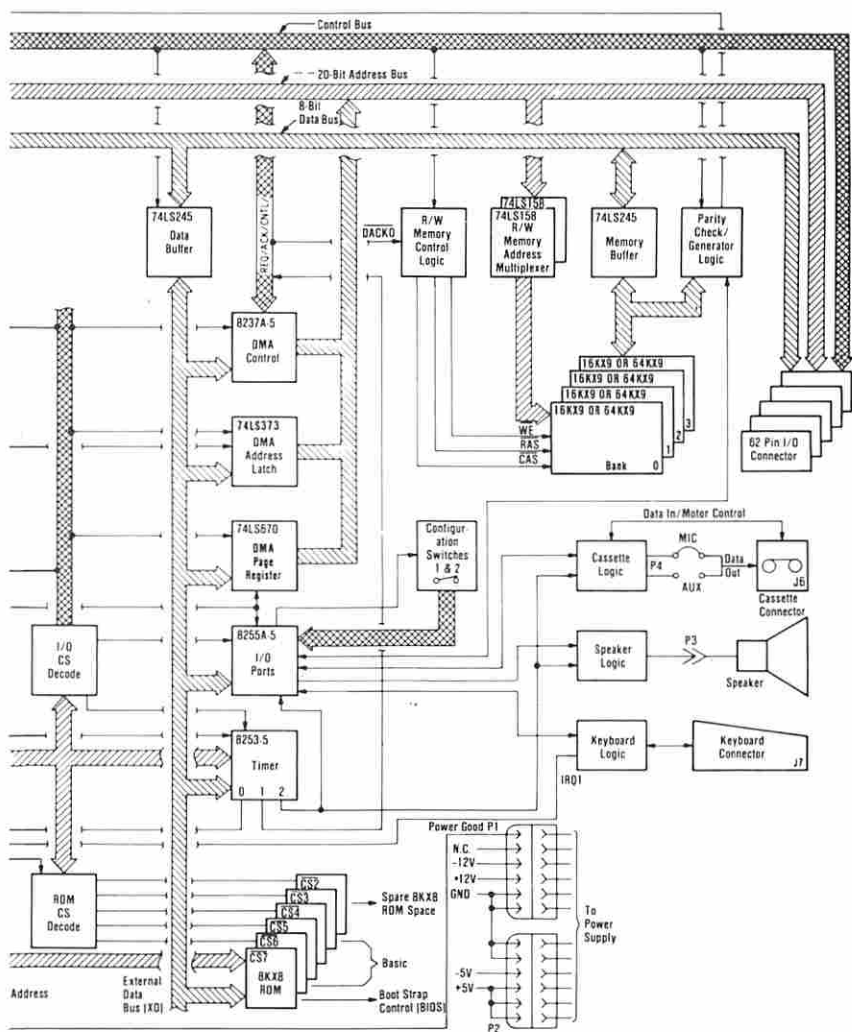
*	PA3	PA2	Amount of Memory Located on System Board	
	Sw1-4	Sw1-3		
	0	0		16K
	0	1		32K
	1	0		48K
	1	1	64 to 256K	
**	PA5	PA4	Display at Power-Up Mode	
	Sw1-6	Sw1-5		
	0	0		No Display
	0	1		Color 40 X 25 (BW Mode)
	1	0		Color 80 X 25 (BW Mode)
	1	1	IBM Monochrome (80 X 25)	
***	PA7	PA6	Number of 5-1/4" Drives in System	
	Sw1-8	Sw1-7		
	0	0		1
	0	1		2
	1	0		3
	1	1	4	

Note: A plus (+) indicates a bit value of 1 performs the specified function.
A minus (-) indicates a bit value of 0 performs the specified function.
PA Bit = 0 implies switch "ON." PA Bit = 1 implies switch "OFF."

8255A I/O Bit Map

Start Address		Function
Decimal	Hex	
0	00000	16 to 64K Read/Write Memory on System Board
16K	04000	
32K	08000	
48K	0C000	
64K	10000	Up to 576K Read/Write Memory in I/O Channel
80K	14000	
96K	18000	
112K	1C000	
128K	20000	
144K	24000	
160K	28000	
176K	2C000	
192K	30000	
208K	34000	
224K	38000	
240K	3C000	
256K	40000	
272K	44000	
288K	48000	
304K	4C000	
320K	50000	
336K	54000	
352K	58000	
368K	5C000	
384K	60000	
400K	64000	
416K	68000	
432K	6C000	
448K	70000	
464K	74000	
480K	78000	
496K	7C000	
512K	80000	
528K	84000	
544K	88000	
560K	8C000	
576K	90000	
592K	94000	
608K	98000	
624K	9C000	

System Memory Map for 16/64K System Board (Part 1 of 2)



System Board Data Flow (Part 2 of 2)

Hex Range	Usage
000-00F	DMA Chip 8237A-5
020-021	Interrupt 8259A
040-043	Timer 8253-5
060-063	PPI 8255A-5
080-083	DMA Page Registers
0A _x *	NMI Mask Register
0C _x	Reserved
0E _x	Reserved
100-1FF	Not Usable
200-20F	Game Control
210-217	Expansion Unit
220-24F	Reserved
278-27F	Reserved
2F0-2F7	Reserved
2F8-2FF	Asynchronous Communications (Secondary)
300-31F	Prototype Card
320-32F	Fixed Disk
378-37F	Parallel Printer
380-38F	SDLC Communications
3A0-3AF	Reserved
3B0-3BF	IBM Monochrome Display/Printer
3C0-3CF	Reserved
3D0-3DF	Color/Graphics
3E0-3E7	Reserved
3F0-3F7	Diskette
3F8-3FF	Asynchronous Communications (Primary)

*At power-on time, the Non Mask Interrupt (NMI) into the 8088 is masked off. This mask bit can be set and reset through system software as follows:

Set mask: write hex 80 to I/O Address hex A0 (enable NMI).

Clear mask: write hex 00 to I/O Address hex A0 (disable NMI).

I/O Address Map

Number	Usage
NMI	Parity
0	Timer
1	Keyboard
2	Reserved
3	Asynchronous Communications (Secondary)
	SDLC Communications
4	Asynchronous Communications (Primary)
	SDLC Communications
5	Fixed Disk
6	Diskette
7	Parallel Printer

8088 Hardware Interrupt Listing

Start Address		Function
Decimal	Hex	
640K	A0000	128K Reserved
656K	A4000	
672K	A8000	
688K	AC000	
704K	B0000	Monochrome
720K	B4000	
736K	B8000	Color/Graphics
752K	BC000	
768K	C0000	192K Read Only Memory Expansion and Control
784K	C4000	
800K	C8000	
816K	CC000	
832K	D0000	
848K	D4000	
864K	D8000	
880K	DC000	
896K	E0000	
912K	E4000	
928K	E8000	
944K	EC000	
960K	F0000	Reserved
976K	F4000	48K Base System ROM
992K	F8000	
1008K	FC000	

System Memory Map for 16/64K System Board (Part 2 of 2)

Start Address		Function
Decimal	Hex	
0	00000	64 to 256K Read/Write Memory on System Board
16K	04000	
32K	08000	
48K	0C000	
64K	10000	
80K	14000	
96K	18000	
112K	1C000	
128K	20000	
144K	24000	
160K	28000	
176K	2C000	
192K	30000	
208K	34000	
224K	38000	
240K	3C000	
256K	40000	Up to 384K Read/Write Memory in I/O Channel
272K	44000	
288K	48000	
304K	4C000	
320K	50000	
336K	54000	
352K	58000	
368K	5C000	
384K	60000	
400K	64000	
416K	68000	
432K	6C000	
448K	70000	
464K	74000	
480K	78000	
496K	7C000	
512K	80000	
528K	84000	
544K	88000	
560K	8C000	
576K	90000	
592K	94000	
608K	98000	
624K	9C000	

System Memory Map for 64/256K System Board (Part 1 of 2)

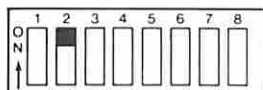
Start Address		Function
Decimal	Hex	
640K	A0000	128K Reserved
656K	A4000	
672K	A8000	
688K	AC000	
704K	B0000	Monochrome
720K	B4000	
736K	B8000	Color/Graphics
752K	BC000	
768K	C0000	Fixed Disk Control
784K	C4000	
800K	C8000	
816K	CC000	
832K	D0000	192K Read Only Memory Expansion and Control
848K	D4000	
864K	D8000	
880K	DC000	
896K	E0000	
912K	E4000	
928K	E8000	
944K	EC000	
960K	F0000	Reserved
976K	F4000	48K Base System ROM
992K	F8000	
1008K	FC000	

System Memory Map for 64/256K System Board (Part 2 of 2)

System Board Switch Settings

The switches on the system board are set as shown in the following figures. These settings are necessary for the system to address the attached components, and to specify the amount of memory installed on the system board.

Switch Block 1

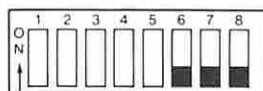


Position

Function

- | | |
|-------|--|
| 1-7-8 | Number of 5-1/4 inch diskette drives installed |
| 2 | With coprocessor installed — off |
| | Without coprocessor — on |
| 3-4 | Amount of memory on system board |
| 5-6 | Type(s) of display adapter(s) you are using |

Switch Block 2



Position

Function

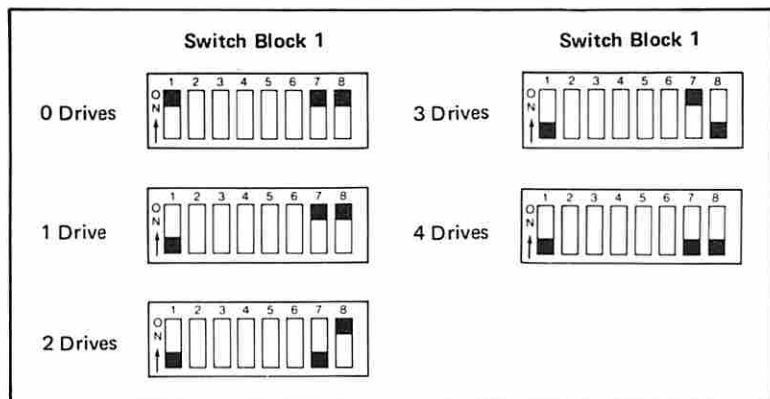
- | | |
|-----------|------------------------------------|
| 1-2-3-4-5 | Amount of memory options installed |
| 6-7-8 | Must be in the off position |

Amount of Memory on System Board	Switch Block 1 Setting	Switch Block 2 Setting
16K		
32K		
48K		
64K		

16/64K System Board Memory Switch Settings

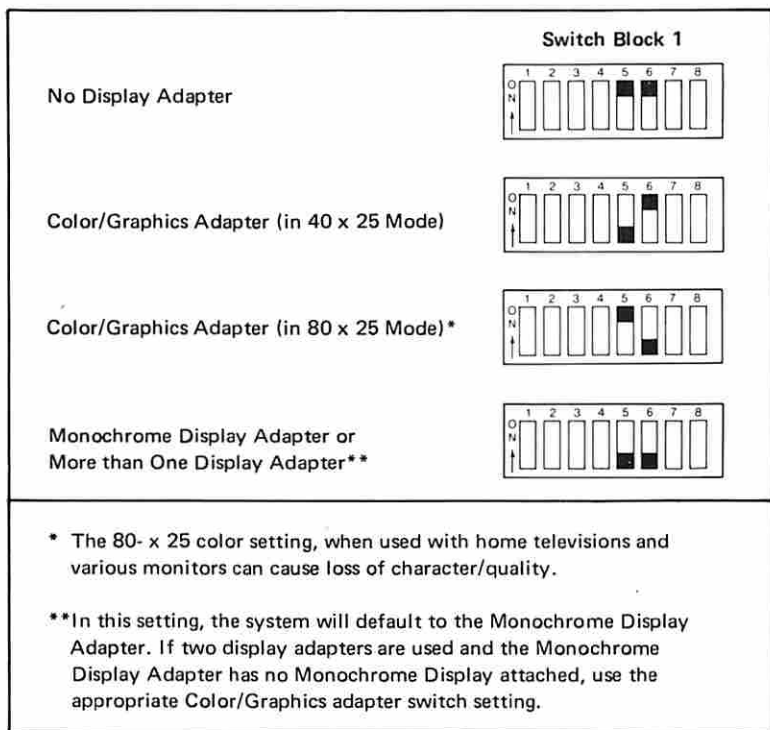
Amount of Memory on System Board	Switch Block 1 Setting	Switch Block 2 Setting
64K		
128K		
192K		
256K		

64/256K System Board Memory Switch Settings



Diskette Drive Switch Settings

Warning: If an IBM Monochrome Display is connected to your system, switch-1 positions 5 and 6 must always be Off. Damage to your display can result with any other switch settings.



Display Adapter Type Switch Settings

I/O Channel

The I/O channel is an extension of the 8088 microprocessor bus. It is, however, demultiplexed, repowered, and enhanced by the addition of interrupts and direct memory access (DMA) functions.

The I/O channel contains an 8-bit, bidirectional data bus, 20 address lines, 6 levels of interrupt, control lines for memory and I/O read or write, clock and timing lines, 3 channels of DMA control lines, memory refresh timing control lines, a channel-check line, and power and ground for the adapters. Four voltage levels are provided for I/O cards: +5 Vdc, -5 Vdc, +12 Vdc, and -12 Vdc. These functions are provided in a 62-pin connector with 100-mil card tab spacing.

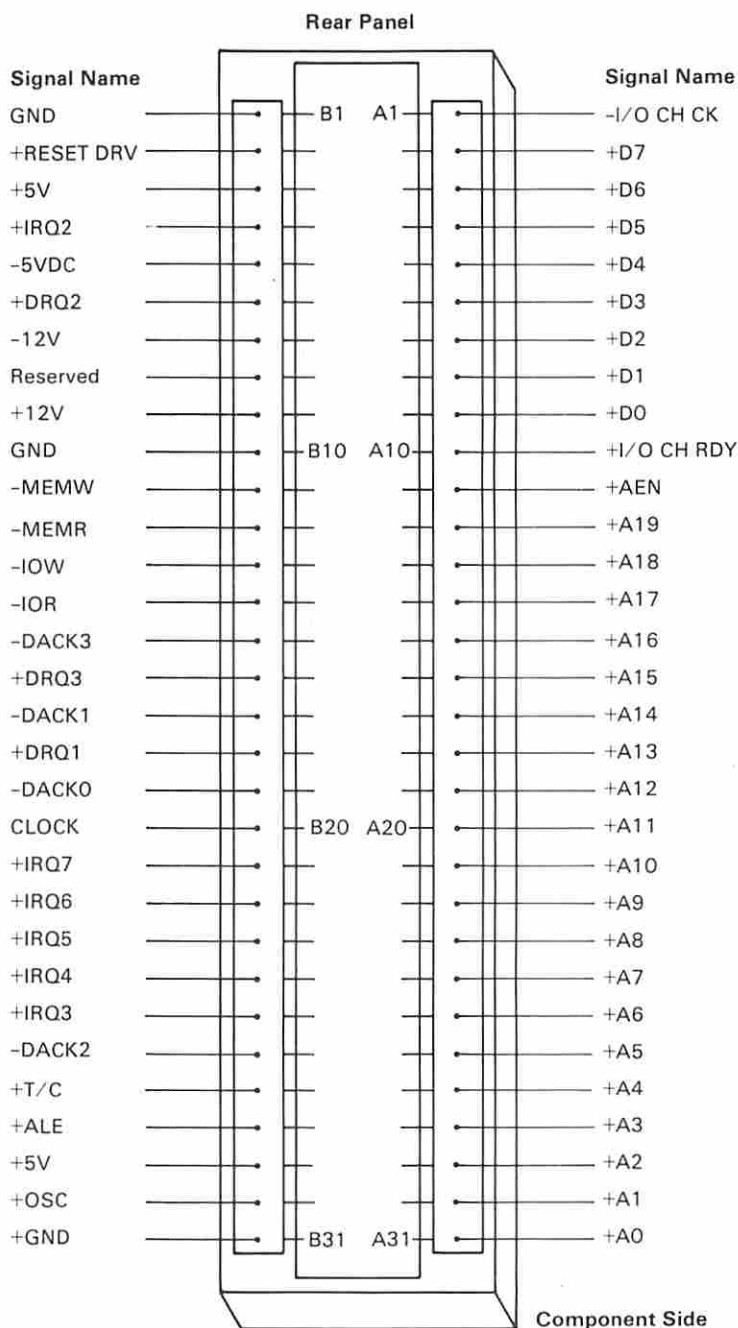
A 'ready' line is available on the I/O channel to allow operation with slow I/O or memory devices. If the channel's ready line is not activated by an addressed device, all processor-generated memory read and write cycles take four 210-ns clock or 840-ns/byte. All processor-generated I/O read and write cycles require five clocks for a cycle time of 1.05 μ s/byte. All DMA transfers require five clocks for a cycle time of 1.05 μ s/byte. Refresh cycles occur once every 72 clocks (approximately 15 μ s) and require four clocks or approximately 7% of the bus bandwidth.

I/O devices are addressed using I/O mapped address space. The channel is designed so that 512 I/O device addresses are available to the I/O channel cards.

A 'channel check' line exists for reporting error conditions to the processor. Activating this line results in a Non-Maskable Interrupt (NMI) to the 8088 processor. Memory expansion options use this line to report parity errors.

The I/O channel is repowered to provide sufficient drive to power all five system unit expansion slots, assuming two low-power Schottky loads per slot. The IBM I/O adapters typically use only one load.

The following pages describe the system board's I/O channel.



I/O Channel Diagram

I/O Channel Description

The following is a description of the IBM Personal Computer I/O Channel. All lines are TTL-compatible.

Signal	I/O	Description
OSC	O	Oscillator: High-speed clock with a 70-ns period (14.31818 MHz). It has a 50% duty cycle.
CLK	O	System clock: It is a divide-by-three of the oscillator and has a period of 210 ns (4.77 MHz). The clock has a 33% duty cycle.
RESET DRV	O	This line is used to reset or initialize system logic upon power-up or during a low line voltage outage. This signal is synchronized to the falling edge of clock and is active high.
A0-A19	O	Address bits 0 to 19: These lines are used to address memory and I/O devices within the system. The 20 address lines allow access of up to 1 megabyte of memory. A0 is the least significant bit (LSB) and A19 is the most significant bit (MSB). These lines are generated by either the processor or DMA controller. They are active high.
D0-D7	I/O	Data Bits 0 to 7: These lines provide data bus bits 0 to 7 for the processor, memory, and I/O devices. D0 is the least significant bit (LSB) and D7 is the most significant bit (MSB). These lines are active high.
ALE	O	Address Latch Enable: This line is provided by the 8288 Bus Controller and is used on the system board to latch valid addresses from the processor. It is available to the I/O channel as an indicator of a valid processor address (when used with AEN). Processor addresses are latched with the falling edge of ALE.
$\overline{\text{I/O CH CK}}$	I	-I/O Channel Check: This line provides the processor with parity (error) information on memory or devices in the I/O channel. When this signal is active low, a parity error is indicated.

Signal	I/O	Description
I/O CH RDY	I	I/O Channel Ready: This line, normally high (ready), is pulled low (not ready) by a memory or I/O device to lengthen I/O or memory cycles. It allows slower devices to attach to the I/O channel with a minimum of difficulty. Any slow device using this line should drive it low immediately upon detecting a valid address and a read or write command. This line should never be held low longer than 10 clock cycles. Machine cycles (I/O or memory) are extended by an integral number of CLK cycles (210 ns).
IRQ2-IRQ7	I	Interrupt Request 2 to 7: These lines are used to signal the processor that an I/O device requires attention. They are prioritized with IRQ2 as the highest priority and IRQ7 as the lowest. An Interrupt Request is generated by raising an IRQ line (low to high) and holding it high until it is acknowledged by the processor (interrupt service routine).
$\overline{\text{IOR}}$	O	-I/O Read Command: This command line instructs an I/O device to drive its data onto the data bus. It may be driven by the processor or the DMA controller. This signal is active low.
$\overline{\text{IOW}}$	O	-I/O Write Command: This command line instructs an I/O device to read the data on the data bus. It may be driven by the processor or the DMA controller. This signal is active low.
$\overline{\text{MEMR}}$	O	Memory Read Command: This command line instructs the memory to drive its data onto the data bus. It may be driven by the processor or the DMA controller. This signal is active low.
$\overline{\text{MEMW}}$	O	Memory Write Command: This command line instructs the memory to store the data present on the data bus. It may be driven by the processor or the DMA controller. This signal is active low.

Signal	I/O	Description
DRQ1-DRQ3	I	DMA Request 1 to 3: These lines are asynchronous channel requests used by peripheral devices to gain DMA service. They are prioritized with DRQ3 being the lowest and DRQ1 being the highest. A request is generated by bringing a DRQ line to an active level (high). A DRQ line must be held high until the corresponding DACK line goes active.
<u>DACK0-</u> <u>DACK3</u>	O	-DMA Acknowledge 0 to 3: These lines are used to acknowledge DMA requests (DRQ1-DRQ3) and to refresh system dynamic memory (DACK0). They are active low.
AEN	O	Address Enable: This line is used to de-gate the processor and other devices from the I/O channel to allow DMA transfers to take place. When this line is active (high), the DMA controller has control of the address bus, data bus, read command lines (memory and I/O), and the write command lines (memory and I/O).
T/C	O	Terminal Count: This line provides a pulse when the terminal count for any DMA channel is reached. This signal is active high.

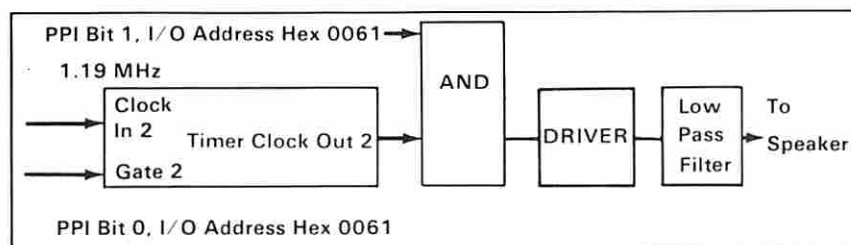
The following voltages are available on the system-board I/O channel:

- +5 Vdc $\pm$ 5%, located on 2 connector pins
- 5 Vdc $\pm$ 10%, located on 1 connector pin
- +12 Vdc $\pm$ 5%, located on 1 connector pin
- 12 Vdc $\pm$ 10%, located on 1 connector pin
- GND (Ground), located on 3 connector pins

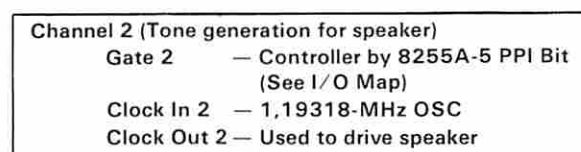
Speaker Interface

The sound system has a small, permanent-magnet, 2-1/4 inch speaker. The speaker can be driven from one or both of two sources:

- An 8255A-5 PPI output bit. The address and bit are defined in the "I/O Address Map."
- A timer clock channel, the output of which is programmable within the functions of the 8253-5 timer when using a 1.19-MHz clock input. The timer gate also is controlled by an 8255A-5 PPI output-port bit. Address and bit assignment are in the "I/O Address Map."



Speaker Drive System Block Diagram



Speaker Tone Generation

The speaker connection is a 4-pin Berg connector. See "System Board Component Diagram," earlier in this section, for speaker connection or placement.

Pin	Function
1	Data
2	Key
3	Ground
4	+5 Volts

Speaker Connector

Power Supply

The system power supply is located at the right rear of the system unit. It is designed to be an integral part of the system-unit chassis. Its housing provides support for the rear panel, and its fan furnishes cooling for the whole system.

It supplies the power and reset signal necessary for the operation of the system board, installable options, and the keyboard. It also provides a switched ac socket for the IBM Monochrome Display and two separate connectors for power to the 5-1/4 inch diskette drives.

It is a dc-switching power supply designed for continuous operation at 63.5 watts. It is available with either a 120 or a 220/240 Vac fused input and provides four regulated dc output voltages: 7 A of +5 Vdc, 2 A of +12 Vdc, 0.3 A of -5 Vdc, and 0.25 A of -12 Vdc. These outputs are over-voltage, over-current, open-circuit, and short-circuit protected. If a dc overload or over-voltage condition occurs, all dc outputs are shut down as long as the condition exists.

The +5 Vdc powers the logic on the system board and the diskette drives and allows approximately 4 A of +5 Vdc for the adapters in the system-unit expansion slots. The +12 Vdc power level is designed to power the system's dynamic memory and the two internal 5-1/4 inch diskette drive motors. It is assumed that only one drive is active at a time. The -5 Vdc level is designed for dynamic memory bias voltage; it tracks the +5 Vdc and +12 Vdc very quickly at power-on and has a longer decay on power-off than the +5 Vdc and +12 Vdc outputs. The +12 Vdc and -12 Vdc levels are used for powering the EIA drivers on the communications adapters. All four of the power supply dc voltages are bussed across each of the five system-unit expansion slots.

Operating Characteristics

Input Requirements

The following are the input requirements for the system unit power supply.

Voltage (Vac)			Frequency (Hz)	Current (Amps)
Nominal	Minimum	Maximum	+/- 3 Hz	Maximum
120	104	127	60	2.5 at 104 Vac
220/240	180	259	50	1.0 at 180 Vac

Vdc Output

The following are the dc outputs for the system unit power supply.

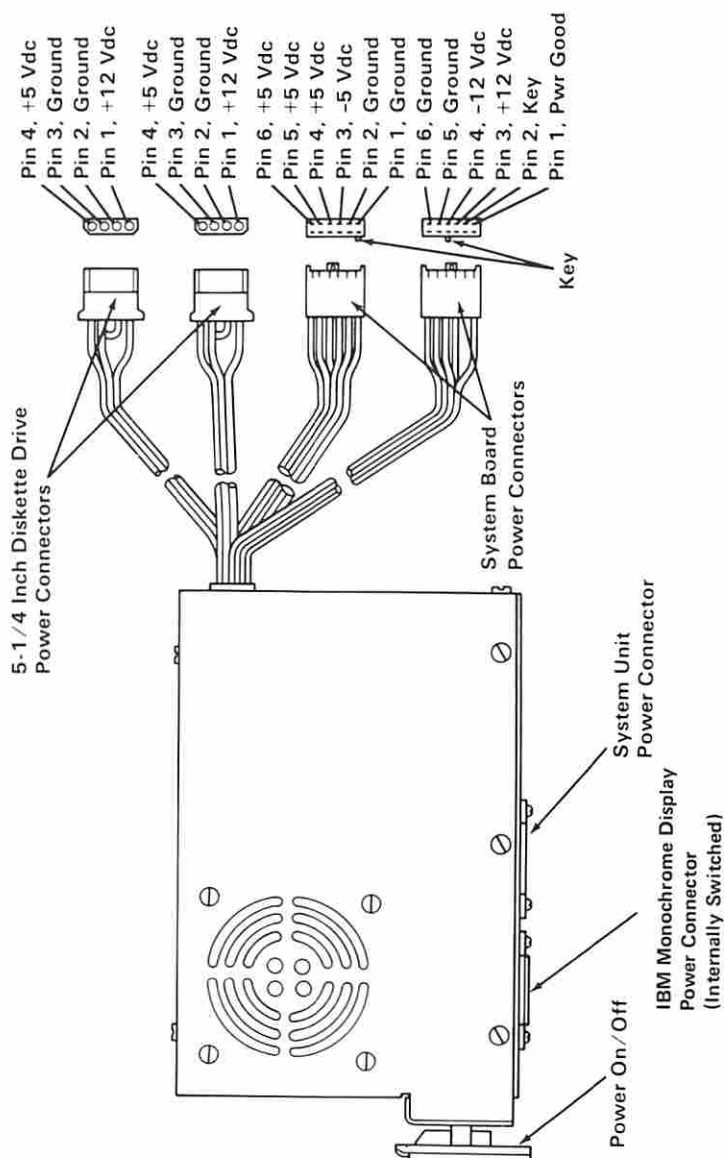
Voltage (Vdc)	Current (Amps)		Regulation (Tolerance)	
Nominal	Minimum	Maximum	+	-
+5.0	2.3	7.0	5	4
-5.0	0.0	0.3	10	8
+12.0	0.4	2.0	5	4
-12.0	0.0	0.25	10	9

Vac Output

The power supply provides a filtered, ac output that is switched on and off with the main power switch. The maximum current available at this output is 0.75 A for the 120 volt supply and 0.38 A for the 220/240 volt supply. The receptacle provided at the rear of the power supply for this ac output is a nonstandard connector designed to be used only for the IBM Monochrome Display.

Power Supply Connectors and Pin Assignments

The power connector on the system board is a 12-pin male connector that plugs into the power-supply connectors. The pin configurations and locations are shown below:



Power Supply and Connectors

Over-Voltage/Over-Current Protection

The system power supply employs protection features which are described below.

Primary (Input)

The following table describes the primary (input voltage) protection for the system-unit power supply.

Voltage Nominal Vac	Type Protection	Rating Amps
120	Fuse	2
220/240	Fuse	1

Secondary (Output)

On over-voltage, the power supply is designed to shut down all outputs when either the +5 Vdc or the +12 Vdc output exceeds 200% of its maximum rated voltage. On over-current, the supply will turn off if any output exceeds 130% of its nominal value.

Power-Good Signal

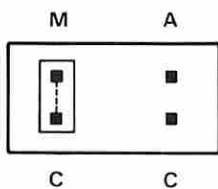
When the power supply is turned on after it has been off for a minimum of 5 seconds, it generates a power-good signal which indicates that there is adequate power for processing. When the four output voltages are above the minimum sense levels, as described below, the signal sequences to a TTL-compatible up level (2.4 Vdc to 5.5 Vdc), which is capable of sourcing 60 μ A. When any of the four output voltages is below its minimum sense level or above its maximum sense level, the power good signal will be a TTL-compatible down level (0.0 Vdc to 0.4 Vdc) capable of sourcing 500 μ A. The power good signal has a turn-on delay of 100 ms after the output voltages have reached their respective minimum sense levels.

Output Voltage	Under-Voltage Nominal Sense Level	Over-Voltage Nominal Sense Level
+5 Vdc	+4.0 Vdc	+5.9 Vdc
-5 Vdc	-4.0 Vdc	-5.9 Vdc
+12 Vdc	+9.6 Vdc	+14.2 Vdc
-12 Vdc	-9.6 Vdc	-14.2 Vdc

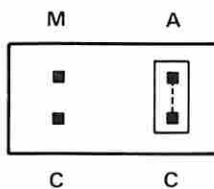
Cassette Interface

The cassette interface is controlled through software. An output from the 8253 timer controls the data to the cassette recorder through pin 5 of the cassette DIN connector at the rear of the system board. The cassette input data is read by an input port bit of the 8255A-5 programmable peripheral interface (8255A-5 PPI). This data is received through pin 4 of the cassette connector. Software algorithms are used to generate and read cassette data. The cassette drive motor is controlled through pins 1 and 3 of the cassette connector. The drive motor on/off switching is controlled by an 8255A-5 PPI output-port bit (hex 61, bit 3). The 8255A-5 address and bit assignments are defined in "I/O Address Map" earlier in this section.

A 2 x 2 Berg pin and a jumper are used on the cassette 'data out' line. The jumper allows use of the 'data out' line as a 0.075-Vdc microphone input when placed across the M and C pins of the Berg connector. A 0.68-Vdc auxiliary input to the cassette recorder is available when the jumper is placed across the A and C pins of the Berg connector. The "System Board Component Diagram" shows the locations of the cassette Berg pins.



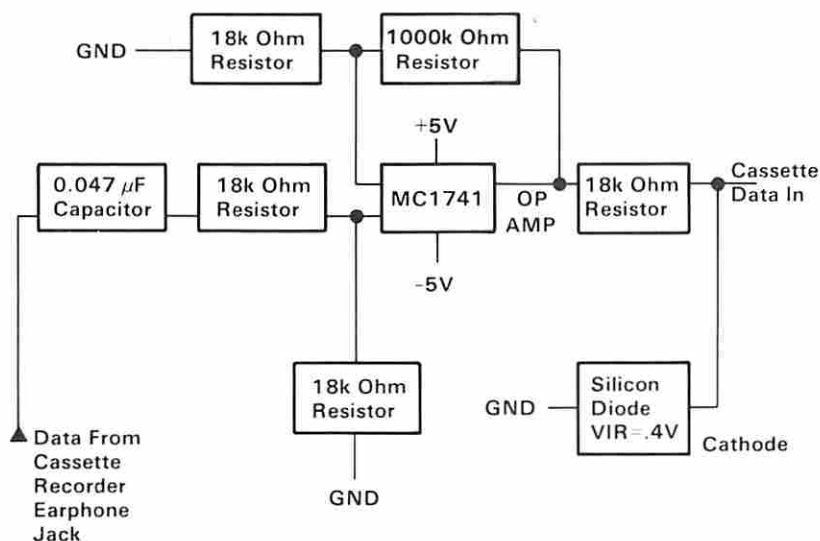
Microphone Input
(0.075 Vdc)



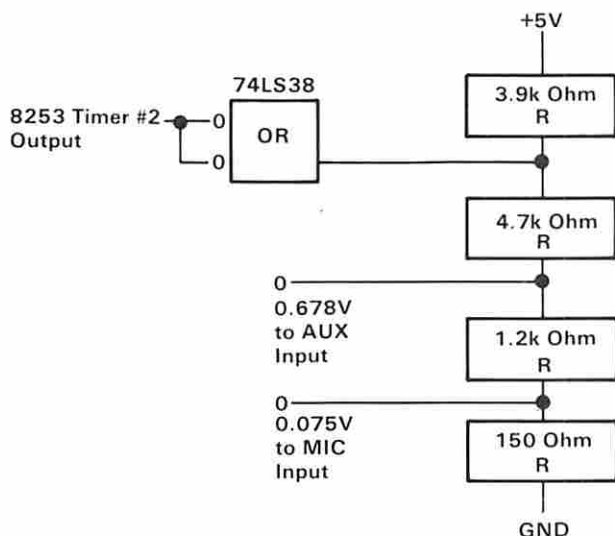
Auxiliary Input
(0.68 Vdc)

Cassette Circuit Block Diagrams

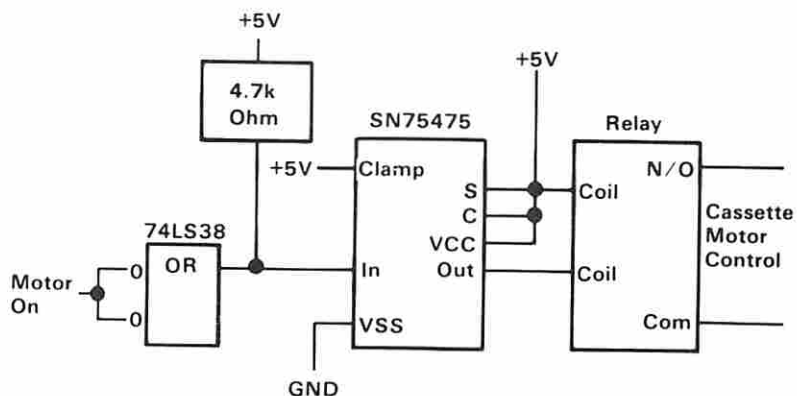
Circuit block diagrams for the cassette-interface read hardware, write hardware, and motor control are illustrated below.



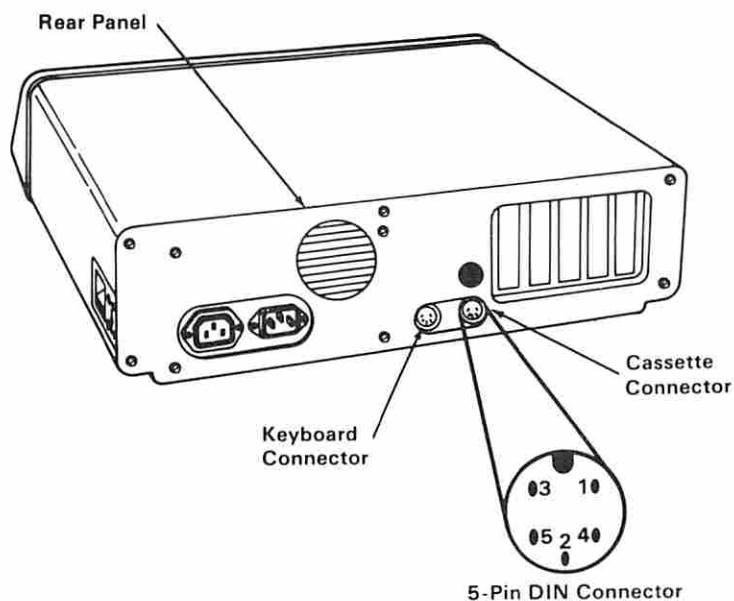
Cassette Interface Read Hardware Block Diagram



Cassette Interface Write Hardware Block Diagram



Cassette Motor Control Block Diagram



Pin	Signal	Electrical Characteristics
1	Motor Control	Common from Relay
2	Ground	
3	Motor Control	Relay N.O. (6 Vdc; 1A)
4	Data In	500nA at $\pm 13V$ - at 1,000 - 2,000 Baud
5	Data Out (Microphone or Auxiliary)	250 μA at 0.68 Vdc or ** 0.075 Vdc

*All voltages and currents are maximum ratings and should not be exceeded.

**Data out can be chosen using a jumper located on the system board.
(Auxiliary $\rightarrow$ 0.68 Vdc or Microphone $\rightarrow$ 0.075 Vdc).

Interchange of these voltages on the cassette recorder could lead to damage of recorder inputs.

Cassette Interface Connector Specifications

IBM Personal Computer Math Coprocessor

The IBM Personal Computer Math Coprocessor enables the IBM Personal Computer to perform high speed arithmetic, logarithmic functions, and trigonometric operations with extreme accuracy.

The coprocessor works in parallel with the processor. The parallel operation decreases operation time by allowing the coprocessor to do mathematical calculations while the processor continues to do other functions.

The first five bits of every instruction opcode for the coprocessor are identical (11011 binary). When the processor and the coprocessor see this instruction opcode, the processor calculates the address, of any variables in memory, while the coprocessor checks the instruction. The coprocessor will then take the memory address from the processor if necessary. To access locations in memory, the coprocessor takes the local bus from the processor when the processor finishes its current instruction. When the coprocessor is finished with the memory transfer, it returns the local bus to the processor.

The IBM Math Coprocessor works with seven numeric data types divided into the three classes listed below.

- Binary integers (3 types)
- Decimal integers (1 type)
- Real numbers (3 types)

Programming Interface

The coprocessor extends the data types, registers, and instructions to the processor.

The coprocessor has eight 80-bit registers which provide the equivalent capacity of 40 16-bit registers found in the processor. This register space allows constants and temporary results to be held in registers during calculations, thus reducing memory access and improving speed as well as bus availability. The register space can be used as a stack or as a fixed register set. When used as a stack, only the top two stack elements are operated on: when used as a fixed register set, all registers are operated on. The Figure below shows representations of large and small numbers in each data type.

Data Type	Bits	Significant Digits (Decimal)	Approximate Range (decimal)
Word Integer	16	4	$-32,768 \leq X \leq +32,767$
Short Integer	32	9	$-2 \times 10^9 \leq X \leq +2 \times 10^9$
Long Integer	64	18	$-9 \times 10^{18} \leq X \leq +9 \times 10^{18}$
Packed Decimal	80	18	$-99...99 \leq X \leq +99...99$ (18 digits)
Short Real*	32	6-7	$8.43 \times 10^{-37} \leq X \leq 3.37 \times 10^{38}$
Long Real*	64	15-16	$4.19 \times 10^{-307} \leq X \leq 1.67 \times 10^{308}$
Temporary Real	80	19	$3.4 \times 10^{-4932} \leq X \leq 1.2 \times 10^{4932}$

*The short and long real data types correspond to the single and double precision data types

Data Types

Hardware Interface

The coprocessor utilizes the same clock generator and system bus interface components as the processor. The coprocessor is wired directly into the processor, as shown in the coprocessor interconnection diagram. The processor's queue status lines (QS0 and QS1) enable the coprocessor to obtain and decode instructions simultaneously with the processor. The coprocessor's busy signal informs the processor that it is executing; the processor's WAIT instruction forces the processor to wait until the coprocessor is finished executing (WAIT for NOT BUSY).

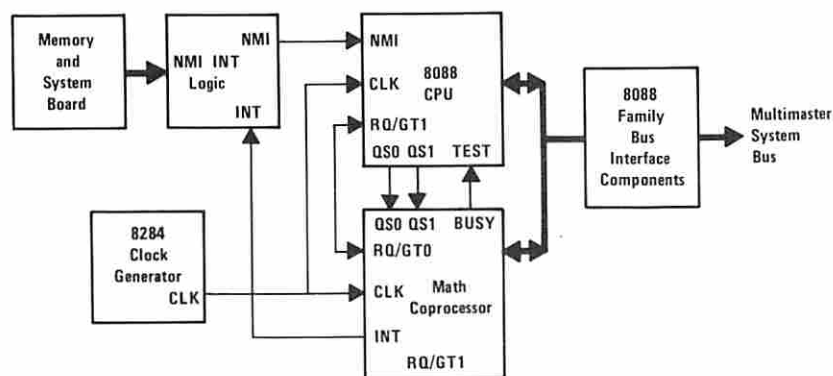
When an incorrect instruction is sent to the coprocessor (for example; divide by zero or load a full register), the coprocessor can signal the processor with an interrupt. There are three conditions that will disable the coprocessor interrupt to the processor:

1. Exception and Interrupt Enable bits of the control word are set to 1's.
2. System board switch block 1 switch 2 set in the On position.
3. NMI Mask REG is set to zero.

At power-on time the NMI Mask REG is cleared to disable the NMI. Any software using the coprocessor's interrupt capability must ensure that conditions 2 and 3 are never met during the operation of the software or an "Endless Wait" will occur. An "Endless Wait" will have the processor waiting for the "Not Busy" signal from the coprocessor while the coprocessor is waiting for the processor to interrupt.

Because a memory parity error may also cause an interrupt to the 8088 NMI line, the program should check that a parity error did not occur (by reading the 8255 port), then clear exceptions by executing the FNSAVE or the FNCLEX instruction. In most cases, the status word would be looked at, and the exception would be identified and acted upon.

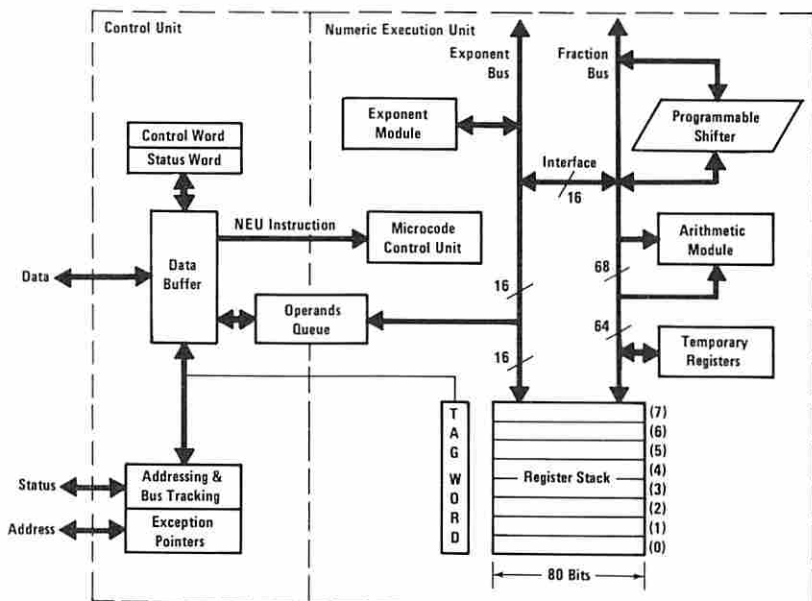
The NMI Mask REG and the coprocessors interrupt are tied to the NMI line through the NMI interrupt logic. Minor conversions of software designed for use with an 8087 must be made before existing software will be compatible with the IBM Personal Computer Math Coprocessor.



Coprocessor Interconnection

Control Unit

The control unit (CU) of the coprocessor and the processor fetch all instructions at the same time, as well as every byte of the instruction stream at the same time. The simultaneous fetching allows the coprocessor to know what the processor is doing at all times. This is necessary to keep a coprocessor instruction from going unnoticed. Coprocessor instructions are mixed with processor instructions in a single data stream. To aid the coprocessor in tracking the processor, nine status lines are interconnected (QS0, QS1, and S0 through S6).



Coprocesor Block Diagram

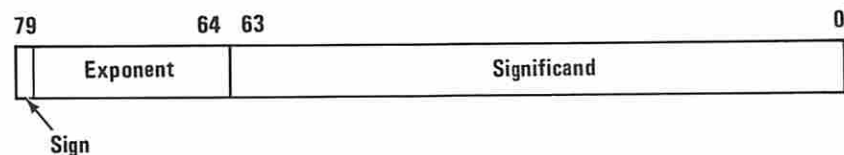
Register Stack

Each of the eight registers in the coprocessor's register stack is 80 bits wide, and each is divided into the "fields" shown in the figure below. The format in the figure below corresponds to the coprocessor's temporary real data type that is used for all calculations.

The ST field in the status word identifies the current top-of-stack register. A load ("push") operation decreases ST by 1 and loads a new value into the top register. A store operation stores the value from the current top register and then increases ST by 1. Thus, the coprocessor's register stack grows "down" toward lower-addressed registers.

Instructions may address registers either implicitly or explicitly. Instructions that operate at the top of the stack, implicitly address the register pointed to by ST. The instruction, FSQRT, replaces the number at the top with its square root; this instruction takes no operands, because the top-of-stack register is implied as the operand. Other instructions specify the register that is to be used. Explicit register addressing is "top-relative." The expression, ST, denotes the current stack top, and ST(i) refers to the *i*th register from the ST in the stack. If ST contains "binary 011" (register 3 is the top of the stack), the instruction, FADD ST,ST(2), would add registers 3 and 5.

Passing subroutine parameters to the register stack eliminates the need for the subroutine to know which registers actually contain the parameters. This allows different routines to call the same subroutine without having to observe a convention for passing parameters in dedicated registers. As long as the stack is not full, each routine simply loads the parameters to the stack and calls the subroutine.



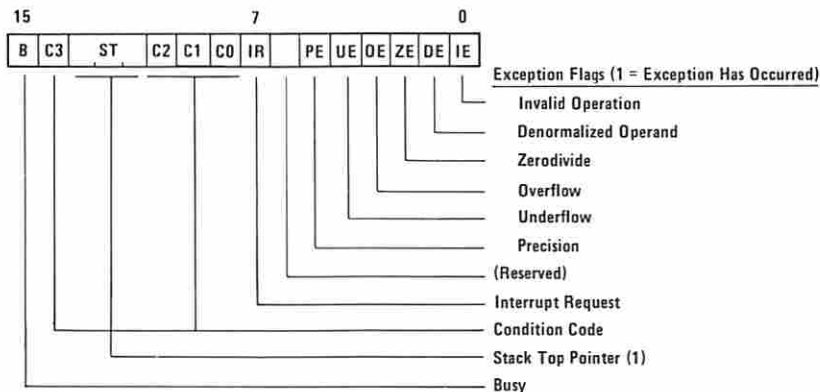
Register Structure

Status Word

The status word reflects the overall condition of the coprocessor. It may be stored in memory with a coprocessor instruction then inspected with a processor code. The status word is divided into the fields shown in the figure below. Bit 15 (BUSY) indicates when the coprocessor is executing an instruction (B=1) or when it is idle (B=0).

Several instructions (for example, the comparison instructions) post their results to the condition code (bits 14 and 10 through 8 of the status word). The main use of the condition code is for conditional branching. This may be accomplished by first executing an instruction that sets the condition code, then storing the status word in memory, and then examining the condition code with processor instructions.

Bits 13 through 11 of the status word point to the coprocessor register that is the current stack top (ST). Bit 7 is the interrupt request field, and bits 5 through 0 are set to indicate that the numeric execution unit has detected an exception while executing the instruction.

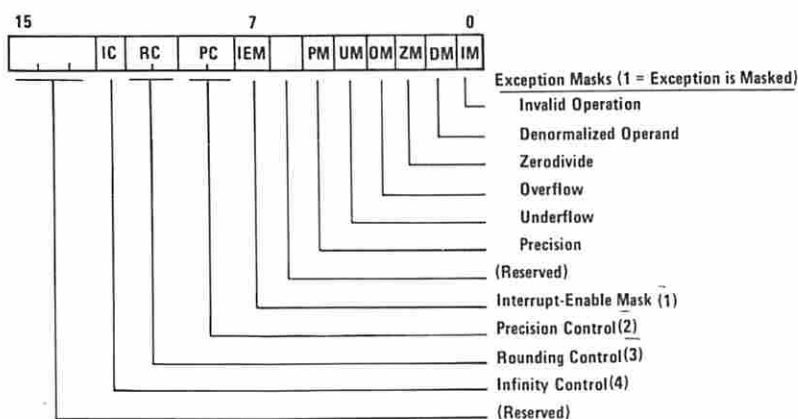


- (1) ST values:
- 000 = register 0 is stack top
 - 001 = register 1 is stack top
 - ...
 - 111 = register 7 is stack top

Status Word Format

Control Word

The coprocessor provides several options that, are selected by loading a control word register.

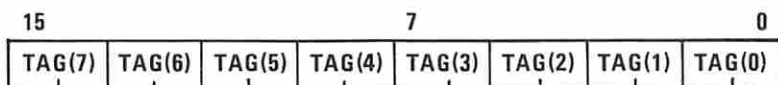


- (1) Interrupt-Enable Mask:
0 = Interrupts Enabled
1 = Interrupts Disabled (Masked)
- (2) Precision Control:
00 = 24 bits
01 = (reserved)
10 = 53 bits
11 = 64 bits
- (3) Rounding Control:
00 = Round to Nearest or Even
01 = Round Down (toward $-\infty$)
10 = Round Up (toward $+\infty$)
11 = Chop (Truncate Toward Zero)
- (4) Infinity Control:
0 = Projective
1 = Affine

Control Word Format

Tag Word

The tag word marks the content of each register, as shown in the Figure below. The main function of the tag word is to optimize the coprocessor's performance under certain circumstances, and programmers ordinarily need not be concerned with it.



Tag values:

00 = Valid (Normal or Unnormal)

01 = Zero (True)

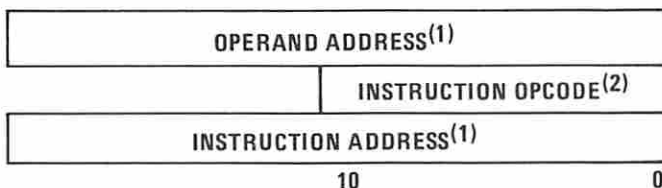
10 = Special (Not-A-Number, ∞ , or Denormal)

11 = Empty

Tag Word Format

Exception Pointers

The exception pointers in the figure below are provided for user-written exception handlers. When the coprocessor executes an instruction, the control unit saves the instruction address and the instruction opcode in the exception pointer registers. An exception handler subroutine can store these pointers in memory and determine which instruction caused the exception.



⁽¹⁾ 20-bit physical address

⁽²⁾ 11 least significant bits of opcode: 5 most significant bits are always
COPROCESSOR HOOK (11011B)

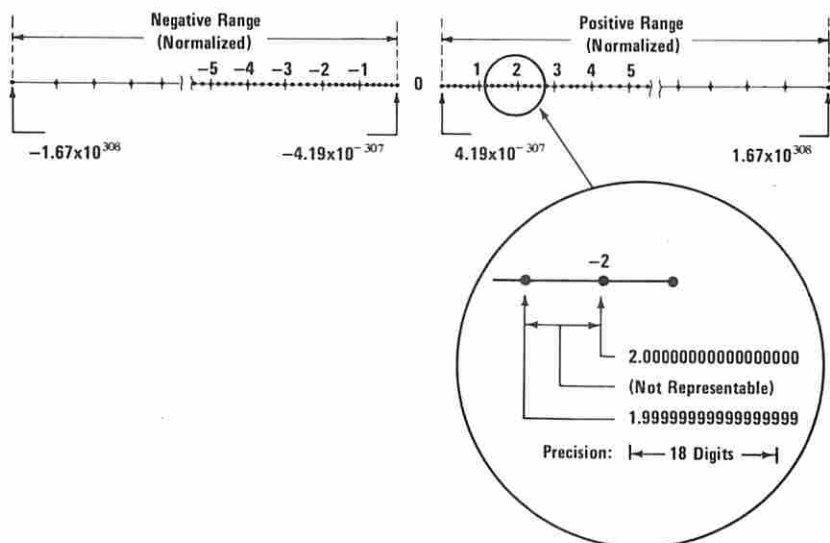
Exception Pointers Format

Number System

The figure below shows the basic coprocessor real number system on a real number line (decimal numbers are shown for clarity, although the coprocessor actually represents numbers in binary). The dots indicate the subset of real numbers the coprocessor can represent as data and final results of calculations. The coprocessor's range is approximately $\pm 4.19 \times 10^{-307}$ to $\pm 1.67 \times 10^{308}$.

The coprocessor can represent a great many of, but not all, the real numbers in its range. There is always a "gap" between two adjacent coprocessor numbers, and the result of a calculation may fall within this space. When this occurs, the coprocessor rounds the true result to a number it can represent.

The coprocessor actually uses a number system that is a superset of that shown in the figure below. The internal format (called temporary real) extends the coprocessor's range to about $\pm 3.4 \times 10^{-4932}$ to $\pm 1.2 \times 10^{4932}$, and its precision to about 19 (equivalent decimal) digits. This format is designed to provide extra range and precision for constants and intermediate results, and is not normally intended for data or final results.



Coprocessor Number System

Instruction Set

On the following pages are descriptions of the operation for the coprocessor's 69 instructions.

An instruction has two basic types of operands – sources and destinations. A source operand simply supplies one of the “inputs” to an instruction; it is not altered by the instruction. A destination operand may also provide an input to an instruction. It is distinguished from a source operand, however, because its content can be altered when it receives the result produced by that operation; that is the destination is replaced by the result.

The operands of any instructions can be coded in more than one way. For example, FADD (add real) may be written without operands, with only a source, or with a destination and a source operand. The instruction descriptions use the simple convention of separating alternative operand forms with slashes; the slashes, however, are not coded. Consecutive slashes indicate there are no explicit operands. The operands for FADD are thus described as:

// source/destination, source

This means that FADD may be written in any of three ways:

FADD

FADD source

FADD destination,source

It is important to bear in mind that memory operands may be coded with any of the processor's memory addressing modes.

FABS

FABS (absolute value) changes the top stack element to its absolute value by making its sign positive.

FABS (no operands)			Exceptions: 1		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	14	10-17	0	2	FABS

FADD

Addition

FADD // source/destination,source

FADDP destination,source

FIADD source

The addition instructions (add real, add real and pop, integer add) add the source and destination operands and return the sum to the destination. The operand at the stack top may be doubled by coding FADD ST,ST(0).

FADD			Exceptions: I, D, O, U, P		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
//ST,ST(i)/ST(i),ST	85	70-100	0	2	FADD ST,ST(4)
short-real	105+EA	90-120+EA	4	2-4	FADD AIR_TEMP [SI]
long-real	110+EA	95-125+EA	8	2-4	FADD [BX],MEAN

FADDP			Exceptions: I, D, O, U, P		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
ST(I),ST	90	75-105	0	2	FADD ST(2), ST

FIADD		Exceptions: I, D, O, P			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	120+EA	102-137+EA	2	2-4	FIADD DISTANCE_TRAVELLED
short-integer	125+EA	108-143+EA	4	2-4	FIADD PULSE_COUNT[SI]

FBLD

FBLD Source

FBLD (packed decimal BCD) load)) converts the content of the source operand from packed decimal to temporary real and loads (pushes) the result onto the stack. The packed decimal digits of the source are assumed to be in the range X '0-9H'.

FBLD		Exceptions: I			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
packed-decimal	300+EA	290-310+EA	10	2-4	FBLD YTD_SALES

FBSTP

FBSTP destination

FBSTP (packed decimal (BCD) store and pop) performs the inverse of FBLD, where the stack top is stored to the destination in the packed-decimal data type.

FBSTP		Exceptions: I			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
packed-decimal	530+EA	520-542+EA	12	2-4	FBSTP [BX].FORECAST

FCHS

FCHS (change sign) complements (reverses) the sign of the top stack element.

FCHS (no operands)			Exceptions: I		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	15	10-17	0	2	FCHS

FCLEX/FNCLEX

FCLEX/FNCLEX (clear exceptions) clears all exception flags, the interrupt request flag, and the busy flag in the status word.

FCLEX/FNCLEX (no operands)			Exceptions: None		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	5	2-8	0	2	FNCLEX

FCOM

FCOM/ /source

FCOM (compare real) compares the stack top to the source operand. This results in the setting of the condition code bits.

FCOM			Exceptions: I, D		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
//ST(i)	45	40-50	0	2	FCOM ST(1)
short-real	65+EA	60-70+EA	4	2-4	FCOM [BP.] UPPER_LIMIT
long-real	70+EA	65-75+EA	8	2-4	FCOM WAVELENGTH

C3	C0	Order
0	0	ST > source
0	1	ST < source
1	0	ST = source
1	1	ST ? source

NANS and ∞ (projective) cannot be compared and return C3=C0=1 as shown above.

FCOMP

FCOMP/ /source

FCOMP (compare real and pop) operates like FCOM, and in addition pops the stack.

FCOMP Exceptions: I, D					
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
//ST(i)	47	42-52	0	2	FCOMP ST(2)
short-real	68+EA	63-73+EA	4	2-4	FCOMP [BP].N_READINGS
long-real	72+EA	67-77+EA	8	2-4	FCOMP DENSITY

FCOMPP

FCOMPP/ /source

FCOMPP (compare real and pop twice) operates like FCOM and, additionally, pops the stack twice, discarding both operands. The comparison is of the stack top to ST(1); no operands may be explicitly coded.

FCOMPP (no operands) Exceptions: I, D					
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	50	45-55	0	2	FCOMPP

FDECSTP

FDECSTP (decrement stack pointer) subtracts 1 from ST, the stack top pointer in the status word.

FDECSTP (no operands)			Exceptions: None		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	9	6-12	0	2	FDECSTP

FDISI/FNDISI

FDISI/FNDISI (disable interrupts) sets the interrupt enable mask in the control word.

FDISI/FNDISI (no operands)			Exceptions: None		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	5	2-8	0	2	FDISI

FDIV

Normal division

FDIV / /source/ destination,source

FDIVP destination,source

FIDIV source

The normal division instructions (divide real, divide real and pop, integer divide) divide the destination by the source and return the quotient to the destination.

FDIV		Exceptions: I, D, Z, O, U, P			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
//ST(i),ST	198	193-203	0	2	FDIV
short-real	220+EA	215-225+EA	4	2-4	FDIV DISTANCE
long-real	225+EA	220-230+EA	8	2-4	FDIV ARC[DI]

FDIVP		Exceptions: I, D, Z, O, U, P			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
ST(i),ST	202	197-207	0	2	FDIVP ST(4), ST

FIDIV		Exceptions: I, D, Z, O, U, P			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	230+EA	224-238+EA	2	2-4	FIDIV SURVEY.OBSERVATIONS
short-integer	236+EA	230-243+EA	4	2-4	FIDIV RELATIVE_ANGLE[DI]

FDIVR

Reversed Division

FDIVR //source/ destination,source

FDIVRP destination,source

FIDIVR source

The reversed division instructions (divide real reversed, divide real reversed and pop, integer divide reversed) divide the source operand by the destination and return the quotient to the destination.

FDIVR					Exceptions: I, D, Z, O, U, P
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
//ST,ST(i)/ST(i),ST	199	194-204	0	2	FDIVR ST(2),ST
short-real	221+EA	216-226+EA	6	2-4	FDIVR [BX].PULSE_RATE
long-real	226+EA	221-231+EA	8	2-4	FDIVR RECORDER.FREQUENCY

FDIVRP					Exceptions: I, D, Z, O, U, P
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
ST(i),ST	203	198-208	0	2	FDIVRP ST(1),ST

FIDIVR					Exceptions: I, D, Z, O, U, P
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	230+EA	225-239+EA	2	2-4	FIDIVR [BP].X_COORD
short-integer	237+EA	231-245+EA	4	2-4	FIDIVR FREQUENCY

FENI/FNENI

FENI/FNENI (enable interrupts) clear the interrupt enable mask in the control word.

FENI/FNENI (no operands)			Exceptions: None		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	5	2-8	0	2	FNENI

FFREE

FFREE destination

FFREE (free register) changes the destination register's tag to empty; the content of the register is not affected.

FFREE			Exceptions: None		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
ST(i)	11	9-16	0	2	FFREE ST(1)

FICOM

FICOM source

FICOM (integer compare) compares the source to the stack top.

FICOM			Exceptions: I, D		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	80+EA	72-86+EA	2	2-4	FICOM TOOL.N_PASSES
short-integer	85+EA	78-91+EA	2	2-4	FICOM [BP+41].PARAM_COUNT

FICOMP

FICOMP source

FICOMP (integer compare and pop) operates the same as FICOM and additionally pops the stack.

FICOMP		Exceptions: I, D			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	82+EA	74-88+EA	2	2-4	FICOMP [BP].LIMIT [SI]
short-integer	87+EA	80-93+EA	4	2-4	FICOMP N_SAMPLES

FILD

FILD source

FILD (integer load) loads (pushes) the source onto the stack.

FILD		Exceptions: I			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	50+EA	46-54+EA	2	2-4	FILD [BX].SEQUENCE
short-integer	56+EA	52-60+EA	4	2-4	FILD STANDOFF[DI]
long-integer	64+EA	60-68+EA	8	2-4	FILD RESPONSE.COUNT

FINCSTP

FINCSTP (increment stack pointer) adds 1 to the stack top pointer (ST) in the status word.

FINCSTP (no operands)		Exceptions: None			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	9	6-12	0	2	FINCSTP

FINIT/FNINIT

FINIT/FNINIT (initialize processor) performs the functional equivalent of a hardware RESET.

FINIT/FNINIT (no operands)			Exceptions: None		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	5	2-8	0	2	FINIT

Field	Value	Interpretation
Control Word		
Infinity Control	0	Projective
Rounding Control	00	Round to nearest
Precision Control	11	64 bits
Interrupt-enable Mask	1	Interrupts disabled
Exception Masks	111111	All exceptions masked
Status Word		
Busy	0	Not Busy
Condition Code	????	(Indeterminate)
Stack Top	000	Empty stack
Interrupt Request	0	No interrupt
Exception Flags	000000	No exceptions
Tag Word		
Tags	11	Empty
Registers	N.C.	Not changed
Exception Pointers		
Instruction Code	N.C.	Not changed
Instruction Address	N.C.	Not changed
Operand Address	N.C.	Not changed

FIST

FIST destination

FIST (integer store) stores the stack top to the destination in the integer format.

FIST		Exceptions: I, P			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	86+EA	80-90+EA	4	2-4	FIST OBS.COUNT[SI]
short-integer	88+EA	82-92+EA	6	2-4	FIST [BP].FACTORED_PULSES

FISTP

FISTP destination

FISTP (integer store and pop) operates like FIST and also pops the stack following the transfer. The destination may be any of the binary integer data types.

FISTP		Exceptions: I, P			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	88+EA	82-92+EA	4	2-4	FISTP [BX].ALPHA_COUNT[SI]
short-integer	90+EA	84-94+EA	6	2-4	FISTP CORRECTED_TIME
long-integer	100+EA	94-105+EA	10	2-4	FISTP PANEL.N_READINGS

FLD

FLD source

FLD (load real) loads (pushes) the source operand onto the top of the register stack.

FLD Exceptions: I, D					
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
ST(i)	20	17-22	0	2	FLD ST(0)
short-real	43+EA	38-56+EA	4	2-4	FLD READING(SI).PRESSURE
long-real	46+EA	40-60+EA	8	2-4	FLD [BP].TEMPERATURE
temp-real	57+EA	53-65+EA	10	2-4	FLD SAVEREADING

FLDCW

FLDCW source

FLDCW (load control word) replaces the current processor control word with the word defined by the source operand.

FLDCW Exceptions: None					
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
2-bytes	10+EA	7-14+EA	2	2-4	FLDCW CONTROL_WORD

FLDENV

FLDENV source

FLDENV (load environment) reloads the coprocessor environment from the memory area defined by the source operand.

FLDENV		Exceptions: None			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
14-bytes	40+EA	35-45+EA	14	2-4	FLDENV [BP+6]

FLDLG2

FLDLG2 (load log base 10 of 2) loads (pushes) the value of $\text{LOG}_{10}2$ onto the stack.

FLDLG2 (no operands)		Exceptions: 1			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	21	18-24	0	2	FLDLG2

FLDLN2

FLDLN2 (load log base e of 2) loads (pushes) the value of LOG_e2 onto the stack.

FLDLN2 (no operands)		Exceptions: 1			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	20	17-23	0	2	FLDLN2

FLDL2E

FLDL2E (load log base 2 of e) loads (pushes) the value $\text{LOG}_2 e$ onto the stack.

FLDL2E (no operands)			Exceptions: 1		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	18	15-21	0	2	FLDL2E

FLDL2T

FLDL2T (load log base 2 of 10) loads (pushes) the value of $\text{LOG}_2 10$ onto the stack.

FLDL2T (no operands)			Exceptions: 1		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	19	16-22	0	2	FLDL2T

FLDPI

FLDPI (load π) loads (pushes) π onto the stack.

FLDPI (no operands)			Exceptions: 1		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	19	16-22	0	2	FLDPI

FLDZ

FLDZ (load zero) loads (pushes) +0.0 onto the stack.

FLDZ (no operands)			Exceptions: 1		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	14	11-17	0	2	FLDZ

FLD1

FLD1 (load one) loads (pushes) +1.0 onto the stack.

FLD1 (no operands)			Exceptions: 1		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	18	15-21	0	2	FLD1

FMUL

Multiplication

FMUL / /source/destination,source

FMULP destination,source

FIMUL source

The multiplication instructions (multiply real, multiply real and pop, integer multiply) multiply the source and destination operands and return the product to the destination. Coding FMUL ST,ST(0) square the content of the stack top.

FMUL		Exceptions: I, D, O, U, P			
Operands	Execution Clocks		Trans- fers 8088	Bytes	Coding Example
	Typical	Range			
//ST(i),ST/ST,ST(i) ¹	97	90-105	0	2	FMUL ST,ST(3)
//ST(i),ST/ST,ST(i)	138	130-145	0	2	FMUL ST,ST(3)
short-real	118+EA	110-125+EA	4	2-4	FMUL SPEED_FACTOR
long-real ¹	120+EA	112-126+EA	8	2-4	FMUL [BP].HEIGHT
long-real	161+EA	154-168+EA	8	2-4	FMUL [BP].HEIGHT

¹ occurs when one or both operands is "short" - it has 40 trailing zeros in its fraction.

FMULP		Exceptions: I, D, O, U, P			
Operands	Execution Clocks		Trans- fers 8088	Bytes	Coding Example
	Typical	Range			
ST(i),ST ¹	100	94-108	0	2	FMULP ST(1),ST
ST(i),ST	142	134-148	0	2	FMULP ST(1),ST

¹ occurs when one or both operands is "short" - it has 40 trailing zeros in its fraction.

FIMUL		Exceptions: I, D, O, P			
Operands	Execution Clocks		Trans- fers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	130+EA	124-138+EA	2	2-4	FIMUL BEARING
short-integer	136+EA	130-144+EA	4	2-4	FIMUL POSITION.Z_AXIS

FNOP

FNOP (no operation) stores the stack top to the stack top (FST ST,ST(0)) and thus effectively performs no operation.

FNOP (no operands)			Exceptions: None		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	13	10-16	0	2	FNOP

FPATAN

FPATAN (partial arctangent) computes the function $\theta = \text{ARCTAN}(Y/X)$. X is taken from the top stack element and Y from ST(1). Y and X must observe the inequality $0 < Y < X < \infty$. The instruction pops the stack and returns θ to the (new) stack top, overwriting the Y operand.

FPATAN (no operands)			Exceptions: U, P (operands not checked)		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	650	250-800	0	2	FPATAN

FPREM

FPREM (partial remainder) performs modulo division on the top stack element by the next stack element, that is, ST(1) is the modulus.

FPREM (no operands)			Exceptions: I, D, U		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	125	15-190	0	2	FPREM

FPTAN

FPTAN (partial tangent) computes the function $Y/X = \tan(\theta)$. θ is taken from the top stack element; it must lie in the range $0 < \theta < \pi/4$. The result of the operation is a ratio; Y replaces θ in the stack and X is pushed, becoming the new stack top.

FPTAN		Exceptions: I, P (operands not checked)			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	450	30-540	0	2	FPTAN

FRNDINT

FRNDINT (round to integer) rounds the top stack element to an integer.

FRNDINT (no operands)		Exceptions: I, P			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	45	16-50	0	2	FRNDINT

FRSTOR

FRSTOR source

FRSTOR (restore state) reloads the coprocessor from the 94-byte memory area defined by the source operand.

FRSTOR		Exceptions: None			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
94-bytes	210+EA	205-215+EA	96	2-4	FRSTOR [BP]

FSAVE/FNSAVE

FSAVE/FNSAVE destination

FSAVE/FNSAVE (save state) writes the full coprocessor state – environment plus register stack – to the memory location defined by the destination operand.

FSAVE/FNSAVE			Exceptions: None		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
94-bytes	210+EA	205-215+EA	94	2-4	FSAVE [BP]

FSCALE

FSCALE (scale) interprets the value contained in ST(1) as an integer, and adds this value to the exponent of the number in ST. This is equivalent to:

$$ST \leftarrow ST \cdot 2^{ST(1)}$$

Thus, FSCALE provides rapid multiplication or division by integral powers of 2.

FSCALE (no operands)			Exceptions: I, O, U		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	35	32-38	0	2	FSCALE

FSQRT

FSQRT (square root) replaces the content of the top stack element with its square root.

Note: the square root of -0 is defined to be -0 .

FSQRT (no operands)			Exceptions: I, D, P		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	183	180-186	0	2	FSQRT

FST

FST destination

FST (store real) transfers the stack top to the destination, which may be another register on the stack or long real memory operand.

FST			Exceptions: I, O, U, P		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
ST(i)	18	15-22	0	2	FST ST(3)
short-real	87+EA	84-90+EA	6	2-4	FST CORRELATION [DI]
long-real	100+EA	96-104+EA	10	2-4	FST MEAN_READING

FSTCW/FNSTCW

FSTCW/FNSTCW destination

FSTCW/FNSTCW (store control word) writes the current processor control word to the memory location defined by the destination.

FSTCW/FNSTCW			Exceptions: None		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
2-bytes	15+EA	12-18+EA	4	2-4	FSTCW SAVE_CONTROL

FSTENV/FNSTENV

FSTENV/FNSTENV destination

FSTENV/FNSTENV (store environment) writes the coprocessor's basic status – control, status and tag words, and exception pointers – to the memory location defined by the destination operand.

FSTENV/FNSTENV			Exceptions: None		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
14-bytes	45+EA	40-50+EA	16	2-4	FSTENV [BP]

FSTP

FSTP destination

FSTP (store real and pop) operates the same as FST, except that the stack is popped following the transfer.

FSTP		Exceptions: I, O, U, P			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
ST(i)	20	17-24	0	2	FSTP ST(2)
short-real	89+EA	86-92+EA	6	2-4	FSTP [BX].ADJUSTED_RPM
long-real	102+EA	98-106+EA	10	2-4	FSTP TOTAL_DOSAGE
temp-real	55+EA	52-58+EA	12	2-4	FSTP REG_SAVE[SI]

FSTSW/FNSTSW

FSTSW/FNSTSW destination

FSTSW/FNSTSW (store status word) writes the current value of the coprocessor status word to the destination operand in memory.

FSTSW/FNSTSW		Exceptions: None			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
2-bytes	14+EA	12-18+EA	4	2-4	FSTSW SAVE_STATUS

FSUB

Subtraction

FSUB / /source/destination,source

FSUBP destination,source

FISUB source

The normal subtraction instructions (subtract real, subtract real and pop, integer subtract) subtract the source operand from the destination and return the difference to the destination.

FSUB		Exceptions: I, D, O, U, P			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
//ST,ST(i)/ST(i),ST	85	70-100	0	2	FSUB ST,ST(2)
short-real	105+EA	90-120+EA	4	2-4	FSUB BASE_VALUE
long-real	110+EA	95-125+EA	8	2-4	FSUB COORDINATE.X

FSUBP		Exceptions: I, D, O, U, P			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
ST(i),ST	90	75-105	0	2	FSUBP ST(2),ST

FISUB		Exceptions: I, D, O, P			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	120+EA	102-137+EA	2	2-4	FISUB BASE_FREQUENCY
short-integer	125+EA	108-143+EA	4	2-4	FISUB TRAIN_SIZE[DI]

FSUBR

Reversed Subtraction

FSUBR / /source/destination,source

FSUBRP destination,source

FISUBR source

The reversed subtraction instructions (subtract real reversed, subtract real reversed and pop, integer subtract reversed) subtract the destination from the source and return the difference to the destination.

FSUBR		Exceptions: I, D, O, U, P			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
//ST,ST(i)/ST(i),ST	87	70-100	0	2	FSUBR ST,ST(1)
short-real	105+EA	90-120+EA	4	2-4	FSUBR VECTOR[SI]
long-real	110+EA	95-125+EA	8	2-4	FSUBR [BX].INDEX

FSUBRP		Exceptions: I, D, O, U, P			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
ST(i),ST	90	75-105	0	2	FSUBRP ST(1),ST

FISUBR		Exceptions: I, D, O, P			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
word-integer	120+EA	103-139+EA	2	2-4	FISUBR FLOOR[BX][SI]
short-integer	125+EA	109-144+EA	4	2-4	FISUBR BALANCE

FTST

FTST (test) tests the top stack element by comparing it to zero. The result is posted to the condition codes.

FTST (no operands)			Exceptions: I, D		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	42	38-48	0	2	FTST

C3	C0	Result
0	0	ST is positive and nonzero
0	1	ST is negative and nonzero
1	0	ST is zero (+ or -)
1	1	ST is not comparable (that is, it is a NAN or projective ∞)

FWAIT

FWAIT (processor instruction)

FWAIT is not actually a coprocessor instruction, but an alternate mnemonic for the processor WAIT instruction. The FWAIT mnemonic should be coded whenever the programmer wants to synchronize the processor to the coprocessor, that is, to suspend further instruction decoding until the coprocessor has completed the current instruction.

FWAIT (no operands)			Exceptions: Non (CPU instruction)		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	3+5n	3+5n	0	1	FWAIT

FXAM

FXAM (examine) reports the content of the top stack element as positive/negative and NAN/unnormal/denormal/normal/zero, or empty.

FXAM		Exceptions: None			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	17	12-23	0	2	FXAM

Condition Code				Interpretation
C3	C2	C1	C0	
0	0	0	0	+ Unnormal
0	0	0	1	+ NAN
0	0	1	0	- Unnormal
0	0	1	1	- NAN
0	1	0	0	+ Normal
0	1	0	1	+ ∞
0	1	1	0	- Normal
0	1	1	1	- ∞
1	0	0	0	+ 0
1	0	0	1	Empty
1	0	1	0	- 0
1	0	1	1	Empty
1	1	0	0	+ Denormal
1	1	0	1	Empty
1	1	1	0	- Denormal
1	1	1	1	Empty

FXCH

FXCH/ /destination

FXCH (exchange registers) swaps the contents of the destination and the stack top registers. If the destination is not coded explicitly, ST(1) is used.

FXCH		Exceptions: 1			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
//ST(i)	12	10-15	0	2	FXCH ST(2)

FXTRACT

FXTRACT (extract exponent and significant) “decomposes” the number in the stack top into two numbers that represent the actual value of the operand’s exponent and significand fields contained in the stack top and ST(1).

FXTRACT		Exceptions: 1			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	50	27-55	0	2	FXTRACT

FYL2X

FYL2X (Y log base 2 of X) calculates the function $Z = Y \cdot \text{LOG}_2 X$. X is taken from the stack top and Y from ST(1). The operands must be in the ranges $0 < X < \infty$ and $-\infty < Y < +\infty$. The instruction pops the stack and returns Z at the (new) stack top, replacing the Y operand.

$$\text{LOG}_2 Y \cdot \text{LOG}_2 X$$

FYL2X			Exceptions: P (operands not checked)		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	950	900-1100	0	2	FYL2X

FYL2XP1

FYL2XP1 (Y log base 2 of $(X + 1)$) calculates the function $Z = Y \cdot \text{LOG}_2 (X + 1)$. X is taken from the stack top and must be in the range $0 < |X| < (1 - \sqrt{2}/2)$. Y is taken from ST(1) and must be in the range $-\infty < Y < \infty$. FYL2XP1 pops the stack and returns Z at the (new) stack top, replacing Y.

FYL2XP1			Exceptions: P (operands not checked)		
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	850	700-1000	0	2	FYL2XP1

F2XM1

F2XM1 (2 to the X minus 1) calculates the function $Y=2^X-1$. X is taken from the stack top and must be in the range $0 < X < 0.5$. The result Y replaces the stack top.

This instruction is designed to produce a very accurate result even when X is close to zero. To obtain $Y=2^X$, add 1 to the result delivered by F2XM1.

F2XM1		Exceptions: U, P (operands not checked)			
Operands	Execution Clocks		Transfers 8088	Bytes	Coding Example
	Typical	Range			
(no operands)	500	310-630	0	2	F2XM1

IBM Keyboard

The keyboard has a permanently attached cable that connects to a DIN connector at the rear of the system unit. This shielded four-wire cable has power (+5 Vdc), ground, and two bidirectional signal lines. The cable is approximately 6-feet long and is coiled, like that of a telephone handset.

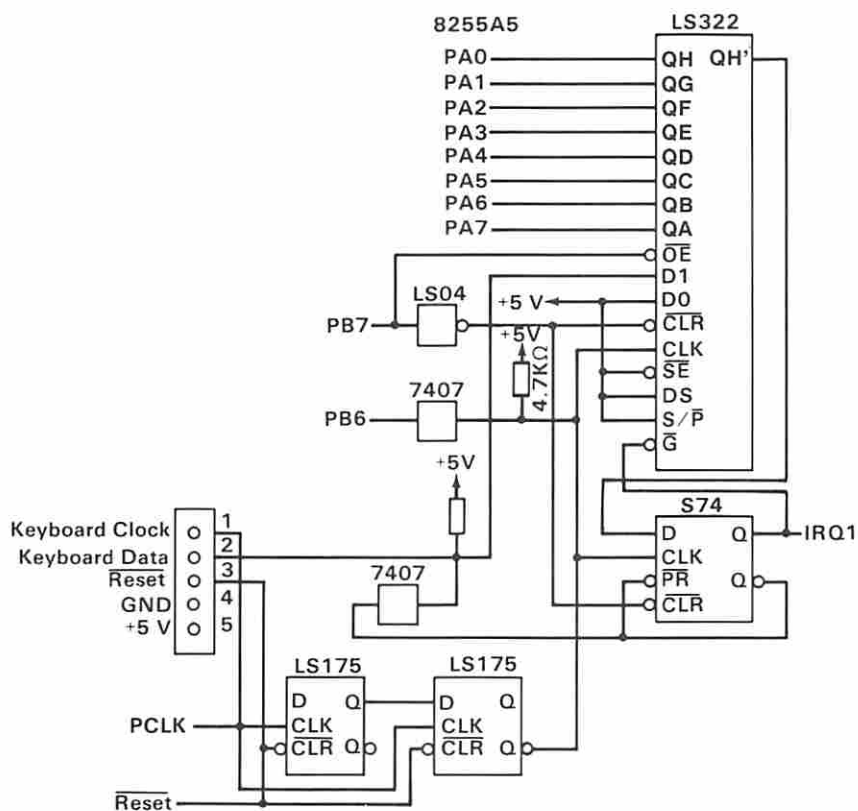
The keyboard uses a capacitive technology with a microcomputer (Intel 8048) performing the keyboard scan function. The keyboard has three tilt positions for operator comfort (5-, 7-, or 15-degree tilt orientations).

The keyboard has 83 keys arranged in three major groupings. The central portion of the keyboard is a standard typewriter keyboard layout. On the left side are 10 function keys. These keys are user-defined by the software. On the right is a 15-key keypad. These keys are also defined by the software, but have legends for the functions of numeric entry, cursor control, calculator pad, and screen edit.

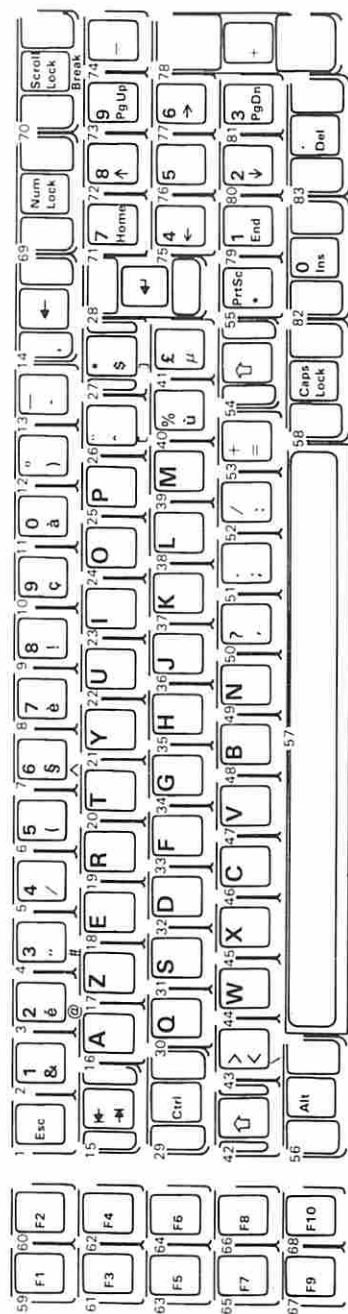
The keyboard interface is defined so that system software has maximum flexibility in defining certain keyboard operations. This is accomplished by having the keyboard return scan codes rather than American Standard Code for Information Interchange (ASCII) codes. In addition, all keys are typematic and generate both a make and a break scan code. For example, key 1 produces scan code hex 01 on make and code hex 81 on break. Break codes are formed by adding hex 80 to make codes. The keyboard I/O driver can define keyboard keys as shift keys or typematic, as required by the application.

The microcomputer (Intel 8048) in the keyboard performs several functions, including a power-on self-test when requested by the system unit. This test checks the microcomputer ROM, tests memory, and checks for stuck keys. Additional functions are: keyboard scanning, buffering of up to 20 key scan codes, maintaining bidirectional serial communications with the system unit, and executing the hand-shake protocol required by each scan-code transfer.

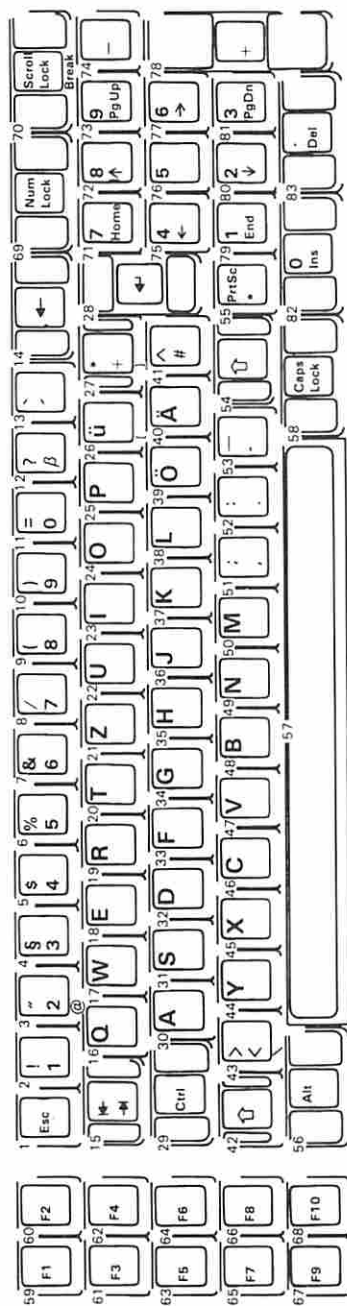
The following pages have figures that show the keyboard, the scan codes, and the keyboard interface connector specifications.



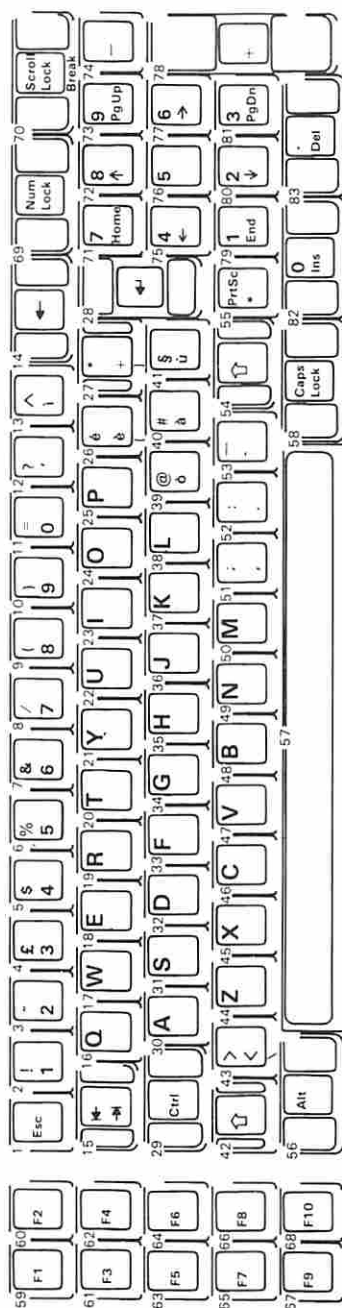
Keyboard Interface Block Diagram



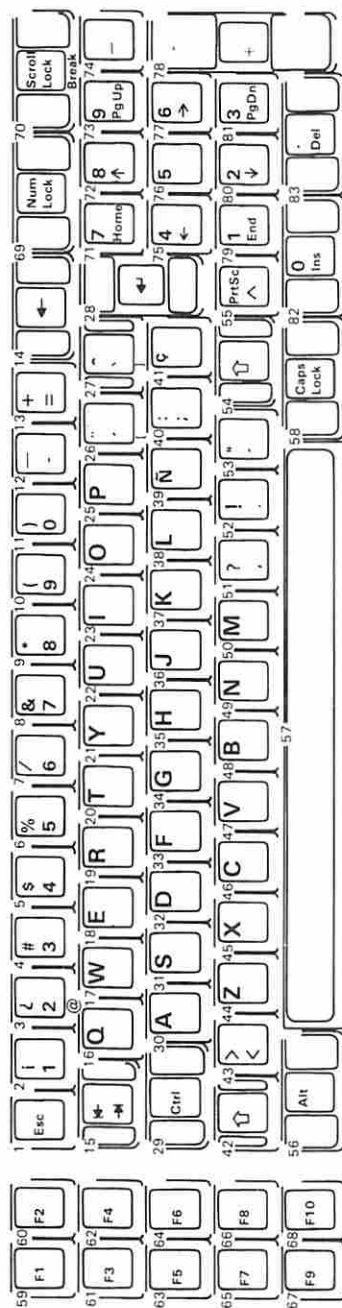
Note: Nomenclature is on both top and front face of keybutton as shown. The number to the upper left designates the button position.



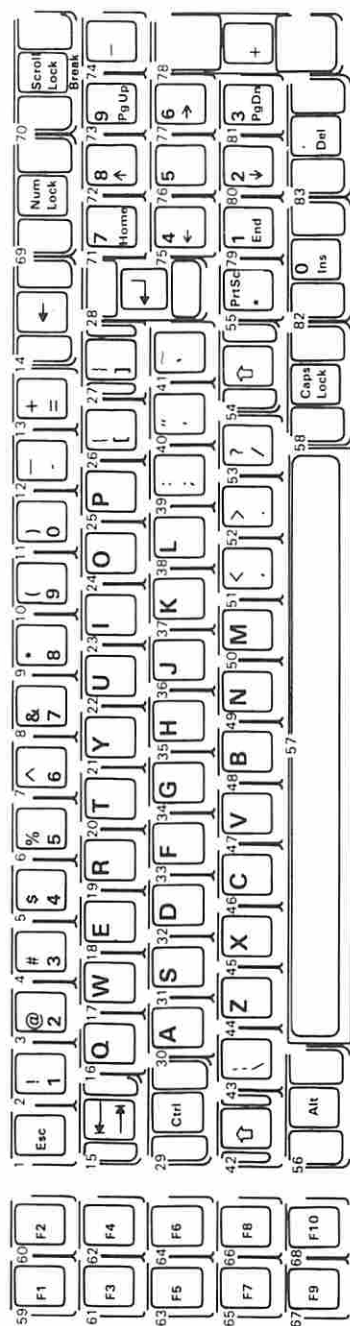
Note: Nomenclature is on both top and front face of keybutton as shown. The number to the upper left designates the button position.



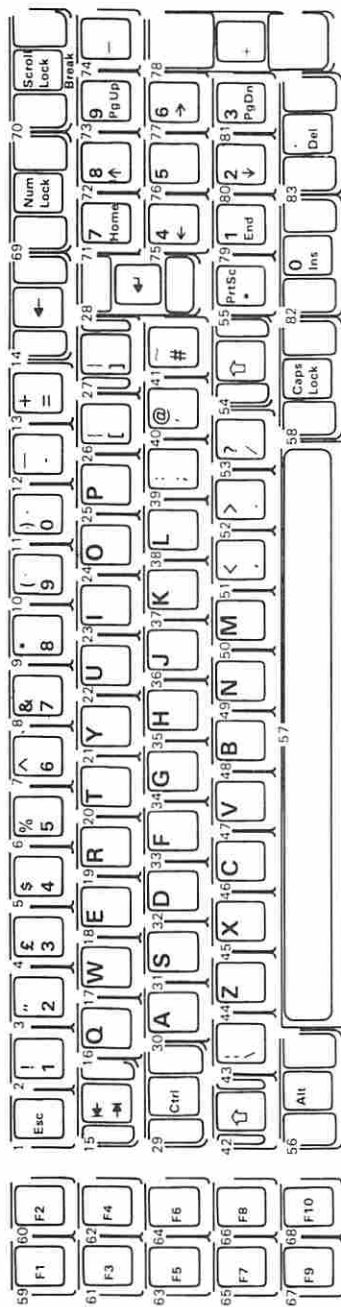
Note: Nomenclature is on both top and front face of keybutton as shown. The number to the upper left designates the button position.



Note: Nomenclature is on both top and front face of keybutton as shown. The number to the upper left designates the button position.



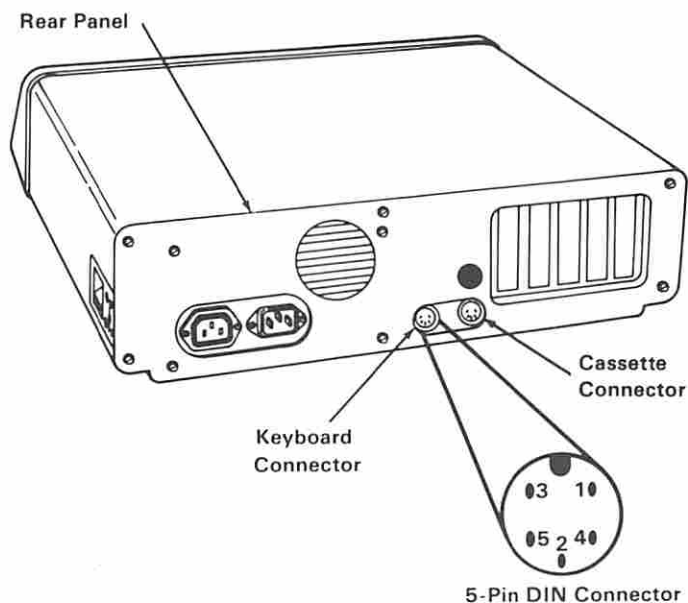
Note: Nomenclature is on both the top and front face of the keybutton as shown. The number to the upper left designates the button position.



Note: Nomenclature is on both top and front face of keybutton as shown. The number to the upper left designates the button position.

Key Position	Scan Code in Hex	Key Position	Scan Code in Hex
1	01	43	2B
2	02	44	2C
3	03	45	2D
4	04	46	2E
5	05	47	2F
6	06	48	30
7	07	49	31
8	08	50	32
9	09	51	33
10	0A	52	34
11	0B	53	35
12	0C	54	36
13	0D	55	37
14	0E	56	38
15	0F	57	39
16	10	58	3A
17	11	59	3B
18	12	60	3C
19	13	61	3D
20	14	62	3E
21	15	63	3F
22	16	64	40
23	17	65	41
24	18	66	42
25	19	67	43
26	1A	68	44
27	1B	69	45
28	1C	70	46
29	1D	71	47
30	1E	72	48
31	1F	73	49
32	20	74	4A
33	21	75	4B
34	22	76	4C
35	23	77	4D
36	24	78	4E
37	25	79	4F
38	26	80	50
39	27	81	51
40	28	82	52
41	29	83	53
42	2A		

Keyboard Scan Codes



Pin	TTL Signal	Signal Level
1	+Keyboard Clock	+5 Vdc
2	+Keyboard Data	+5 Vdc
3	-Keyboard Reset (Not used by keyboard)	
Power Supply Voltages		Voltage
4	Ground	0
5	+5 Volts	+5 Vdc

Keyboard Interface Connector Specifications

Expansion Unit

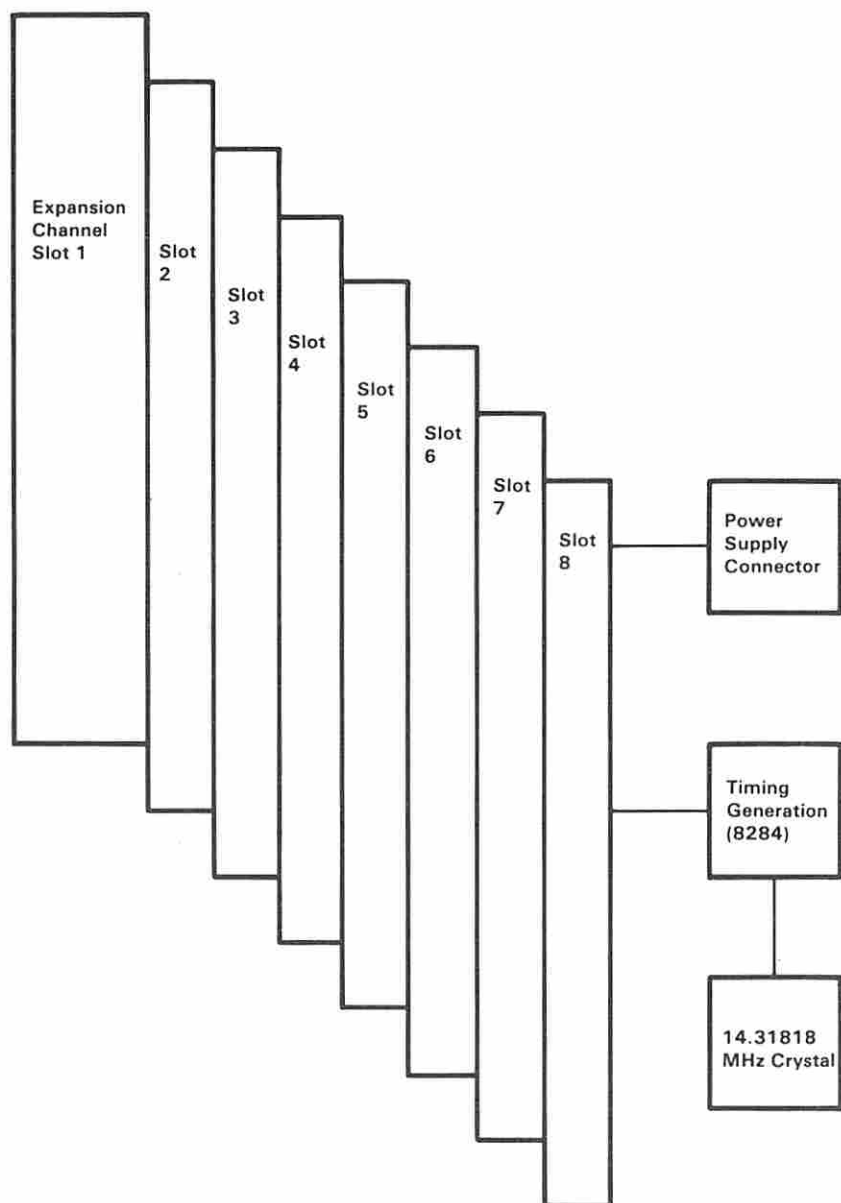
The expansion unit option upgrades the IBM Personal Computer by adding expansion slots in a separate unit. This option consists of an extender card, an expansion cable, and the expansion unit. The expansion unit contains a power supply, an expansion board, and a receiver card. This option utilizes one expansion slot in the system unit to provide seven additional expansion slots in the expansion unit.

Expansion Unit Cable

The expansion unit cable consists of a 56-wire, foil-shielded cable terminated on each end with a 62-pin D-shell male connector. Either end of the expansion unit cable can be plugged into the extender card or the receiver card.

Expansion Board

The expansion board is a support board that carries the I/O channel signals from the option adapters and receiver card. These signals, except 'osc,' are carried over the expansion cable. Because 'osc' is not sent over the expansion cable, a 14.31818-MHz signal is generated on the expansion board. This signal may not be in phase with the 'osc' signal in the system unit. Decoupling capacitors provided on the expansion board aid in noise filtering.



Expansion Board Block Diagram

Expansion Channel

All signals found on the system unit's I/O channel will be provided to expansion slots in the expansion unit, with the exception of the 'osc' signal and the voltages mentioned previously.

A 'ready' line on the expansion channel makes it possible to operate with slow I/O or memory devices. If the channel's 'I/O ch rdy' line is not activated by an addressed device, all processor-generated memory cycles take five processor clock cycles per byte for memory in the expansion unit.

The following table contains a list of all the signals that are redriven by the extender and receiver cards, and their associated time delays. The delay times include the delay due to signal propagation in the expansion cable. Assume a nominal cable delay of 3 ns. As such, device access will be less than 260 ns.

Signal	Nominal Delay (ns)	Maximum Delay (ns)	Direction (*)
A0 - A19	27	39	Output
AEN	27	39	Output
DACK0 - DACK3	27	39	Output
MEMR	27	39	Output
MEMW	51	75	Output
IOR	51	75	Output
IOW	27	39	Output
ALE	27	39	Output
CLK	27	39	Output
T/C	27	39	Output
RESET	27	39	Output
IRQ2 - IRQ7	36	(**)	Input
DRQ1 - DRQ3	36	(**)	Input
I/O CH RDY	36	51	Input
I/O CH CK	36	51	Input
D0 - D7 (Read)	84	133	Input
D0 - D7 (Write)	19	27	Output

(*) With respect to the system unit.

(**) Asynchronous nature of interrupts and other requests are more dependent on processor recognition than electrical signal propagation through expansion logic.

Power Supply

The expansion unit dc power supply is a 130-watt, 4 voltage level switching regulator. It is integrated into the expansion unit and supplies power for the expansion unit, and its options. The supply provides 15 A of +5 Vdc, plus or minus 5%, 4.2 A of +12 Vdc, plus or minus 5%, 300 mA of -5 Vdc, plus or minus 10%, and 250 mA of -12 Vdc, plus or minus 10%. All power levels are regulated with over-voltage and over-current protection. The input is 120 Vac and fused. If dc over-load or over-voltage conditions exist, the supply automatically shuts down until the condition is corrected. The supply is designed for continuous operation at 130 watts.

The power supply is located at the right rear of the expansion unit. It supplies operating voltages to the expansion board, and provides two separate connections for power to the fixed disk drives. The nominal power requirements and output voltages are listed in the following tables:

Voltage (Vac at 50/60 Hz)			Frequency (Hz)	Current (Amps)
Nominal	Minimum	Maximum	+/- 3 Hz	Maximum
120	90	137	50/60	4.1 at 90 Vac
220/240	180	259	50/60	1.75 at 180 Vac

Input Requirements

Voltage (Vdc)	Current (Amps)		Regulation (Tolerance)	
Nominal	Minimum	Maximum	+	-
+5.0	2.3	15.0	5	4
-5.0	0.0	0.3	10	8
+12.0	0.4	4.2	5	4
-12.0	0.0	0.25	10	9

Vdc Output

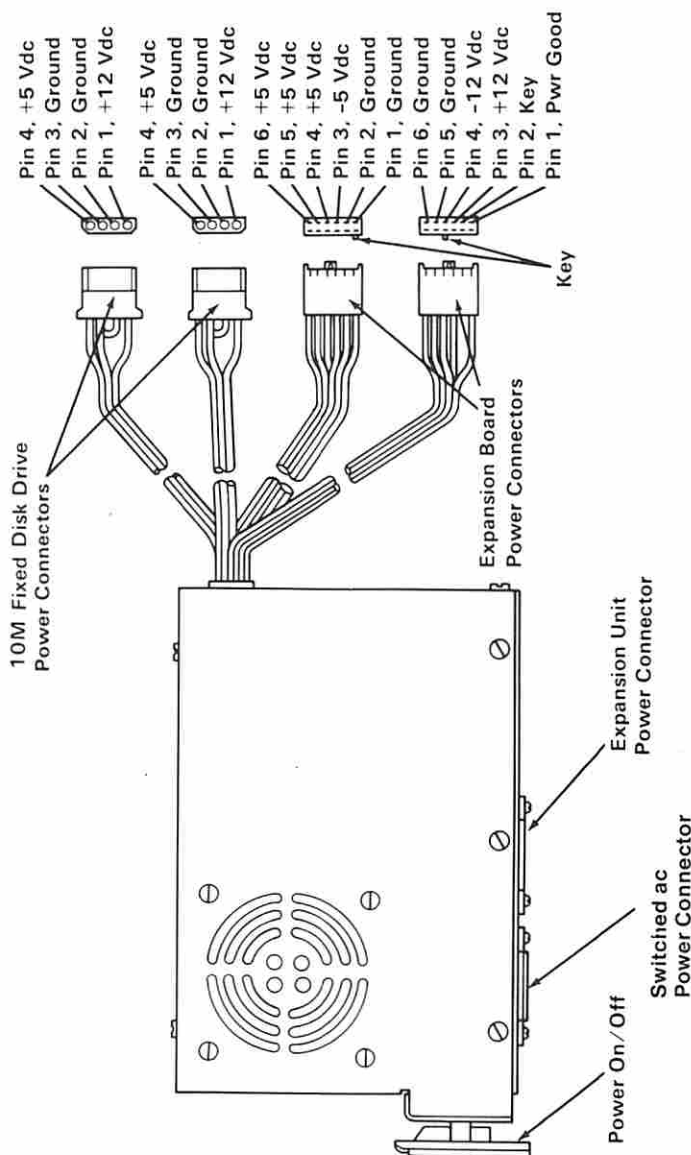
Voltage (Vac)	Current (Amps)		Voltage Limits (Vac)	
Nominal	Minimum	Maximum	Minimum	Maximum
120	0.0	1.0	88	137
220/240	0.0	0.5	178	257

Vac Output

1-42 Expansion Unit

Power Supply Connectors and Pin Assignments

The power connector on the expansion board is a 12-pin male connector that plugs into the power-supply connectors. The pin configurations and locations are shown below:



Power Supply and Connectors

Over-Voltage/Over-Current Protection

Voltage (Nominal Vac)	Type Protection	Rating (Amps)
120	Fuse	5
220/240	Fuse	5

Power On/Off Cycle: When the supply is turned off for a minimum of 1.0 second, and then turned on, the power-good signal will be regenerated.

The power-good signal indicates that there is adequate power to continue processing. If the power goes below the specified levels, the power-good signal triggers a system shutdown.

This signal is the logical AND of the dc output-voltage sense signal and the ac input voltage fail signal. This signal is TTL-compatible up-level for normal operation or down-level for fault conditions. The ac fail signal causes power-good to go to a down-level when any output voltage falls below the regulation limits.

The dc output-voltage sense signal holds the power-good signal at a down level (during power-on) until all output voltages have reached their respective minimum sense levels. The power-good signal has a turn-on delay of at least 100 ms but no greater than 500 ms.

The sense levels of the dc outputs are:

Output (Vdc)	Minimum (Vdc)	Sense Voltage Nominal (Vdc)	Maximum (Vdc)
+5	+4.5	+5.0	+5.5
-5	-4.3	-5.0	-5.5
+12	+10.8	+12.0	+13.2
-12	-10.2	-12.0	-13.2

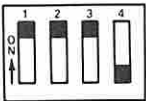
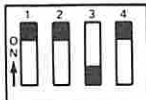
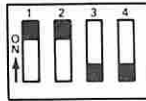
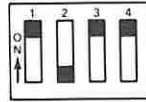
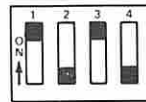
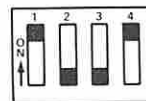
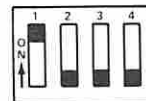
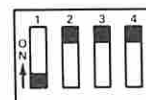
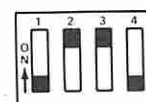
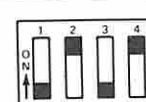
Extender Card

The extender card is a four-plane card. The extender card redrives the I/O channel to provide sufficient power to avoid capacitive effects of the cable. The extender card presents only one load per line of the I/O channel.

The extender card has a wait-state generator that inserts a wait-state on 'memory read' and 'memory write' operations (except refreshing) for all memory contained in the expansion unit. The address range for wait-state generation is controlled by switch settings on the extender card.

The DIP switch on the extender card should be set to indicate the maximum contiguous read/write memory housed in the system unit. The extender card switch setting figure lists memory in the system unit and the appropriate switch settings. Switch positions 1 through 4 correspond to address bits hex A19 to hex A16, respectively.

The switch settings determine which address segments have a wait state inserted during 'memory read' and 'memory write' operations. Wait states are required for any memory, including ROM on option adapters, in the expansion unit. Wait states are not inserted in the highest segment, hex addresses F0000 to FFFFF (segment F).

System Memory		Memory Segment
16K to 64K		1
96K to 128K		2
160K to 192K		3
224K to 256K		4
288K to 320K		5
352K to 384K		6
416K to 448K		7
480K to 512K		8
544K to 576K		9
608K to 640K		A

Extender Card Switch Settings

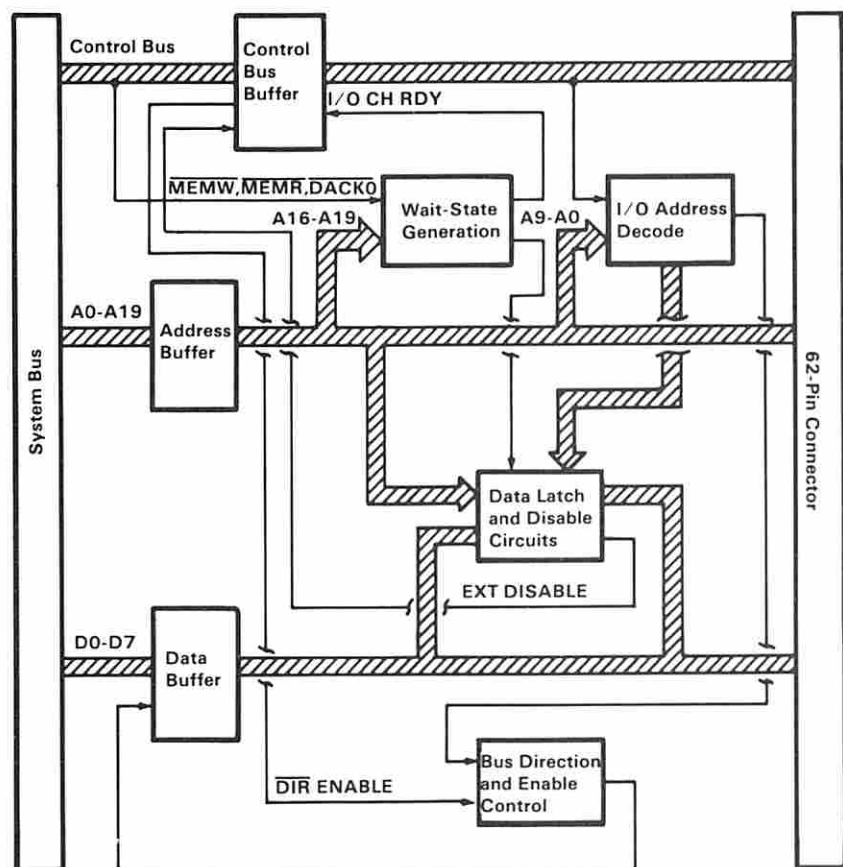
1-46 Expansion Unit

Extender Card Programming Considerations

Several registers associated with the expansion option are programmable and readable for diagnostic purposes. The following figure indicates the locations and functions of the registers on the extender card.

Location	Function
Memory FXXXX(*)	Write to memory to latch address bits
Port 210	Write to latch expansion bus data (ED7 - ED0)
Port 210	Read to verify expansion bus data (ED7 - ED0)
Port 211	Read high-order address bits (A8 - A15)
Port 211	Write to clear wait test latch
Port 212	Read low-order address bits (A0 - A7)
Port 213	Write 00 to disable expansion unit
Port 213	Write 01 to enable expansion unit
Port 213	Read status of expansion unit D0 = enable/disable D1 = wait-state request flag D2-D3 = not used D4-D7 = switch position 1 = Off 0 = On
(*) Example: Write to memory location F123:4=00 Read Port 211 = 12 Read Port 212 = 34 (All values in hex)	

The expansion unit is automatically enabled upon power-up. The extender card and receiver card will both be written to, if the expansion unit is not disabled when writing to FXXXX. However, the system unit and the expansion unit are read back separately.



Extender Card Block Diagram

Receiver Card

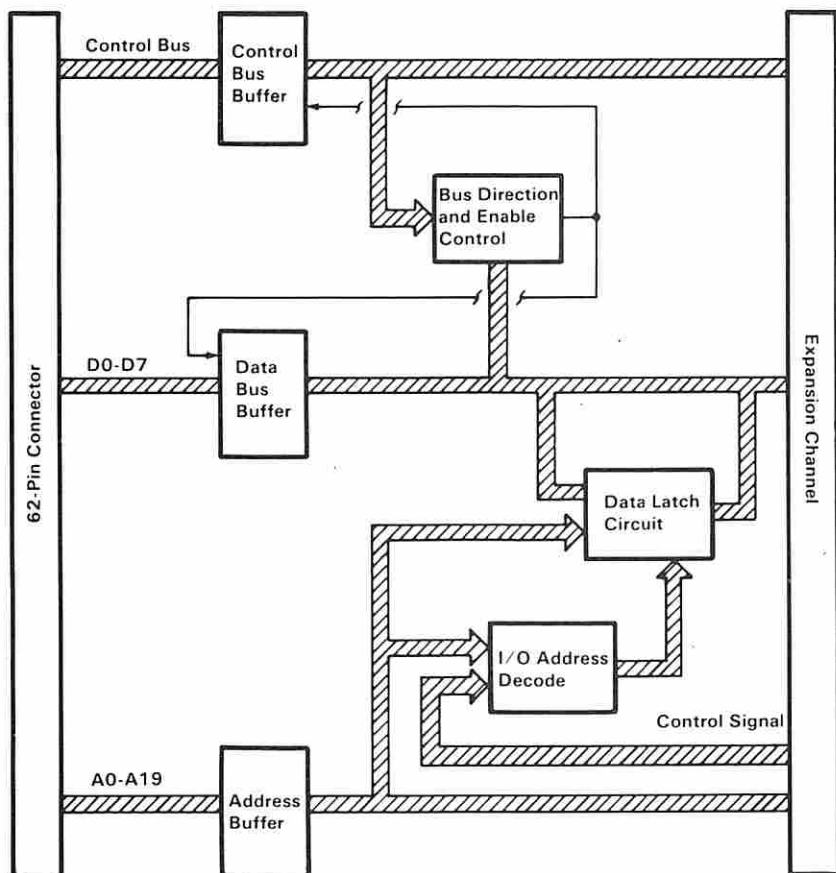
The receiver card is a four-plane card that fits in expansion slot 8 of the expansion unit. The receiver card redrives the I/O channel to provide sufficient power for additional options and to avoid capacitive effects. Directional control logic is contained on the receiver card to resolve contention and direct data flow on the I/O channel. Steering signals are transmitted back over the expansion cable for use on the extender card.

Receiver Card Programming Considerations

Several registers associated with the expansion option are programmable and readable for diagnostic purposes. The following figure indicates the locations and functions of the registers on the receiver card.

Location	Function
Memory FXXXX(*)	Write to memory to latch address bits
Port 214	Write to latch data bus bits (D0 - D7)
Port 214	Read data bus bits (D0 - D7)
Port 215	Read high-order address bits (A8 - A15)
Port 216	Read low-order address bits (A0 - A7)
(*) Example: Write to memory location F123:4=00 Read Port 215 =12 Read Port 216 =34 (All values in hex)	

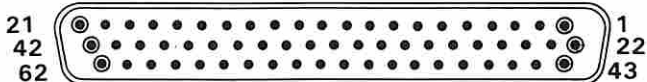
The expansion unit is automatically enabled upon power-up. The expansion unit and the system unit will be written to, if the expansion unit is not disabled when writing to FXXXX. However, the system unit and the expansion unit are read back separately.



Receiver Card Block Diagram

Expansion Unit Interface Information

The extender card and receiver card rear-panel connectors are the same. Pin and signal assignments for the extender and receiver cards are shown below.

					
Pin	Signal	Pin	Signal	Pin	Signal
1	+E IRQ6	22	+E D5	43	+E IRQ7
2	+E DRQ2	23	+E DRQ1	44	+E D6
3	+E DIR	24	+E DRQ3	45	+E I/O CH RDY
4	+E ENABLE	25	RESERVED	46	+E IRQ3
5	+E CLK	26	+E ALE	47	+E D7
6	-E MEM IN EXP	27	+E T/C	48	+E D1
7	+E A17	28	+E RESET	49	-E I/O CH CK
8	+E A16	29	+E AEN	50	+E IRQ2
9	+E A5	30	+E A19	51	+E D0
10	-E DACK0	31	+E A14	52	+E D2
11	+E A15	32	+E A12	53	+E D4
12	+E A11	33	+E A18	54	+E IRQ5
13	+E A10	34	-E MEMR	55	+E IRQ4
14	+E A9	35	-E MEMW	56	+E D3
15	+E A1	36	+E A0	57	GND
16	+E A3	37	-E DACK3	58	GND
17	-E DACK1	38	+E A6	59	GND
18	+E A4	39	-E IOR	60	GND
19	-E DACK2	40	+E A8	61	GND
20	-E IOW	41	+E A2	62	GND
21	+E A13	42	+E A7		

E = Extended

Connector Specifications

Notes:

IBM 80 CPS Graphics Printer

The IBM 80 CPS (characters-per-second) Printers are self-powered, stand-alone, tabletop units. They attach to the system unit through a parallel signal cable, 6 feet in length. The units obtain ac power from a standard wall outlet (120 Vac). The printers are 80 cps, bidirectional, wire-matrix devices. They print characters in a 9 by 9 dot matrix with a 9-wire head. They can print in a compressed mode of 132 characters per line, in a standard mode of 80 characters per line, in a double width, compressed mode of 66 characters per line, and in a double width mode of 40 characters per line. The printers can print double-size characters and double-strike characters. The printers print the standard ASCII, 96-character, uppercase and lowercase character sets. A printer without an extended character set also has a set of 64 special block graphic characters.

The IBM 80 CPS Graphics Printer has additional capabilities including: an extended character set for international languages, subscript, superscript, an underline mode, and programmable graphics.

The printers can also accept commands setting the line-feed control desired for the application. They attach to the system unit through the printer adapter or the combination monochrome display and printer adapter. The cable is a 25-lead shielded cable with a 25-pin D-shell connector at the system unit end, and a 36-pin connector at the printer end.

(1) Print Method:	Serial Impact dot matrix	
(2) Print Speed:	80 CPS	
(3) Print Direction:	Bidirectional with logical seeking	
(4) Number of Pins in Head	9	
(5) Line Spacing:	4.23 mm (1/6 inch) or programmable	
(6) Printing Characteristics		
Matrix:	9x9	
Character Set 1:	Full 96-character ASCII with descenders, additional ASCII numbers 160 to 175 are European characters. 176 to 223 are graphic characters. 224 to 239 are selected Greek characters. 240 to 255 contain math and extra symbols.	
Character Set 2:	Same as above except: ASCII numbers 3,4,5,6, and 21 contain symbols. ASCII numbers 128 to 175 contain European Characters.	
Graphics:	20 block characters and programmable graphics.	
(7) Printing Sizes	Characters	Maximum characters
	per inch	per line
Normal:	10	80
Enlarged:	5	40
Condensed:	16.5	132
Condensed Enlarged:	8.25	66
Subscript or Superscript:	10	80
(8) Media Handling		
Paper Feed:	Adjustable sprocket pin feed	
Paper Width Range:	101.6 mm (4 inches) to 254 mm (10 inches)	
Copies:	One original plus two carbon copies, total thickness not to exceed 0.3 mm (0.012 inch). Minimum paper thickness is 0.064 mm or 0.0025 inch.	
Paper Path:	Rear	
(9) Interfaces		
Standard:	Parallel 8-bit Data Data and Control lines	
(10) Inked Ribbon		
Color:	Black	
Type:	Cartridge	
Life Expectancy:	3 million characters	

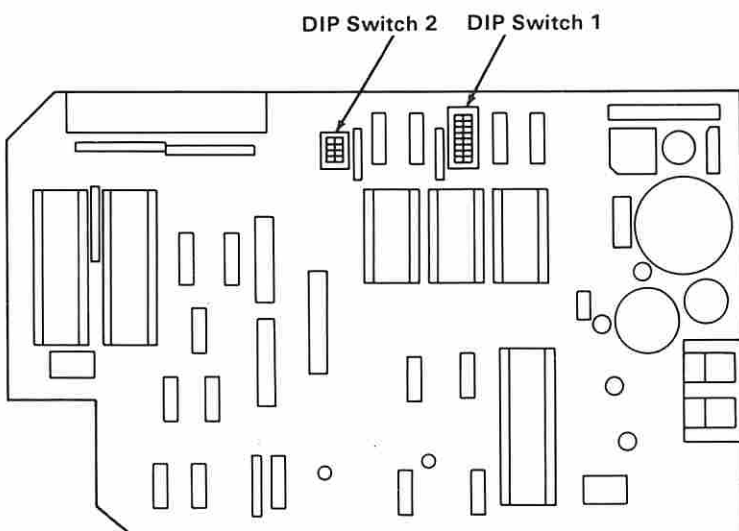
Printer Specifications

Printer Specifications (continued)

(11) Environmental Conditions				
Operation Temperature Range:		5 to 35°C (41 to 95°F)		
Operating Humidity:		10 to 80% non-condensing		
(12) Power Requirements		Three models available:		
Voltage (AC):		120	220	240
Frequency (Hz):		60	50/60	50/60
Maximum Current (Amps):		1	.5	.5
Maximum Power (Watts):		100	100	100
(13) Physical Characteristics				
Height:		107 mm (4.2 inches)		
Width:		374 mm (14.7 inches)		
Depth:		305 mm (12.0 inches)		
Weight:		5.5 kg (12 lbs.)		

Setting the DIP Switches

There are two DIP switches on the control circuit board. In order to satisfy the user's specific requirements, desired control modes are selectable by the DIP switches. The functions of the switches and their preset conditions at the time of shipment are as shown in the following figures.



Location of Printer DIP Switches

Switch Number	Function	On	Off	Factory-Set Condition
1-1	Not Applicable	—	—	On
1-2	CR	Print Only	Print & Line Feed	On
1-3	Buffer Full	Print Only	Print & Line Feed	Off
1-4	Cancel Code	Invalid	Valid	Off
1-5	Not Applicable	—	—	On
1-6	Error Buzzer	Sound	Does Not Sound	On
1-7	Character Generator	Set 2	Set 1	On
1-8	SLCT IN Signal	Fixed Internally	Not Fixed Internally	On

Functions and Conditions of DIP Switch 1

Switch Number	Function	On	Off	Factory-Set Condition
2-1	Form Length	12 Inches 304.8 mm	11 Inches 279.4 mm	Off
2-2	Line Spacing	1/8 Inch 3.175 mm	1/6 Inch 4.23 mm	Off
2-3	Auto Feed XT Signal	Fixed Internally	Not Fixed Internally	Off
2-4	1 Inch Skip Over Perforation	Valid	Not Valid	Off

Functions and Conditions of DIP Switch 2

Parallel Interface Description

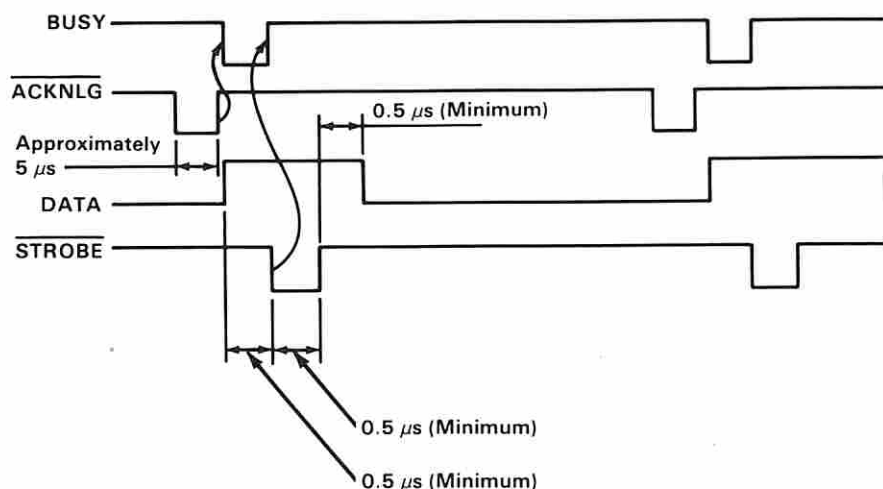
Specifications:

- Data transfer rate: 1000 cps (maximum)
- Synchronization: By externally-supplied STROBE pulses.
- Handshaking ACKNLG or BUSY signals.
- Logic level: Input data and all interface control signals are compatible with the TTL level.

Connector: Plug: 57-30360 (Amphenol)

Connector pin assignment and descriptions of respective interface signals are provided on the following pages.

Data transfer sequence:



Parallel Interface Timing Diagram

Signal Pin No.	Return Pin No.	Signal	Direction	Description
1	19	$\overline{\text{STROBE}}$	In	STROBE pulse to read data in. Pulse width must be more than 0.5 μs at receiving terminal. The signal level is normally "high"; read-in of data is performed at the "low" level of this signal.
2	20	DATA 1	In	These signals represent information of the 1st to 8th bits of parallel data respectively. Each signal is at "high" level when data is logical "1" and "low" when logical "0."
3	21	DATA 2	In	
4	22	DATA 3	In	
5	23	DATA 4	In	
6	24	DATA 5	In	
7	25	DATA 6	In	
8	26	DATA 7	In	
9	27	DATA 8	In	
10	28	$\overline{\text{ACKNLG}}$	Out	Approximately 5 μs pulse; "low" indicates that data has been received and the printer is ready to accept other data.
11	29	BUSY	Out	A "high" signal indicates that the printer cannot receive data. The signal becomes "high" in the following cases: 1. During data entry. 2. During printing operation. 3. In "offline" state. 4. During printer error status.

Connector Pin Assignment and Descriptions of Interface Signals (Part 1 of 3)

Signal Pin No.	Return Pin No.	Signal	Direction	Description
12	30	PE	Out	A "high" signal indicates that the printer is out of paper.
13	—	SLCT	Out	This signal indicates that the printer is in the selected state.
14	—	AUTO FEED XT	In	With this signal being at "low" level, the paper is automatically fed one line after printing. (The signal level can be fixed to "low" with DIP SW pin 2-3 provided on the control circuit board.)
15	—	NC		Not used.
16	—	OV		Logic GND level.
17	—	CHASSIS- GND	—	Printer chassis GND. In the printer, the chassis GND and the logic GND are isolated from each other.
18	—	NC	—	Not used.
19-30	—	GND	—	"Twisted-Pair Return" signal; GND level.
31	—	INIT	In	When the level of this signal becomes "low" the printer controller is reset to its initial state and the print buffer is cleared. This signal is normally at "high" level, and its pulse width must be more than 50 μ s at the receiving terminal.

Connector Pin Assignment and Descriptions of Interface Signals (Part 2 of 3)

Signal Pin No.	Return Pin No.	Signal	Direction	Description
32		<u>ERROR</u>	Out	The level of this signal becomes "low" when the printer is in "Paper End" state, "Offline" state and "Error" state.
33	—	GND	—	Same as with pin numbers 19 to 30.
34	—	NC	—	Not used.
35				Pulled up to +5 Vdc through 4.7 k-ohms resistance.
36	—	<u>SLCT IN</u>	In	Data entry to the printer is possible only when the level of this signal is "low." (Internal fixing can be carried out with DIP SW 1-8. The condition at the time of shipment is set "low" for this signal.)

- Notes:**
1. "Direction" refers to the direction of signal flow as viewed from the printer.
 2. "Return" denotes "Twisted-Pair Return" and is to be connected at signal-ground level.
When wiring the interface, be sure to use a twisted-pair cable for each signal and never fail to complete connection on the return side. To prevent noise effectively, these cables should be shielded and connected to the chassis of the system unit and printer, respectively.
 3. All interface conditions are based on TTL level. Both the rise and fall times of each signal must be less than 0.2 μ s.
 4. Data transfer must not be carried out by ignoring the ACKNLG or BUSY signal. (Data transfer to this printer can be carried out only after confirming the ACKNLG signal or when the level of the BUSY signal is "low.")

Connector Pin Assignment and Descriptions of Interface Signals (Part 3 of 3)

Printer Modes for the IBM 80 CPS Graphics Printer

The IBM 80 CPS Graphics Printer can use any of the combinations listed below, and the print mode can be changed at any place within a line.

The allowed combinations of print modes that can be selected are listed in the following table. Modes can be selected and combined if they are in the same vertical column.

Printer Modes											
Normal	X	X	X								
Compressed					X	X	X				
Emphasized									X	X	X
Double Strike	X				X				X		
Subscript		X				X				X	
Superscript			X				X				X
Double Width	X	X	X		X	X	X		X	X	X
Underline	X	X	X		X	X	X		X	X	X

Printer Control Codes

On the following pages you will find complete codes for printer control, characters, and graphics. You may want to keep them handy for future reference. The printer codes are listed in ASCII decimal order (e.g. NUL is zero, BEL is 7, etc.). Examples given in the Printer Function descriptions are written in BASIC. The "Format" description is given when more information is needed for programming considerations. ASCII decimal values for the printer control codes can be found in the Printer Character Sets which follow these control codes.

Printer

The descriptions that follow assume that the printer DIP switches have not been changed from their factory settings.

Printer Code	Printer Function
NUL	Null Used with ESC B and ESC D as a list terminator. NUL is also used with other Printer Control Codes to select options (for example, ESC S). Example: <pre>LPRINT CHR\$(0);</pre>
BEL	Bell Sounds the printer buzzer for 1 second. Example: <pre>LPRINT CHR\$(7);</pre>
HT	Horizontal Tab Tabs to the next horizontal tab stop. Tab stops are set with ESC D. When the printer is powered on or reset, tab stops are set every eight columns. Example: <pre>LPRINT CHR\$(9);</pre>
LF	Line Feed Prints data remaining in the printer buffer and spaces the paper up one line. Line spacing is 1/6-inch (4.23 mm) unless reset by ESC A, ESC 0, ESC 1, ESC 2, or ESC 3. Example: <pre>LPRINT CHR\$(10);</pre>
VT	Vertical Tab The VT code is treated as a LF. Example: <pre>LPRINT CHR\$(11);</pre>
FF	Form Feed Advances the paper to the top of the next page. Note: The location of the paper, when the printer is powered on or reset, determines the top of the page. The next top of page is 11 inches (279.4 mm) from that position. ESC C can be used to change the page length. Example: <pre>LPRINT CHR\$(12);</pre>

Printer Code	Printer Function
CR	Carriage Return Ends the line the printer is on and prints the data remaining in the printer buffer. (No line feed operation is initiated by the printer.) Example: LPRINT CHR\$(13); Note: IBM Personal Computer Basic adds a Line Feed command to every Carriage Return unless you select character set 1, add 128 to the CR, and return to character set 2. (See ESC 6 and ESC 7.) Example: LPRINT "0000";CHR\$(27);CHR\$(55);CHR\$(13+128); CHR\$(27);CHR\$(54);" / / /"
SO	Shift Out (Double Width) Changes to the Double Width print mode. Note: A Carriage Return, Line Feed, DC 4, or CAN cancels Double Width print mode initiated with SO. (See ESC W for continuous Double Width printing.) Example: LPRINT CHR\$(14);
SI	Shift In (Compressed) Changes the printer to the Compressed Character print mode. Example: LPRINT CHR\$(15);
DC2	Device Control 2 (Compressed Off) Stops printing in the Compressed Character print mode. Example: LPRINT CHR\$(18);
DC4	Device Control 4 (Double Width Off) Stops printing in the Double Width print mode started by the SO command. Example: LPRINT CHR\$(20);
CAN	Cancel Clears the printer buffer without printing data. Control codes, except SO, remain in effect. Example: LPRINT CHR\$(24);

Printer Code	Printer Function
ESC	Escape Lets the printer know that the next data sent is a printer command. (See the following commands.) Example: LPRINT CHR\$(27);
ESC -	Escape Minus (Underline) Format: ESC -;n; ESC - followed by a 1, prints all of the data following it with an underline. ESC - followed by a 0 (zero) cancels the Underline print mode. Example: LPRINT CHR\$(27);CHR\$(45);CHR\$(1);
ESC 0	Escape Zero (1/8 inch Line Feeding) Changes paper feeding to 1/8 inch (3.175 mm). Example: LPRINT CHR\$(27);CHR\$(48);
ESC 1	Escape One (7/72 inch Line Feeding) Changes paper feeding to 7/72 inch (2.47 mm). Example: LPRINT CHR\$(27);CHR\$(49);
ESC 2	Escape Two (Starts Variable Line Feeding) ESC 2 is an execution command for ESC A. If no ESC A command has been given, line feeding is executed at 1/6-inch (4.23 mm). Example: LPRINT CHR\$(27);CHR\$(50);
ESC 3	Escape Three (Variable Line Feeding) Format: ESC 3;n; Changes the paper feeding to n/216 inch (1/216-inch is 0.1176 mm). A value from 1 to 255 must be assigned to n. The example below sets the paper feeding to 54/216 (1/4) inch (6.35 mm). Example: LPRINT CHR\$(27);CHR\$(51);CHR\$(54);
ESC 6	Escape Six (Select Character Set 2) Selects character set 2. (See "Printer Character Set 2.") Character Set 2 is selected when the printer is powered on or reset. Example: LPRINT CHR\$(27);CHR\$(54);
ESC 7	Escape Seven (Select Character Set 1) Selects character set 1. (See "Printer Character Set 1.") Example: LPRINT CHR\$(27);CHR\$(55);

Printer Code	Printer Function
ESC 8	Escape Eight (Ignore Paper End) Allows the printer to print to the end of the paper. The printer¹ ignores the paper end switch. This command must be sent prior to an actual paper end. Example: LPRINT CHR\$(27);CHR\$(56);
ESC 9	Escape Nine (Cancel Ignore Paper End) Cancels the Ignore Paper End command. ESC 9 is selected when the printer is powered on or reset. Example: LPRINT CHR\$(27);CHR\$(57);
ESC <	Escape Less Than (Home Head) The print head will return to the left margin to print the line following ESC <. This will occur for one line only. Example: LPRINT CHR\$(27);CHR\$(60);
ESC A	Escape A (Sets the variable line spacing) Format: ESC A;n; Escape A sets the line feed spacing to n/72 inch (1/72 inch is 0.3528 mm). The example below tells the printer to set line feeding to 24/72 inch (8.47 mm). ESC 2 must be sent to the printer before the line feeding will change, e.g. ESC A;24 (text) ESC 2 (text). The text following ESC A;24 will space at previously set line feed increments. The text following ESC 2 will be printed with new line feed increments of 24/72 inch (8.47 mm). Any increment between 1/72 and 85/72 may be used. Example: LPRINT CHR\$(27);CHR\$(65);CHR\$(24);CHR\$(27);CHR\$(50);
ESC C	Escape C (Set lines per page) Format: ESC C;n; Sets the page length in number of lines. The ESC C command must have a value (from 1 to 127) following it to specify the number of lines desired on the page. The example below sets the page length to 55 lines. The printer defaults to 66 lines per page when powered on or reset. Example: LPRINT CHR\$(27);CHR\$(67);CHR\$(55); Escape C (Set inches per page) Format: ESC C;n;m; Sets the length of the page in inches (one inch is 25.4 mm). This command requires a value of 0 (zero) for n and a value between 1 and 22 for m. Example: LPRINT CHR\$(27);CHR\$(67);CHR\$(0);CHR\$(12);

Printer Code	Printer Function
ESC D	Escape D (Set Horizontal Tab Stops) Format: ESC D;n₁;n₂...n_K;NUL; Sets the horizontal tab stop positions. The example below shows the horizontal tab stop positions set at printer column positions of 10, 20, and 40. They are followed by CHR\$(0), the NUL code. They must be in ascending numerical order as shown. Tab stops can be set between 1 and 80 in normal print mode. When in the compressed mode, tab stops can be set up to 132. Double Width characters take up two column positions. The maximum number of tabs that can be set is 28. The HT [CHR\$(9)] is used to execute a tab operation. Example: <pre>LPRINT CHR\$(27);CHR\$(68);CHR\$(10);CHR\$(20);CHR\$(40);CHR\$(0);</pre>
ESC E	Escape E (Emphasized) Changes the printer to the Emphasized print mode. The printer speed is reduced to half during the Emphasized print mode. Example: <pre>LPRINT CHR\$(27);CHR\$(69);</pre>
ESC F	Escape F (Emphasized Off) Stops printing in the Emphasized print mode. Example: <pre>LPRINT CHR\$(27);CHR\$(70);</pre>
ESC G	Escape G (Double Strike) Changes the printer to the Double Strike print mode. The paper is spaced 1/216-inch (0.1176 mm) before the second pass of the print head. Example: <pre>LPRINT CHR\$(27);CHR\$(71);</pre>
ESC H	Escape H (Double Strike Off) Stops printing in the Double Strike mode. Example: <pre>LPRINT CHR\$(27);CHR\$(72);</pre>
ESC J	Escape J (Feed Paper n/216 inch) Format: ESC J;n; When ESC J is sent to the printer, paper will be fed n/216 inch, (1/216 inch is 0.1176 mm). The value of "n" must be from 1 to 255. The example below produces a paper feed of 50/216 inch (5.88 mm). ESC J is canceled after the paper feed takes place. Example: <pre>LPRINT CHR\$(27);CHR\$(74);CHR\$(50);</pre>

Printer Code	Printer Function																																																																									
ESC K	Escape K (480 Bit-Image Graphics Mode) Format: ESC K; n_1; n_2; v_1; v_2; ... v_K Changes from the Text mode to the Bit Image Graphics mode. n_1 and n_2 are numbers, each consisting of 1 byte, which specify the total number of bit-image data bytes to be transferred. v_1 through v_K are the bytes of bit-image data whose total number (K) cannot exceed 480 and must be equal to $n_1 + 256n_2$. At every horizontal position, each byte can print up to 8 vertical dots. Bit-image data may be mixed with Text data on the same line. Note: Assign values to n_1 and n_2 as follows. n_1 represents values from 0 – 255. n_2 represents values from 0 – 255 × 256. MSB is Most Significant Bit and LSB is Least Significant Bit. n_2 n_1 <table border="1" style="width: 100%; text-align: center;"> <tr> <td colspan="4">MSB</td> <td colspan="4">LSB</td> <td colspan="4">MSB</td> <td colspan="4">LSB</td> </tr> <tr> <td>2^{15}</td><td>2^{14}</td><td>2^{13}</td><td>2^{12}</td> <td>2^{11}</td><td>2^{10}</td><td>2^9</td><td>2^8</td> <td>2^7</td><td>2^6</td><td>2^5</td><td>2^4</td> <td>2^3</td><td>2^2</td><td>2^1</td><td>2^0</td> </tr> </table> Data sent to the printer. <table border="1" style="width: 100%; text-align: center;"> <tr> <td>Text (20 characters)</td> <td>ESC K</td> <td>$n=360$</td> <td>Bit-image data</td> <td>Next data</td> </tr> </table> 20 characters in text mode correspond to 120 bit-image positions ($20 \times 6 = 120$). The printable portion left in Bit-image mode is 360 dot positions. ($480 - 120 = 360$) Data sent to the printer. <table border="1" style="width: 100%; text-align: center;"> <tr> <td colspan="4"></td> <td colspan="2">n_2, n_1</td> <td colspan="4"></td> <td colspan="2">n'_2, n'_1</td> </tr> <tr> <td>Data A</td> <td>ESC K</td> <td>n_1</td> <td>n_2</td> <td>Data B</td> <td>Data C</td> <td>ESC K</td> <td>n'_1</td> <td>n'_2</td> <td>Data D</td> <td></td> <td></td> </tr> <tr> <td>Text data</td> <td></td> <td>Length of data</td> <td>Bit-image data</td> <td>Text data</td> <td></td> <td></td> <td>Length of data</td> <td>Bit-image data</td> <td></td> <td></td> <td></td> </tr> </table> ← → 480 bit-image dot positions	MSB				LSB				MSB				LSB				2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	Text (20 characters)	ESC K	$n=360$	Bit-image data	Next data					n_2, n_1						n'_2, n'_1		Data A	ESC K	n_1	n_2	Data B	Data C	ESC K	n'_1	n'_2	Data D			Text data		Length of data	Bit-image data	Text data			Length of data	Bit-image data			
MSB				LSB				MSB				LSB																																																														
2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0																																																											
Text (20 characters)	ESC K	$n=360$	Bit-image data	Next data																																																																						
				n_2, n_1						n'_2, n'_1																																																																
Data A	ESC K	n_1	n_2	Data B	Data C	ESC K	n'_1	n'_2	Data D																																																																	
Text data		Length of data	Bit-image data	Text data			Length of data	Bit-image data																																																																		

Printer Code	Printer Function
ESC K (cont)	Example: <pre> 1 'OPEN PRINTER IN RANDOM MODE WITH LENGTH OF 255 2 OPEN "LPT1:" AS #1: 3 WIDTH "LPT1:",255 4 PRINT #1,CHR\$(13);CHR\$(10); 5 SLASH\$=CHR\$(1)+CHR\$(2)+CHR\$(4)+CHR\$(8) 6 SLASH\$=SLASH\$+CHR\$(16)+CHR\$(32)+CHR\$(64) +CHR\$(128)+CHR\$(0) 7 GAPS=CHR\$(0)+CHR\$(0)+CHR\$(0) 8 NDOTS=480 9 'ESC K N1 N2 10 PRINT #1,CHR\$(27);"K";CHR\$(NDOTS MOD 256); CHR\$(INT (NDOTS/256)); 11 'SEND NDOTS NUMBER OF BIT IMAGE BYTES 12 FOR I=1 TO NDOTS/12 'NUMBER OF SLASHES TO PRINT USING GRAPHICS 13 PRINT #1,SLASH\$;GAPS; 14 NEXT I 15 CLOSE 16 END </pre> This example will give you a row of slashes printed in the 480 Bit Image mode.
ESC L	Escape L (960 Bit Image Graphics Mode) Format: ESC L;n_1;n_2;v_1;v_2;...v_K; Changes from the Text mode to the Bit Image Graphics mode. The input is similar to ESC K. The 960 Bit Image mode prints at half the speed of the 480 Bit Image mode, but can produce a denser graphic image. The number of bytes of Bit Image Data (v_K) is $n_1 + 256n_2$ but cannot exceed 960. n_1 is in the range of 0 to 255.
ESC N	Escape N (Set Skip Perforation) Format: ESC N;n; Sets the Skip Perforation function between pages. The number following ESC N sets the value for the number of lines of Skip Perforation. This value can be from 1 to 127. The example below shows a 12-line perforation skip. With a page length of 66 lines this will print 54 lines and feed the paper 12 lines. ESC N is reset anytime the page length (ESC C) is changed. Example: <pre>LPRINT CHR\$(27);CHR\$(78);CHR\$(12);</pre>
ESC O	Escape O (Cancel Skip Perforation) Cancels the Skip Perforation function. Example: <pre>LPRINT CHR\$(27);CHR\$(79);</pre>

Printer Code	Printer Function
ESC S	Escape S (Subscript/Superscript) Format: ESC S;n; Changes the printer to the Subscript print mode when ESC S is followed by a 1, as in the example below. When ESC S is followed by a 0 (zero), the printer will print in the Superscript print mode. Example: LPRINT CHR\$(27);CHR\$(83);CHR\$(1);
ESC T	Escape T (Subscript/Superscript Off) The printer stops printing in the Subscript or Superscript print mode. Example: LPRINT CHR\$(27);CHR\$(84);
ESC U	Escape U (Unidirectional Printing) Format: ESC U;n; The printer will print only from left to right following the input of ESC U;1. When ESC U is followed by a 0 (zero) the left to right printing operation is canceled. The Unidirectional print mode (ESC U) assures a more accurate printing start position for better print quality. Example: LPRINT CHR\$(27);CHR\$(85);CHR\$(1);
ESC W	Escape W (Double Width) Format: ESC W;n; Changes to the Double Width mode when ESC W is followed by a 1. This mode is not canceled by a line feed operation or by the DC4 code. It must be canceled with ESC W followed by a 0 (zero). Example: LPRINT CHR\$(27);CHR\$(87);CHR\$(1);
ESC Y	Escape Y (960 Bit Image Graphics Mode) Format: ESC Y;n₁;n₂;v₁;v₂;...v_K; Changes from the Text mode to the 960 Bit Image Graphics mode. The printer prints at normal speed during this operation and cannot print dots on consecutive dot positions. The input of data is similar to ESC L.
ESC Z	Escape Z (1920 Bit Image Graphics Mode) Format: ESC Z;n₁;n₂;v₁;v₂;...v_K; Changes from the Text mode to the 1920 Bit Image Graphics mode. The input is similar to the other Bit Image Graphics modes during this operation, but the total number (K) of Bit Image Data bytes allowed is 1920. ESC Z prints at normal speed but can only print every third dot position.

0	1	2	3	4	5	6	7	8	9
NUL							BEL		HT
10	11	12	13	14	15	16	17	18	19
LF	VT	FF	CR	SO	SI			DC2	
20	21	22	23	24	25	26	27	28	29
DC4				CAN			ESC		
30	31	32	33	34	35	36	37	38	39
		SP	!	"	#	\$	%	&	'
40	41	42	43	44	45	46	47	48	49
(	)	*	+	,	—	.	/	0	1
50	51	52	53	54	55	56	57	58	59
2	3	4	5	6	7	8	9	:	;
60	61	62	63	64	65	66	67	68	69
<	=	>	?	⌚	A	B	C	D	E
70	71	72	73	74	75	76	77	78	79
F	G	H	I	J	K	L	M	N	O
80	81	82	83	84	85	86	87	88	89
P	Q	R	S	T	U	V	W	X	Y
90	91	92	93	94	95	96	97	98	99
Z	[	\	]	^	_	`	a	b	c
100	101	102	103	104	105	106	107	108	109
d	e	f	g	h	i	j	k	l	m
110	111	112	113	114	115	116	117	118	119
n	o	p	q	r	s	t	u	v	w
120	121	122	123	124	125	126	127	128	129
x	y	z	{		}	~		NUL	

Graphics Printer Character Set 1 (Part 1 of 2)

130	131	132	133	134	135	136	137	138	139
					BEL		HT	LF	VT
140	141	142	143	144	145	146	147	148	149
FF	CR	SO	SI			DC2		DC4	
150	151	152	153	154	155	156	157	158	159
		CAN			ESC				
160	161	162	163	164	165	166	167	168	169
á	í	ó	ú	ñ	Ñ	<u>a</u>	<u>o</u>	¿	␣
170	171	172	173	174	175	176	177	178	179
␣	½	¼	¡	<<	>>				
180	181	182	183	184	185	186	187	188	189
									
190	191	192	193	194	195	196	197	198	199
									
200	201	202	203	204	205	206	207	208	209
									
210	211	212	213	214	215	216	217	218	219
									
220	221	222	223	224	225	226	227	228	229
									
230	231	232	233	234	235	236	237	238	239
									
240	241	242	243	244	245	246	247	248	249
≡	±	≥	≤		J	÷	≈	°	■
250	251	252	253	254	255				
-	√	n	2	■	SP				

Graphics Printer Character Set 1 (Part 2 of 2)

0	1	2	3	4	5	6	7	8	9
NUL			♥	♦	♣	♠	BEL		HT
10	11	12	13	14	15	16	17	18	19
LF	VT	FF	CR	SO	SI			DC2	
20	21	22	23	24	25	26	27	28	29
DC4	§			CAN			ESC		
30	31	32	33	34	35	36	37	38	39
		SP	!	''	#	\$	%	&	'
40	41	42	43	44	45	46	47	48	49
(	)	*	+	,	—	.	/	0	1
50	51	52	53	54	55	56	57	58	59
2	3	4	5	6	7	8	9	:	;
60	61	62	63	64	65	66	67	68	69
<	=	>	?	⌚	A	B	C	D	E
70	71	72	73	74	75	76	77	78	79
F	G	H	I	J	K	L	M	N	O
80	81	82	83	84	85	86	87	88	89
P	Q	R	S	T	U	V	W	X	Y
90	91	92	93	94	95	96	97	98	99
Z	[	\	]	^	_	`	a	b	c
100	101	102	103	104	105	106	107	108	109
d	e	f	g	h	i	j	k	l	m
110	111	112	113	114	115	116	117	118	119
n	o	p	q	r	s	t	u	v	w
120	121	122	123	124	125	126	127	128	129
x	y	z	{		}	~		ç	ü

Graphics Printer Character Set 2 (Part 1 of 2)

130	131	132	133	134	135	136	137	138	139
é	â	ä	à	å	ç	ê	ë	è	ï
140	141	142	143	144	145	146	147	148	149
î	ì	Ä	Å	É	æ	Æ	ô	ö	ò
150	151	152	153	154	155	156	157	158	159
û	ù	ÿ	ö	ü	ç	£	¥	₤	₣
160	161	162	163	164	165	166	167	168	169
á	í	ó	ú	ñ	Ñ	à	ó	¿	⌞
170	171	172	173	174	175	176	177	178	179
⌞	½	¼	¡	<<	>>	▒	▒	▒	▒
180	181	182	183	184	185	186	187	188	189
⌞	⌞	⌞	⌞	⌞	⌞	⌞	⌞	⌞	⌞
190	191	192	193	194	195	196	197	198	199
⌞	⌞	⌞	⌞	⌞	⌞	⌞	⌞	⌞	⌞
200	201	202	203	204	205	206	207	208	209
⌞	⌞	⌞	⌞	⌞	⌞	⌞	⌞	⌞	⌞
210	211	212	213	214	215	216	217	218	219
⌞	⌞	⌞	⌞	⌞	⌞	⌞	⌞	⌞	⌞
220	221	222	223	224	225	226	227	228	229
⌞	⌞	⌞	⌞	α	β	Γ	Π	Σ	σ
230	231	232	233	234	235	236	237	238	239
μ	τ	ϕ	θ	Ω	δ	∞	∅	€	∩
240	241	242	243	244	245	246	247	248	249
≡	±	≥	≤	∫	∫	÷	≈	°	■
250	251	252	253	254	255				
-	√	n	²	■	SP				

Graphics Printer Character Set 2 (Part 2 of 2)

Notes:

IBM Printer Adapter

The printer adapter is specifically designed to attach printers with a parallel port interface, but it can be used as a general input/output port for any device or application that matches its input/output capabilities. It has 12 TTL-buffer output points, which are latched and can be written and read under program control using the processor In or Out instruction. The adapter also has five steady-state input points that may be read using the processor's In instructions.

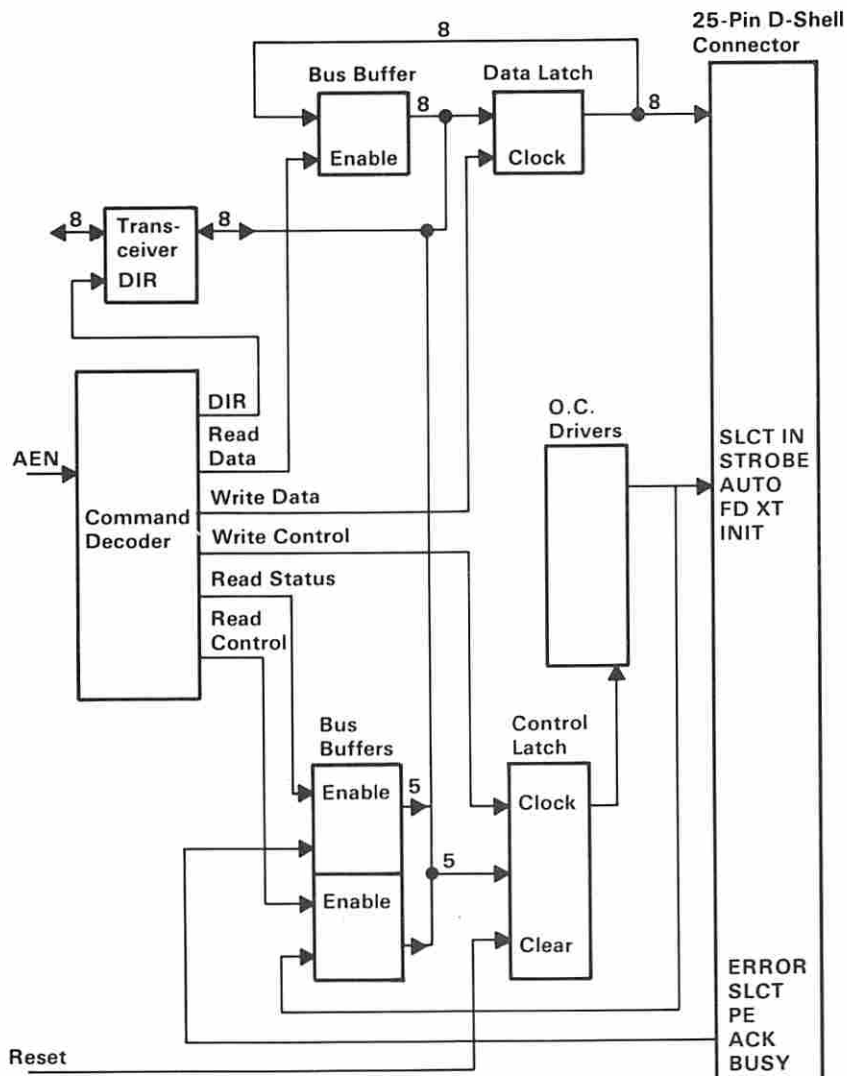
In addition, one input can also be used to create a processor interrupt. This interrupt can be enabled and disabled under program control. Reset from the power-on circuit is also ORed with a program output point, allowing a device to receive a power-on reset when the processor is reset.

The input/output signals are made available at the back of the adapter through a right-angled, PCB-mounted, 25-pin, D-shell connector. This connector protrudes through the rear panel of the system or expansion unit, where a cable may be attached.

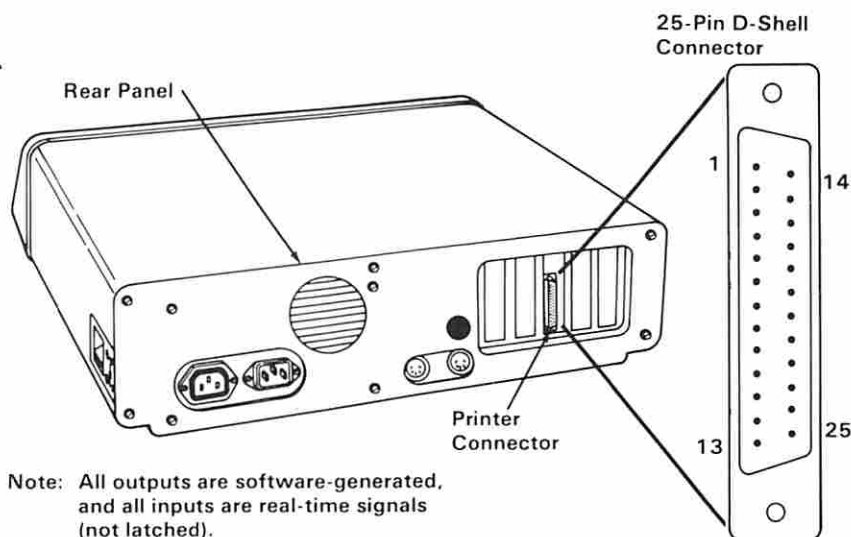
When this adapter is used to attach a printer, data or printer commands are loaded into an 8-bit, latched, output port, and the strobe line is activated, writing data to the printer. The program then may read the input ports for printer status indicating when the next character can be written, or it may use the interrupt line to indicate "not busy" to the software.

The output ports may also be read at the card's interface for diagnostic loop functions. This allows faults to be isolated between the adapter and the attaching device.

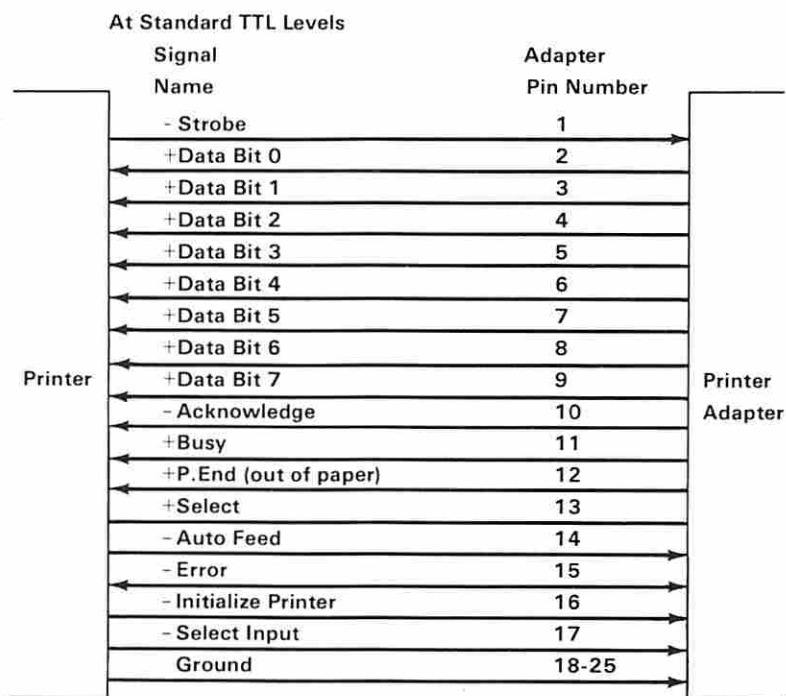
This same function is also part of the combination IBM Monochrome Display and Printer Adapter. A block diagram of the printer adapter is on the next page.



Printer Adapter Block Diagram



Note: All outputs are software-generated, and all inputs are real-time signals (not latched).



Connector Specifications

Notes:

Programming Considerations

The printer adapter responds to five I/O instructions: two output and three input. The output instructions transfer data into 2 latches whose outputs are presented on pins of a 25-pin D-shell connector.

Two of the three input instructions allow the processor to read back the contents of the two latches. The third allows the processor to read the real time status of a group of pins on the connector.

A description of each instruction follows.

IBM Monochrome Display & Printer Adapter				Printer Adapter			
Output to address hex 3BC				Output to address hex 378			
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Pin 9	Pin 8	Pin 7	Pin 6	Pin 5	Pin 4	Pin 3	Pin 2

The instruction captures data from the data bus and is present on the respective pins. These pins are each capable of sourcing 2.6 mA and sinking 24 mA.

It is essential that the external device not try to pull these lines to ground.

IBM Monochrome Display & Printer Adapter		Printer Adapter			
Output to address hex 3BE		Output to address hex 37A			
	Bit 4	$\overline{\text{Bit 3}}$	Bit 2	$\overline{\text{Bit 1}}$	$\overline{\text{Bit 0}}$
	IRQ Enable	Pin 17	Pin 16	Pin 14	Pin 1

This instruction causes the latch to capture the five least significant bits of the data bus. The four least significant bits present their outputs, or inverted versions of their outputs, to the respective pins shown above. If bit 4 is written as 1, the card will interrupt the processor on the condition that pin 10 transitions high to low.

These pins are driven by open collector drivers pulled to +5 Vdc through 4.7 k-ohm resistors. They can each sink approximately 7 mA and maintain 0.8 volts down-level.

IBM Monochrome Display & Printer Adapter	Printer Adapter
Input from address hex 3BC	Input from address hex 378

This command presents the processor with data present on the pins associated with the out to hex 3BC. This should normally reflect the exact value that was last written to hex 3BC. If an external device should be driving data on these pins (in violation of usage groundrules) at the time of an input, this data will be ORed with the latch contents.

IBM Monochrome Display & Printer Adapter	Printer Adapter
Input from address hex 3BD	Input from address hex 379

This command presents realtime status to the processor from the pins as follows.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Pin 11	Pin 10	Pin 12	Pin 13	Pin 15	-	-	-

IBM Monochrome Display & Printer Adapter	Printer Adapter
Input from address hex 3BE	Input from address hex 37A

This instruction causes the data present on pins 1, 14, 16, 17, and the IRQ bit to be read by the processor. In the absence of external drive applied to these pins, data read by the processor will exactly match data last written to hex 3BE in the same bit positions. Note that data bits 0-2 are not included. If external drivers are dotted to these pins, that data will be ORed with data applied to the pins by the hex 3BE latch.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
			IRQ Enable	Pin 17	Pin 16	Pin 14	Pin 1
			Por=0	Por=1	Por=0	Por=1	Por=1

These pins assume the states shown after a reset from the processor.

1-80 Printer Adapter

IBM Monochrome Display and Printer Adapter

This adapter has two functions. The first is to provide the interface to the IBM Monochrome Display. The second provides a parallel interface for the IBM 80 CPS Printer. This second function is fully discussed in the "IBM Printer Adapter" section.

The monitor adapter is designed around the Motorola 6845 CRT controller module. There are 4K bytes of static memory on the adapter which is used for the display buffer. This buffer has two ports and may be accessed directly by the processor. No parity is provided on the display buffer.

Two bytes are fetched from the display buffer in 553 ns, providing a data rate of 1.8M bytes/second.

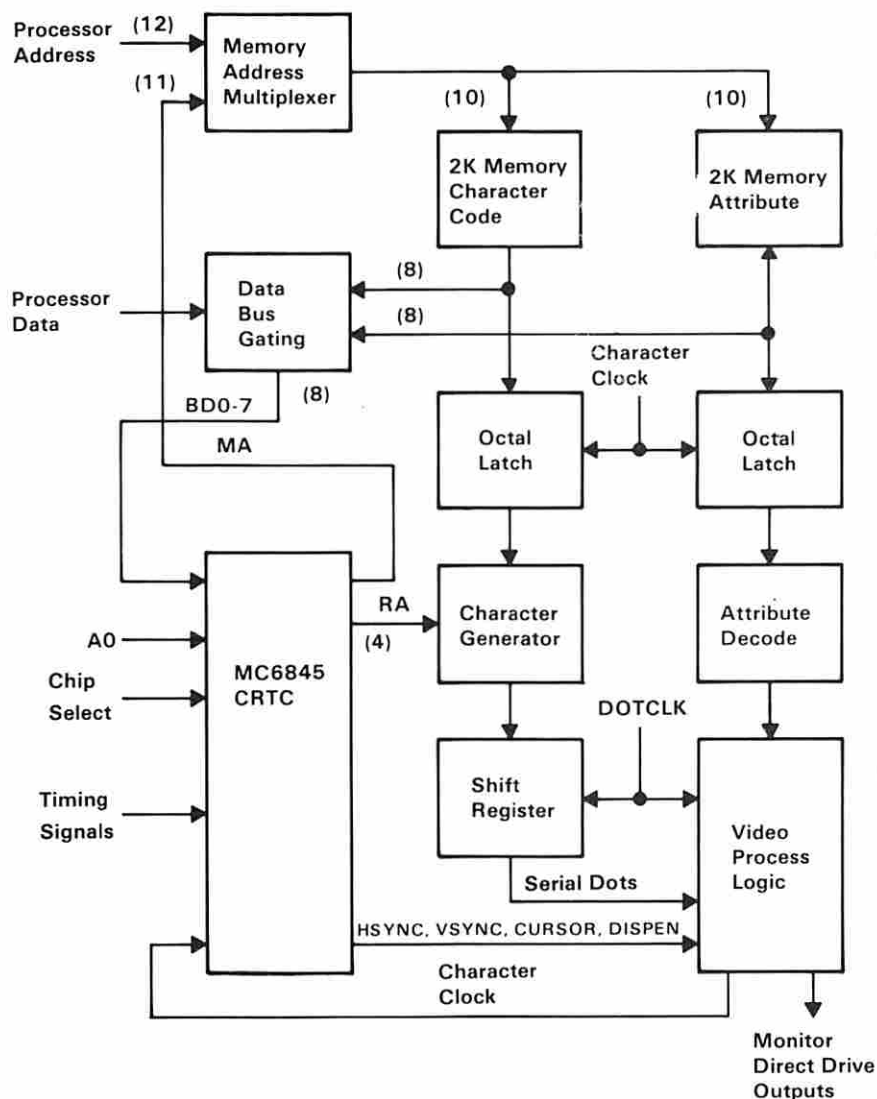
The monitor adapter supports 256 different character codes. An 8K-byte character generator contains the fonts for the character codes. The characters, values, and screen characteristics are given in "Appendix C: Of Characters, Keystrokes, and Color."

This monitor adapter, when used with a display containing P39 phosphor, will not support a light pen.

Where possible, only one low-power Schottky (LS) load is present on any I/O slot. Some of the address bus lines have two LS loads. No signal has more than two LS loads.

Characteristics of the monitor adapter are listed below:

- 80 x 25 screen
- Direct-drive output
- 9 x 14 character box
- 7 x 9 character
- 18 kHz monitor
- Character attributes



IBM Monochrome Display Adapter Block Diagram

Programming Considerations

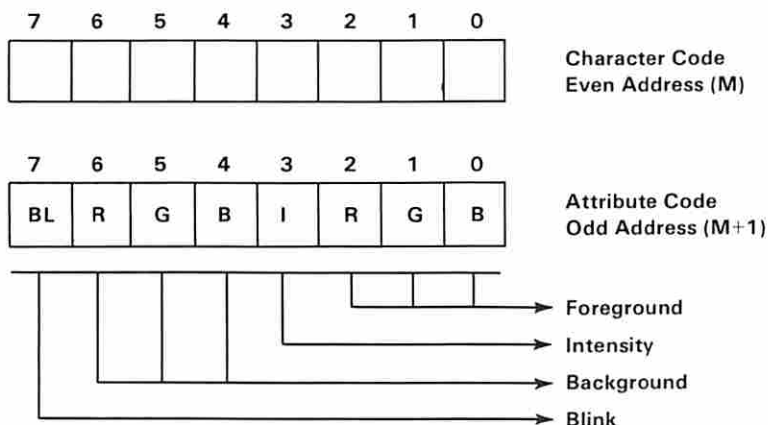
The following table summarizes the 6845 internal data registers, their functions, and their parameters. For the IBM Monochrome Display, the values must be programmed into the 6845 to ensure proper initialization of the device.

Register Number	Register File	Program Unit	IBM Monochrome Display (Address in hex)
R0	Horizontal Total	Characters	61
R1	Horizontal Displayed	Characters	50
R2	Horizontal Sync Position	Characters	52
R3	Horizontal Sync Width	Characters	F
R4	Vertical Total	Character Rows	19
R5	Vertical Total Adjust	Scan Line	6
R6	Vertical Displayed	Character Row	19
R7	Vertical Sync Position	Character Row	19
R8	Interlace Mode	-----	02
R9	Maximum Scan Line Address	Scan Line	D
R10	Cursor Start	Scan Line	B
R11	Cursor End	Scan Line	C
R12	Start Address (H)	-----	00
R13	Start Address (L)	-----	00
R14	Cursor (H)	-----	00
R15	Cursor (L)	-----	00
R16	Reserved	-----	--
R17	Reserved	-----	--

To ensure proper initialization, the first command issued to the attachment must be to send to CRT control port 1 (hex 3B8), a hex 01, to set the high-resolution mode. If this bit is not set, then the processor access to the monochrome adapter must never occur. If the high-resolution bit is not set, the processor will stop running.

System configurations that have both an IBM Monochrome Display Adapter and Printer Adapter, and an IBM Color/Graphics Monitor Adapter, must ensure that both adapters are properly initialized after a power-on reset. Damage to either display may occur if not properly initialized.

The IBM Monochrome Display and Printer Adapter supports 256 different character codes. In the character set are alphanumerics and block graphics. Each character in the display buffer has a corresponding character attribute. The character code must be an even address, and the attribute code must be an odd address in the display buffer.



The adapter decodes the character attribute byte as defined above. The blink and intensity bits may be combined with the foreground and background bits to further enhance the character attribute functions listed below.

Background R G B	Foreground R G B	Function
0 0 0	0 0 0	Non-Display
0 0 0	0 0 1	Underline
0 0 0	1 1 1	White Character/Black Background
1 1 1	0 0 0	Reverse Video

The 4K display buffer supports one screen of 25 rows of 80 characters, plus a character attribute for each display character. The starting address of the buffer is hex B0000. The display buffer can be read from using DMA; however, at least one wait-state will be inserted by the processor. The duration of the wait-state will vary, because the processor/monitor access is synchronized with the character clock on this adapter.

Interrupt level 7 is used on the parallel interface. Interrupts can be enabled or disabled through the printer control port. The interrupt is a high-level active signal.

The figure below breaks down the functions of the I/O address decode for the adapter. The I/O address decode is from hex 3B0 through hex 3BF. The bit assignment for each I/O address follows:

I/O Register Address	Function
3B0	Not Used
3B1	Not Used
3B2	Not Used
3B3	Not Used
3B4*	6845 Index Register
3B5*	6845 Data Register
3B6	Not Used
3B7	Not Used
3B8	CRT Control Port 1
3B9	Reserved
3BA	CRT Status Port
3BB	Reserved
3BC	Parallel Data Port
3BD	Printer Status Port
3BE	Printer Control Port
3BF	Not Used

*The 6845 Index and Data Registers are used to program the CRT controller to interface the high-resolution IBM Monochrome Display.

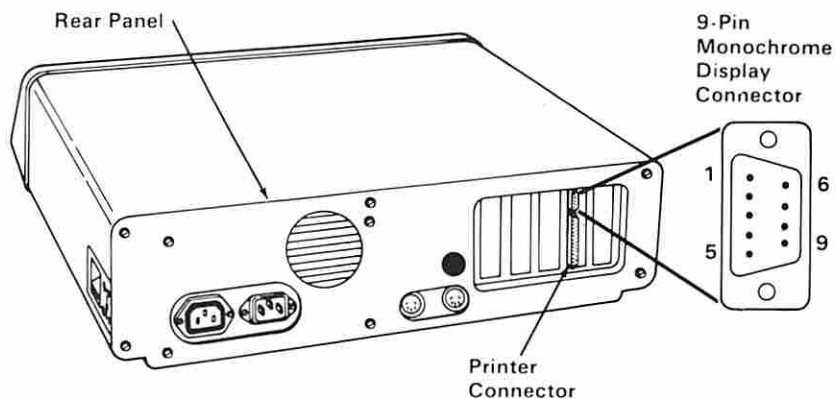
I/O Address and Bit Map

Bit Number	Function
0	+High Resolution Mode
1	Not Used
2	Not Used
3	+Video Enable
4	Not Used
5	+Enable Blink
6,7	Not Used

6845 CRT Control Port 1 (Hex 3B8)

Bit Number	Function
0	+Horizontal Drive
1	Reserved
2	Reserved
3	+Black/White Video

6845 CRT Status Port (Hex 3BA)



At Standard TTL Levels		
IBM Monochrome Display	Ground	1
	Ground	2
	Not Used	3
	Not Used	4
	Not Used	5
	+Intensity	6
	+Video	7
	+Horizontal	8
	- Vertical	9
		IBM Monochrome Display and Printer Adapter

Note: Signal voltages are 0.0 to 0.6 Vdc at down level and +2.4 to 3.5 Vdc at high level.

Connector Specifications

Notes:

IBM Monochrome Display

The high-resolution IBM Monochrome Display attaches to the system unit through two cables approximately 3 feet (914 millimeters) in length. One cable is a signal cable that contains the direct drive interface from the IBM Monochrome Display and Printer Adapter.

The second cable provides ac power to the display from the system unit. This allows the system-unit power switch to also control the display unit. An additional benefit is a reduction in the requirements for wall outlets to power the system. The display contains an 11-1/2 inch (283 millimeters), diagonal 90° deflection CRT. The CRT and analog circuits are packaged in an enclosure so the display may either sit on top of the system unit or on a nearby tabletop or desk. The unit has both brightness and contrast adjustment controls on the front surface that are easily accessible to the operator.

Operating Characteristics

Screen

- High-persistence green phosphor (P 39).
- Etched surface to reduce glare.
- Size is 80 characters by 25 lines.
- Character box is 9 dots wide by 14 dots high.

Video Signal

- Maximum bandwidth of 16.257 MHz.

Vertical Drive

- Screen refreshed at 50 Hz with 350 lines of vertical resolution and 720 lines of horizontal resolution.

Horizontal Drive

- Positive-level, TTL-compatibility, at a frequency of 18.432 kHz.

Notes:

IBM Color/Graphics Monitor Adapter

The IBM Color/Graphics Monitor Adapter is designed to attach to the IBM Color Display, to a variety of NTSC standard (U.S. National Television System Committee) television-frequency monitors, or to NTSC-standard home television sets (user-supplied RF modulator is required for home television sets). The adapter is capable of operating in black-and-white or color. It provides three video interfaces: a composite-video port, a direct-drive port, and a connection interface for driving a user-supplied RF modulator. In addition, a light pen interface is provided.

The adapter has two basic modes of operation: alphanumeric (A/N) and all-points-addressable graphics (APA). Additional modes are available within the A/N and APA modes. In the A/N mode, the display can be operated in either a 40-column by 25-row mode for a low-resolution monitor or home television, or in an 80-column by 25-row mode for high-resolution monitors. In both modes, characters are defined in an 8-wide by 8-high character box and are 7-wide by 7-high, with one line of descender for lowercase characters. Both uppercase and lowercase characters are supported in all modes.

The character attributes of reverse video, blinking, and highlighting are available in the black-and-white mode. In the color mode, sixteen foreground and eight background colors are available for each character. In addition, blinking on a per-character basis is available.

The monitor adapter contains 16K bytes of storage. As an example, a 40-column by 25-row display screen uses 1000 bytes to store character information, and 1000 bytes to store attribute/color information. This would mean that up to eight display screens can be stored in the adapter memory. Similarly, in an 80-column by 25-row mode, four display screens may be stored in the adapter. The entire 16K bytes of storage on the display adapter are directly addressable by the processor, which allows maximum software flexibility in managing the screen.

In A/N color modes, it is also possible to select the color of the screen's border. One of sixteen colors can be selected.

In the APA mode, there are two resolutions available: a medium-resolution color graphics mode (320 PELs by 200 rows) and a high-resolution black-and-white graphics mode (640 PELs by 200 rows). In the medium-resolution mode, each picture element (PEL) may have one of four colors. The background color (color 0) may be any of the 16 possible colors. The remaining three colors come from one of the two software-selectable palettes. One palette contains green/red/brown; the other contains cyan/magenta/white.

The high-resolution mode is available only in black-and-white because the entire 16K bytes of storage in the adapter is used to define the on or off state of the PELs.

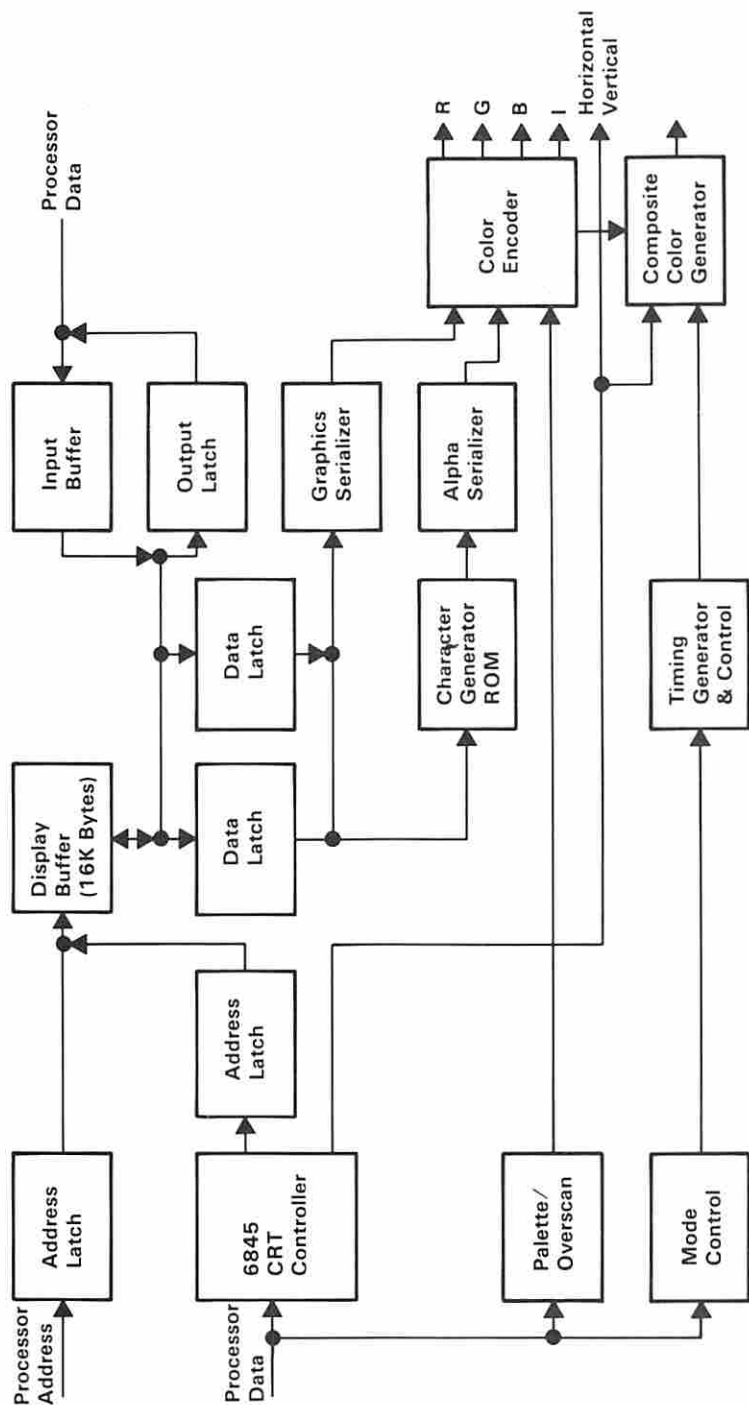
The adapter operates in noninterlace mode at either 7 or 14 MHz, depending on the mode of operation selected.

In the A/N mode, characters are formed from a ROM character generator. The character generator contains dot patterns for 256 different characters. The character set contains the following major groupings of characters:

- 16 special characters for game support
- 15 characters for word-processing editing support
- 96 characters for the standard ASCII graphics set
- 48 characters for foreign-language support
- 48 characters for business block-graphics support (allowing drawing of charts, boxes, and tables using single and double lines)
- 16 selected Greek characters
- 15 selected scientific-notation characters

The color/graphics monitor adapter function is packaged on a single card. The direct-drive and composite-video ports are right-angle mounted connectors on the adapter, and extend through the rear panel of the unit. The direct-drive video port is a 9-pin D-shell female connector. The composite-video port is a standard female phono-jack.

The display adapter is implemented using a Motorola 6845 CRT controller device. This adapter is highly programmable with respect to raster and character parameters. Therefore, many additional modes are possible with clever programming of the adapter.



Color/Graphics Monitor Adapter Block Diagram

Descriptions of Major Components

Motorola 6845 CRT Controller

This device provides the necessary interface to drive a raster-scan CRT.

Mode Set Register

This is a general-purpose, programmable, I/O register. It has I/O ports that may be individually programmed. Its function in this attachment is to provide mode selection and color selection in the medium-resolution color-graphics mode.

Display Buffer

The display buffer resides in the processor-address space, starting at address hex B8000. It provides 16K bytes of dynamic read/write memory. A dual-ported implementation allows the processor and the graphics control unit to access this buffer. The processor and the CRT control unit have equal access to this buffer during all modes of operation, except in the high-resolution alphanumeric mode. In this mode, only the processor should access this buffer during the horizontal-retrace intervals. While the processor may write to the required buffer at any time, a small amount of display interference will result if this does not occur during the horizontal-retrace intervals.

Character Generator

This attachment utilizes a ROM character generator. It consists of 8K bytes of storage that cannot be read from or written to under software control. This is a general-purpose ROM character generator with three different character fonts. Two character fonts are used on the color/graphics adapter: a 7-high by 7-wide double-dot font and a 5-wide by 7-high single-dot font. The font is selected by a jumper (P3). The single-dot font is selected by inserting the jumper; the double-dot font is selected by removing the jumper.

Timing Generator

This generator produces the timing signals used by the 6845 CRT controller and by the dynamic memory. It also resolves the processor/graphic controller contentions for accessing the display buffer.

Composite Color Generator

This generator produces base band video color information.

Alphanumeric Mode

Every display-character position in the alphanumeric mode is defined by two bytes in the regen buffer (a part of the monitor adapter), not the system memory. Both the color/graphics and the monochrome display adapter use the following 2-byte character/attribute format.

Display-Character Code Byte								Attribute Byte							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0

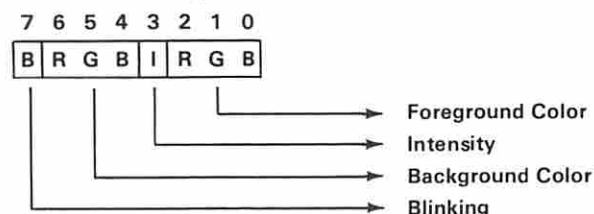
The functions of the attribute byte are defined by the following table:

Attribute Function	Attribute Byte							
	7	6	5	4	3	2	1	0
	B	R	G	B	I	R	G	B
	FG	Background			Foreground			
Normal	B	0	0	0	I	1	1	1
Reverse Video	B	1	1	1	I	0	0	0
Nondisplay (Black)	B	0	0	0	I	0	0	0
Nondisplay (White)	B	1	1	1	I	1	1	1

I = Highlighted Foreground (Character)

B = Blinking Foreground (Character)

The attribute byte definitions are:



In the alphanumeric mode, the display can be operated in either a low-resolution mode or a high-resolution mode.

The low-resolution alphanumeric mode has the following features:

- Supports home color televisions or low-resolution monitors
- Displays up to 25 rows of 40 characters each
- ROM character generator that contains dot patterns for a maximum of 256 different characters
- Requires 2,000 bytes of read/write memory (on the adapter)
- Character box is 8-high by 8-wide
- Two jumper-controlled character fonts are available:
 - 5-wide by 7-high single-dot character font with one descender
 - 7-wide by 7-high double-dot character font with one descender
- One character attribute for each character

The high-resolution alphanumeric mode has the following features:

- Supports the IBM Color Display or other color monitor with direct-drive input capability
- Supports a black-and-white composite-video monitor
- Displays up to 25 rows of 80 characters each
- ROM displays generator that contains dot patterns for a maximum of 256 different characters
- Requires 4,000 bytes of read/write memory (on the adapter)
- Character box is 8-high by 8-wide
- Two jumper-controlled character fonts are available:
 - 5-wide by 7-high single-dot character font with one descender
 - 7-wide by 7-high double-dot character font with one descender
- One character attribute for each character

Monochrome vs Color/Graphics Character Attributes

Foreground and background colors are defined by the attribute byte of each character, whether using the IBM Monochrome Display and Printer Adapter or the IBM Color/Graphics Monitor Adapter. The following table describes the colors for each adapter:

Attribute Byte								Monochrome Display Adapter		Color/Graphics Monitor Adapter	
7	6	5	4	3	2	1	0				
B	R	G	B	I	R	G	B	Background Color	Character Color	Background Color	Character Color
FG	Background				Foreground						
B	0	0	0	I	1	1	1	Black	White	Black	White
B	1	1	1	I	0	0	0	White	Black	White	Black
B	0	0	0	I	0	0	0	Black	Black	Black	Black
B	1	1	1	I	1	1	1	White	White	White	White

The monochrome display adapter will produce white characters on a white background with any other code. The color/graphics adapter will change foreground and background colors according to the color value selected. The color values for the various red, green, blue, and intensity bit settings are given in the table below.

R	G	B	I	Color
0	0	0	0	Black
0	0	1	0	Blue
0	1	0	0	Green
0	1	1	0	Cyan
1	0	0	0	Red
1	0	1	0	Magenta
1	1	0	0	Brown
1	1	1	0	White
0	0	0	1	Gray
0	0	1	1	Light Blue
0	1	0	1	Light Green
0	1	1	1	Light Cyan
1	0	0	1	Light Red
1	0	1	1	Light Magenta
1	1	0	1	Yellow
1	1	1	1	White (High Intensity)

Code written with an underline attribute for the IBM Monochrome Display, when executed on a color/graphics monitor adapter, will result in a blue character where the underline attribute is encountered. Also, code written on a color/graphics monitor adapter with blue characters will be displayed as white characters on a black background, with a white underline on the IBM Monochrome Display.

Remember that not all monitors recognize the intensity (I) bit.

Graphics Mode

The IBM Color/Graphics Monitor Adapter has three modes available within the graphics mode. They are low-resolution color graphics, medium-resolution color graphics, and high-resolution color graphics. However, only medium- and high-resolution graphics are supported in ROM. The following table summarizes the three modes.

Mode	Horizontal (PELs)	Vertical (Rows)	Number of Colors Available (Includes Background Color)
Low Resolution	160	100	16 (Includes black-and-white)
Medium Resolution	320	200	4 Colors Total 1 of 16 for Background and 1 of Green, Red, or Brown or 1 of Cyan, Magenta, or White
High Resolution	640	200	Black-and-white only

Low-Resolution Color-Graphics Mode

The low-resolution mode supports home television or color monitors. This mode is not supported in ROM. It has the following features:

- Contains a maximum of 100 rows of 160 PELs, with each PEL being 2-high by 2-wide
- Specifies 1 of 16 colors for each PEL by the I, R, G, and B bits
- Requires 16,000 bytes of read/write memory (on the adapter)
- Uses memory-mapped graphics

Medium-Resolution Color-Graphics Mode

The medium-resolution mode supports home televisions or color monitors. It has the following features:

- Contains a maximum of 200 rows of 320 PELs, with each PEL being 1-high by 1-wide
- Preselects one of four colors for each PEL
- Requires 16,000 bytes of read/write memory (on the adapter)
- Uses memory-mapped graphics
- Formats 4 PELs per byte in the following manner:

7	6	5	4	3	2	1	0
C1	C0	C1	C0	C1	C0	C1	C0
First Display PEL		Second Display PEL		Third Display PEL		Fourth Display PEL	

- Organizes graphics storage in two banks of 8,000 bytes, using the following format:

Memory Address (in hex)	Function
B8000	Even Scans (0,2,4,...198) 8,000 bytes
B9F3F	Not Used
BA000	Odd Scans (1,3,5...199) 8,000 Bytes
BBF3F	Not Used
BBFFF	

Address hex B8000 contains PEL instruction for the upper-left corner of the display area.

- Color selection is determined by the following logic:

C1	C0	Function
0	0	Dot takes on the color of 1 of 16 preselected background colors
0	1	Selects first color of preselected Color Set 1 or Color Set 2
1	0	Selects second color of preselected Color Set 1 or Color Set 2
1	1	Selects third color of preselected Color Set 1 or Color Set 2

C1 and C0 will select 4 of 16 preselected colors. This color selection (palette) is preloaded in an I/O port.

The two color sets are:

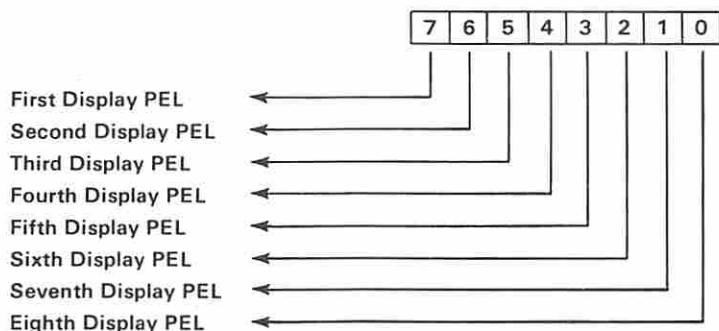
Color Set 1	Color Set 2
Color 1 is Green	Color 1 is Cyan
Color 2 is Red	Color 2 is Magenta
Color 3 is Brown	Color 3 is White

The background colors are the same basic 8 colors as defined for low-resolution graphics, plus 8 alternate intensities defined by the intensity bit, for a total of 16 colors, including black and white.

High-Resolution Black-and-White Graphics Mode

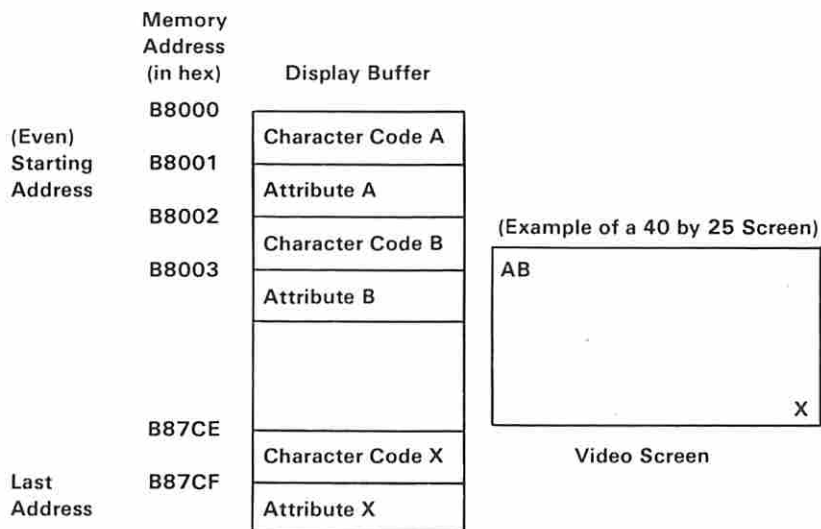
The high-resolution mode supports color monitors. This mode has the following features:

- Contains a maximum of 200 rows of 640 PELs, with each PEL being 1-high by 1-wide.
- Supports black-and-white mode only.
- Requires 16,000 bytes of read/write memory (on the adapter).
- Addressing and mapping procedures are the same as medium-resolution color graphics, but the data format is different. In this mode, each bit in memory is mapped to a PEL on the screen.
- Formats 8 PELs per byte in the following manner:



Description of Basic Operations

In the alphanumeric mode, the adapter fetches character and attribute information from its display buffer. The starting address of the display buffer is programmable through the 6845, but it must be an even address. The character codes and attributes are then displayed according to their relative positions in the buffer.



The processor and the display control unit have equal access to the display buffer during all the operating modes, except the high-resolution alphanumeric mode. During this mode, the processor should access the display buffer during the vertical retrace time. If it does not, the display will be affected with random patterns as the processor is using the display buffer. In the alphanumeric mode, the characters are displayed from a prestored ROM character generator that contains the dot patterns of all the displayable characters.

In the graphics mode, the displayed dots and colors (up to 16K bytes) are also fetched from the display buffer. The bit configuration for each graphics mode is explained in "Graphics Mode."

I	R	G	B	Color
0	0	0	0	Black
0	0	0	1	Blue
0	0	1	0	Green
0	0	1	1	Cyan
0	1	0	0	Red
0	1	0	1	Magenta
0	1	1	0	Brown
0	1	1	1	White
1	0	0	0	Gray
1	0	0	1	Light Blue
1	0	1	0	Light Green
1	0	1	1	Light Cyan
1	1	0	0	Light Red
1	1	0	1	Light Magenta
1	1	1	0	Yellow
1	1	1	1	High Intensity White

Note: "I" provides extra luminance (brightness) to each available shade. This results in the light colors listed above, except for monitors that do not recognize the "I" bit.

Summary of Available Colors

Programming Considerations

Programming the 6845 CRT Controller

The 6845 has 19 accessible internal registers, which are used to define and control a raster-scan CRT display. One of these registers, the Index register, is actually used as a pointer to the other 18 registers. It is a write-only register, which is loaded from the processor by executing an 'out' instruction to I/O address hex 3D4. The five least significant bits of the I/O bus are loaded into the Index register.

In order to load any of the other 18 registers, the Index register is first loaded with the necessary pointer; then the Data Register is loaded with the information to be placed in the selected register. The Data Register is loaded from the processor by executing an Out instruction to I/O address hex 3D5.

The following table defines the values that must be loaded into the 6845 CRT Controller registers to control the different modes of operation supported by the attachment:

Address Register	Register Number	Register Type	Units	I/O	40 by 25 Alpha-numeric	80 by 25 Alpha-numeric	Graphic Modes
0	R0	Horizontal Total	Character	Write Only	38	71	38
1	R1	Horizontal Displayed	Character	Write Only	28	50	28
2	R2	Horizontal Sync Position	Character	Write Only	2D	5A	2D
3	R3	Horizontal Sync Width	Character	Write Only	0A	0A	0A
4	R4	Vertical Total	Character Row	Write Only	1F	1F	7F
5	R5	Vertical Total Adjust	Scan Line	Write Only	06	06	06
6	R6	Vertical Displayed	Character Row	Write Only	19	19	64
7	R7	Vertical Sync Position	Character Row	Write Only	1C	1C	70
8	R8	Interlace Mode	-	Write Only	02	02	02
9	R9	Maximum Scan Line Address	Scan Line	Write Only	07	07	01
A	R10	Cursor Start	Scan Line	Write Only	06	06	06
B	R11	Cursor End	Scan Line	Write Only	07	07	07
C	R12	Start Address (H)	-	Write Only	00	00	00
D	R13	Start Address (L)	-	Write Only	00	00	00
E	R14	Cursor Address (H)	-	Read/Write	XX	XX	XX
F	R15	Cursor Address (L)	-	Read/Write	XX	XX	XX
10	R16	Light Pen (H)	-	Read Only	XX	XX	XX
11	R17	Light Pen (L)	-	Read Only	XX	XX	XX

Note: All register values are given in hexadecimal

6845 Register Description

1-106 Color/Graphics Adapter

Programming the Mode Control and Status Register

The following I/O devices are defined on the color/graphics adapter.

Hex Address	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Function of Register
3D8	1	1	1	1	0	1	1	0	0	0	Mode Control Register (D0)
3D9	1	1	1	1	0	1	1	0	0	1	Color Select Register (D0)
3DA	1	1	1	1	0	1	1	0	1	0	Status Register (D1)
3DB	1	1	1	1	0	1	1	0	1	1	Clear Light Pen Latch
3DC	1	1	1	1	0	1	1	1	0	0	Preset Light Pen Latch
3D4	1	1	1	1	0	1	0	Z	Z	0	6845 Index Register
3D5	1	1	1	1	0	1	0	Z	Z	1	6845 Data Register
3D0	1	1	1	1	0	1	0	Z	Z	0	6845 Registers
3D1	1	1	1	1	0	1	0	Z	Z	1	6845 Registers

Z = don't care condition

Color-Select Register

This is a 6-bit output-only register (cannot be read). Its I/O address is hex 3D9, and it can be written to by using the 8088 I/O Out command.

Bit 0	Selects B (Blue) Border Color in 40 x 25 Alphanumeric Mode Selects B (Blue) Background Color in 320 x 200 Graphics Mode Selects B (Blue) Foreground Color in 640 x 200 Graphics Mode
Bit 1	Selects G (Green) Border Color in 40 x 25 Alphanumeric Mode Selects G (Green) Background Color in 320 x 200 Graphics Mode Selects G (Green) Foreground Color in 640 x 200 Graphics Mode
Bit 2	Selects R (Red) Border Color in 40 x 25 Alphanumeric Mode Selects R (Red) Background Color in 320 x 200 Graphics Mode Selects R (Red) Foreground Color in 640 x 200 Graphics Mode
Bit 3	Selects I (Intensified) Border Color in 40 x 25 Alphanumeric Mode Selects I (Intensified) Background Color in 320 x 200 Graphics Mode Selects I (Intensified) Foreground Color in 640 x 200 Graphics Mode
Bit 4	Selects Alternate, Intensified Set of Colors in Graphics Mode Selects Background Colors in the Alphanumeric Mode
Bit 5	Selects Active Color Set in 320 x 200 Graphics Mode
Bit 6	Not Used
Bit 7	Not Used

- Bits 0, 1, 2, 3 These bits select the screen's border color in the 40 x 25 alphanumeric mode. They select the screen's background color (C0-C1) in the medium-resolution (320 by 200) color-graphics mode.
- Bits 4 This bit, when set, will select an alternate, intensified set of colors. Selects background colors in the alphanumeric mode.
- Bit 5 This bit is only used in the medium-resolution (320 by 200) color-graphics mode. It is used to select the active set of screen colors for the display.

When bit 5 is set to 1, colors are determined as follows:

C1	C0	Set Selected
0	0	Background (Defined by bits 0-3 of port hex 3D9)
0	1	Cyan
1	0	Magenta
1	1	White

When bit 5 is set to 0, colors are determined as follows:

C1	C0	Set Selected
0	0	Background (Defined by bits 0-3 of port hex 3D9)
0	1	Green
1	0	Red
1	1	Brown

Mode-Select Register

This is a 6-bit output-only register (cannot be read). Its I/O address is hex 3D8, and it can be written to using the 8088 I/O Out command.

The following is a description of the register's functions:

Bit 0	80 x 25 Alphanumeric Mode
Bit 1	Graphics Select
Bit 2	Black/White Select
Bit 3	Enable Video Signal
Bit 4	High-Resolution (640 x 200) Black/White Mode
Bit 5	Change Background Intensity to Blink Bit
Bit 6	Not Used
Bit 7	Not Used

- Bit 0 A 1 selects 80 by 25 alphanumeric mode
 A 0 selects 40 by 25 alphanumeric mode
- Bit 1 A 1 selects 320 by 200 graphics mode
 A 0 selects alphanumeric mode
- Bit 2 A 1 selects black-and-white mode
 A 0 selects color mode
- Bit 3 A 1 enables the video signal at certain times when modes are being changed. The video signal should be disabled when changing modes.
- Bit 4 A 1 selects the high-resolution (640 x 200) black-and-white graphics mode. One color of 8 can be selected on direct-drive sets in this mode by using register hex 3D9.
- Bit 5 When on, this bit will change the character background intensity to the blinking attribute function for alphanumeric modes. When the high-order attribute bit is not selected, 16 background colors (or intensified colors) are available. For normal operation, this bit should be set to 1 to allow the blinking function.

Mode Register Summary

Bits

0	1	2	3	4	5
0	0	1	1	0	1
0	0	0	1	0	1
1	0	1	1	0	1
1	0	0	1	0	1
0	1	1	1	0	z
0	1	0	1	0	z
0	1	1	1	1	z

40 x 25 Alphanumeric Black-and-White

40 x 25 Alphanumeric Color

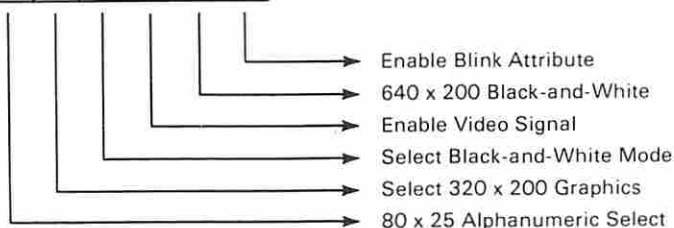
80 x 25 Alphanumeric Black-and-White

80 x 25 Alphanumeric Color

320 x 200 Black-and-White Graphics

320 x 200 Color Graphics

640 x 200 Black-and-White Graphics



z = don't care condition

Note: The low-resolution (160 by 100) mode requires special programming and is set up as the 40 by 25 alphanumeric mode.

Status Register

The status register is a 4-bit read-only register. Its I/O address is hex 3DA, and it can be read using the 8088 I/O In instruction. The following is a description of the register functions:

Bit 0	Display Enable
Bit 1	Light-Pen Trigger Set
Bit 2	Light-Pen Switch Made
Bit 3	Vertical Sync
Bit 4	Not Used
Bit 5	Not Used
Bit 6	Not Used
Bit 7	Not Used

- Bit 0 This bit, when active, indicates that a regen buffer memory access can be made without interfering with the display.
- Bit 1 This bit, when active, indicates that a positive-going edge from the light-pen has set the light pen's trigger. This trigger is reset upon power-on and may also be cleared by performing an I/O Out command to hex address 3DB. No specific data setting is required; the action is address-activated.
- Bit 2 The light-pen switch status is reflected in this status bit. The switch is not latched or debounced. A 0 indicates that the switch is on.
- Bit 3 This bit, when active, indicates that the raster is in a vertical retrace mode. This is a good time to perform screen-buffer updating.

Sequence of Events for Changing Modes

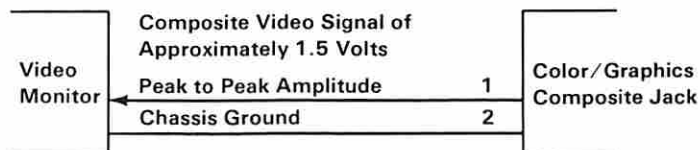
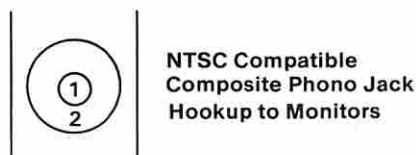
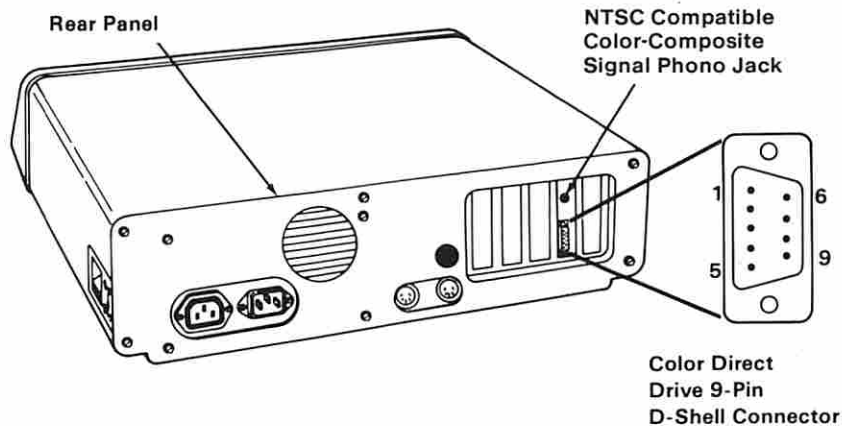
1. Determine the mode of operation.
2. Reset 'video enable' bit in mode-select register.
3. Program 6845 to select mode.
4. Program mode/color select registers including re-enabling video.

Memory Requirements

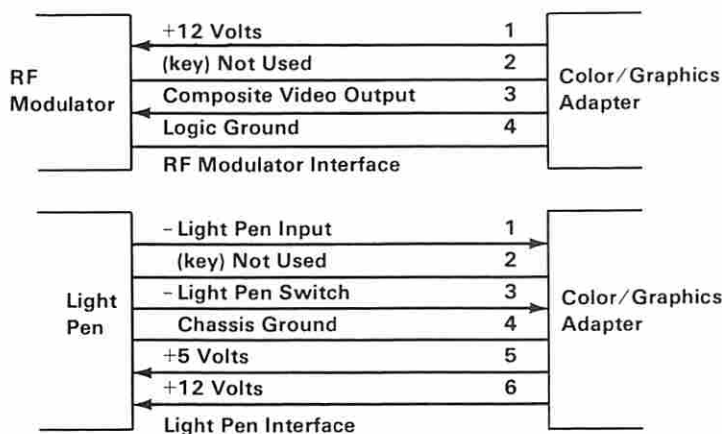
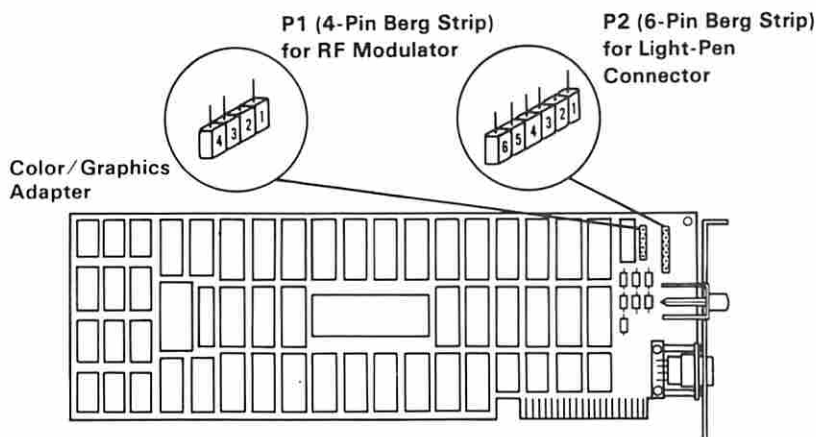
The memory used by this adapter is self-contained. It consists of 16K bytes of memory without parity. This memory is used as both a display buffer for alphanumeric data and as a bit map for graphics data. The regen buffer's address starts at hex B8000.

Read/Write Memory
Address Space (in hex)

System Read/Write Memory	01000	
	A0000	} 128K Reserved Regen Area
Display Buffer (16K Bytes)	B8000	
	BC000	
	C0000	



Connector Specifications (Part 1 of 2)



Connector Specifications (Part 2 of 2)

IBM Color Display

The IBM Color Display attaches to the system unit by a signal cable that is approximately 5 feet (1.5 meters) in length. This signal cable provides a direct-drive interface from the IBM Color/Graphics Monitor Adapter.

A second cable provides ac power to the display from a standard wall outlet. The display has its own power control and indicator. The display will accept either 120-volt 60-Hz, or 220-volt 50-Hz power. The power supply in the display automatically switches to match the applied power.

The display has a 13-inch (340-millimeters) CRT. The CRT and analog circuits are packaged in an enclosure so the display may sit either on top of the system unit or on a nearby tabletop or desk. Front panel controls and indicators include: Power-On control, Power-On indicator, Brightness and Contrast controls. Two additional rear-panel controls are the Vertical Hold and Vertical Size controls.

Operating Characteristics

Screen

- High contrast (black) screen.
- Displays up to 16 colors, when used with the IBM Color/Graphics Monitor Adapter.
- Characters defined in an 8-high by 8-wide matrix.

Video Signal

- Maximum video bandwidth of 14 MHz.
- Red, green, and blue video signals and intensity are all independent.

Vertical Drive

- Screen refreshed at 60 Hz with 200 vertical lines of resolution.

Horizontal Drive

- Positive-level, TTL compatibility, at a frequency of 15.75 kHz.

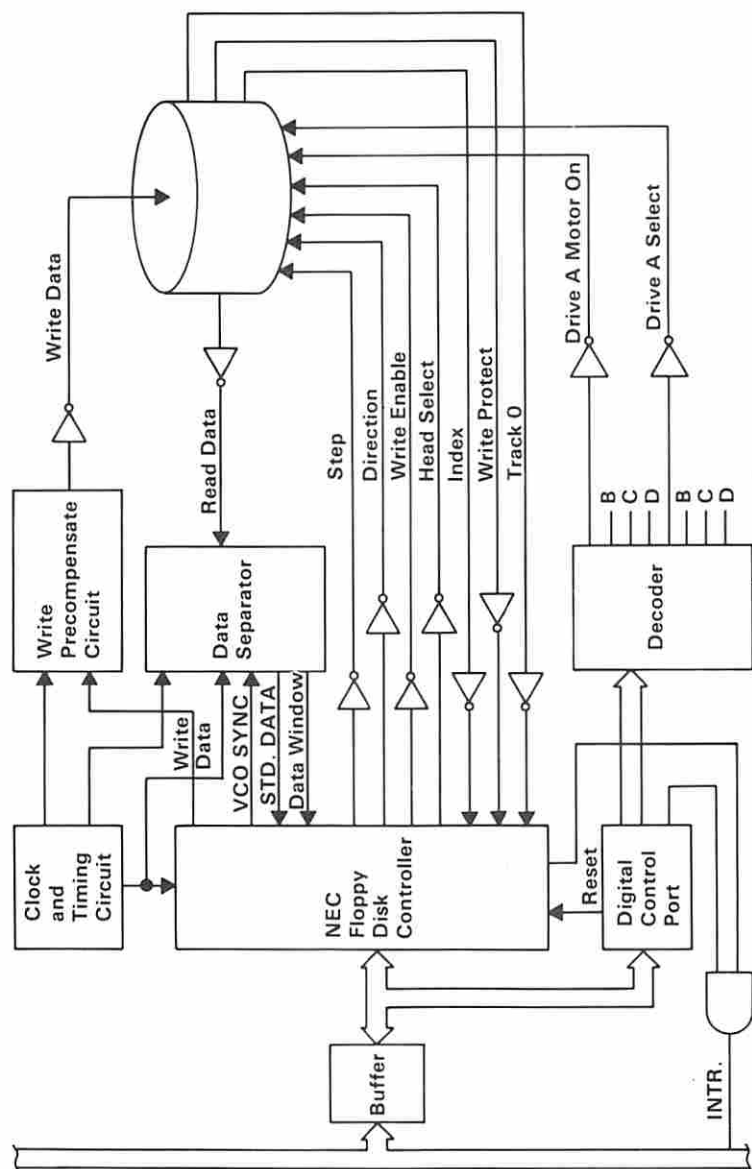
Notes:

IBM 5-1/4" Diskette Drive Adapter

The 5-1/4 inch diskette drive adapter fits into one of the expansion slots in the system unit. It attaches to one or two diskette drives through an internal, daisy-chained flat cable that connects to one end of the drive adapter. The adapter has a connector at the other end that extends through the rear panel of the system unit. This connector has signals for two additional external diskette drives; thus, the 5-1/4 inch diskette drive adapter can attach four 5-1/4 inch drives - two internal and two external.

The adapter is designed for double-density, MFM-coded, diskette drives and uses write precompensation with an analog phase-lock loop for clock and data recovery. The adapter is a general-purpose device using the NEC μ PD765 compatible controller. Therefore, the diskette drive parameters are programmable. In addition, the attachment supports the diskette drive's write-protect feature. The adapter is buffered on the I/O bus and uses the system board's direct memory access (DMA) for record data transfers. An interrupt level is also used to indicate when an operation is complete and that a status condition requires processor attention.

In general, the 5-1/4 inch diskette drive adapter presents a high-level command interface to software I/O drivers. A block diagram of the 5-1/4 inch diskette drive adapter is on the following page.



5-1/4 Inch Diskette Drive Adapter Block Diagram

Functional Description

From a programming point of view, this attachment consists of an 8-bit digital-output register in parallel with an NEC μ PD765 or equivalent floppy disk controller (FDC).

In the following description, drive numbers 0, 1, 2, and 3 are equivalent to drives A, B, C, and D.

Digital-Output Register

The digital-output register (DOR) is an output-only register used to control drive motors, drive selection, and feature enable. All bits are cleared by the I/O interface reset line. The bits have the following functions:

Bits 0 and 1 These bits are decoded by the hardware to select one drive if its motor is on:

Bit	1	0	Drive
	0	0	0 (A)
	0	1	1 (B)
	1	0	2 (C)
	1	1	3 (D)

Bit 2 The FDC is held reset when this bit is clear. It must be set by the program to enable the FDC.

Bit 3 This bit allows the FDC interrupt and DMA requests to be gated onto the I/O interface. If this bit is cleared, the interrupt and DMA request I/O interface drivers are disabled.

Bits 4, 5, 6, and 7 These bits control, respectively, the motors of drives 0, 1, 2, (A, B, C), and 3 (D). If a bit is clear, the associated motor is off, and the drive cannot be selected.

Floppy Disk Controller

The floppy disk controller (FDC) contains two registers that may be accessed by the main system processor: a status register and a data register. The 8-bit main status register contains the status information of the FDC and may be accessed at any time. The 8-bit data register (actually consisting of several registers in a stack with only one register presented to the data bus at a time) stores data, commands, parameters, and provides floppy disk drive (FDD) status information. Data bytes are read from or written to the data register in order to program or obtain results after a particular command. The main status register may only be read and is used to facilitate the transfer of data between the processor and FDC.

The bits in the main status register (hex 34F) are defined as follows:

Bit Number	Name	Symbol	Description
DB0	FDD A Busy	DAB	FDD number 0 is in the Seek mode.
DB1	FDD B Busy	DBB	FDD number 1 is in the Seek mode.
DB2	FDD C Busy	DCB	FDD number 2 is in the Seek mode.
DB3	FDD D Busy	DDB	FDD number 3 is in the Seek mode.
DB4	FDC Busy	CB	A read or write command is in process.
DB5	Non-DMA Mode	NDM	The FDC is in the non-DMA mode.
DB6	Data Input/Output	DIO	Indicates direction of data transfer between FDC and processor. If DIO = "1," then transfer is from FDC data register to the processor. If DIO = "0," then transfer is from the processor to FDC data register.
DB7	Request for Master	RQM	Indicates data register is ready to send or receive data to or from the processor. Both bits DIO and RQM should be used to perform the handshaking functions of "ready" and "direction" to the processor.

The FDC is capable of performing 15 different commands. Each command is initiated by a multi-byte transfer from the processor, and the result after execution of the command may also be a multi-byte transfer back to the processor. Because of this multi-byte interchange of information between the FDC and the processor, it is convenient to consider each command as consisting of three phases:

Command Phase

The FDC receives all information required to perform a particular operation from the processor.

Execution Phase

The FDC performs the operation it was instructed to do.

Result Phase

After completion of the operation, status and other housekeeping information is made available to the processor.

Programming Considerations

The following tables define the symbols used in the command summary, which follows.

Symbol	Name	Description
A0	Address Line 0	A0 controls selection of main status register (A0 = 0) or data register (A0 = 1).
C	Cylinder Number	C stands for the current/selected cylinder (track) number of the medium.
D	Data	D stands for the data pattern that is going to be written into a sector.
D7-D0	Data Bus	8-bit data bus, where D7 stands for a most significant bit, and D0 stands for a least significant bit.
DTL	Data Length	When N is defined as 00, DTL stands for the data length that users are going to read from or write to the sector.
EOT	End of Track	EOT stands for the final sector number on a cylinder.
GPL	Gap Length	GPL stands for the length of gap 3 (spacing between sectors excluding VCO sync field).
H	Head Address	H stands for head number 0 or 1, as specified in ID field.
HD	Head	HD stands for a selected head number 0 or 1. (H = HD in all command words.)
HLT	Head Load Time	HLT stands for the head load time in the FDD (4 to 512 ms in 4-ms increments).
HUT	Head Unload Time	HUT stands for the head unload time after a read or write operation has occurred (0 to 480 ms in 32-ms increments).
MF	FM or MFM Mode	If MF is low, FM mode is selected; if it is high, MFM mode is selected only if MFM is implemented.
MT	Multi-Track	If MT is high, a multi-track operation is to be performed. (A cylinder under both HD0 and HD1 will be read or written.)
N	Number	N stands for the number of data bytes written in a sector.

Symbol Descriptions (Part 1 of 2)

Symbol	Name	Description
NCN	New Cylinder Number	NCN stands for a new cylinder number, which is going to be reached as a result of the seek operation. (Desired position of the head.)
ND	Non-DMA Mode	ND stands for operation in the non-DMA mode.
PCN	Present Cylinder Number	PCN stands for cylinder number at the completion of sense-interrupt-status command indicating the position of the head at present time.
R	Record	R stands for the sector number, which will be read or written.
R/W	Read/Write	R/W stands for either read (R) or write (W) signal.
SC	Sector	SC indicates the number of sectors per cylinder.
SK	Skip	SK stands for skip deleted-data address mark.
SRT	Step Rate Time	SRT stands for the stepping rate for the FDD (2 to 32 ms in 2-ms increments).
ST 0 ST 1 ST 2 ST 3	Status 0 Status 1 Status 2 Status 3	ST 0-3 stand for one of four registers that store the status information after a command has been executed. This information is available during the result phase after command execution. These registers should not be confused with the main status register (selected by A0 = 0). ST 0-3 may be read only after a command has been executed and contain information relevant to that particular command.
STP	Scan Test	During a scan operation, if STP = 1, the data in contiguous sectors is compared byte-by-byte with data sent from the processor (or DMA), and if STP = 2, then alternate sectors are read and compared.
US0, US1	Unit Select	US stands for a selected drive number encoded the same as bits 0 and 1 of the digital output register (DOR).

Symbol Descriptions (Part 2 of 2)

Command Summary

In the following table, 0 indicates "logical 0" for that bit, 1 means "logical 1," and X means "don't care."

Phase	R/W	Data Bus								Remarks
		D7	D6	D5	D4	D3	D2	D1	D0	
Command	W	MT	MF	SK	0	0	1	1	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					
	W				H					
	W				R					
	W				N					
	W				EOT					
	W				GPL					
Execution	W				DTL					Data transfer between the FDD and main system.
Result	R				ST 0					Status information after command execution.
	R				ST 1					
	R				ST 2					
	R				C					
	R				H					
	R				R					
	R				N					
	R									
Command	W	MT	MF	SK	0	1	1	0	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					
	W				H					
	W				R					
	W				N					
	W				EOT					
	W				GPL					
Execution	W				DTL					Data transfer between the FDD and main system.
Result	R				ST 0					Status information after command execution.
	R				ST 1					
	R				ST 2					
	R				C					
	R				H					
	R				R					
	R				N					
	R									

Phase	R/W	Data Bus								Remarks
		D7	D6	D5	D4	D3	D2	D1	D0	
Command	W	MT	MF	0	0	0	1	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					
	W				H					
	W				R					
	W				N					
	W				EOT					
	W				GPL					
Execution	W				DTL					Data transfer between the main system and FDD.
Result	R				ST 0					Status information after command execution.
	R				ST 1					
	R				ST 2					
	R				C					
	R				H					
	R				R					
	R				N					
Command	W	MT	MF	0	0	1	0	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					
	W				H					
	W				R					
	W				N					
	W				EOT					
	W				GPL					
Execution	W				DTL					Data transfer between FDD and main system.
Result	R				ST 0					Status ID information after command execution.
	R				ST 1					
	R				ST 2					
	R				C					
	R				H					
	R				R					
	R				N					

Phase	R/W	Data Bus								Remarks
		D7	D6	D5	D4	D3	D2	D1	D0	
Command	W	0	MF	SK	0	0	0	1	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W				C					
	W				H					
	W				R					
	W				N					
	W				EOT					
	W				GPL					
Execution	W				DTL					Data transfer between the FDD and main system. FDC reads all of cylinder's contents from index hole to EOT.
Result	R				ST 0					Status information after command execution. Sector ID information after command execution.
	R				ST 1					
	R				ST 2					
	R				C					
	R				H					
	R				R					
	R				N					
Command	W	0	MF	0	0	1	0	1	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
Execution										The first correct ID information on the cylinder is stored in data register.
Result	R				ST 0					Status information after command execution. Sector ID information during execution phase.
	R				ST 1					
	R				ST 2					
	R				C					
	R				H					
	R				R					
	R				N					

Phase	R/W	Data Bus								Remarks
		D7	D6	D5	D4	D3	D2	D1	D0	
Command	W	0	MF	0	0	1	1	0	0	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W					N				
	W					SC				
	W					GPL				
Execution	W					D				FDC formats an entire cylinder. Status information after command execution. In this case, the ID information has no meaning.
Result	R					ST 0				Status information after command execution. In this case, the ID information has no meaning.
	R					ST 1				
	R					ST 2				
	R					C				
	R					H				
Execution	R					R				Data compared between the FDD and the main system. Status information after command execution. Sector ID information
	R					N				
Command	W	MT	MF	SK	1	0	0	0	1	Command Codes
	W	X	X	X	X	X	HD	US1	US0	
	W					C				
	W					H				
	W					R				
Execution	W					N				Sector ID information prior to command execution.
	W					EOT				
	W					GPL				
	W					STP				
Result	R					ST 0				Data compared between the FDD and the main system. Status information after command execution. Sector ID information
	R					ST 1				
	R					ST 2				
	R					C				
	R					H				
Execution	R					R				Data compared between the FDD and the main system. Status information after command execution. Sector ID information
	R					N				

Phase	R/W	Data Bus								Remarks
		D7	D6	D5	D4	D3	D2	D1	D0	
Command	W	Scan Low or Equal								Command Codes Sector ID information prior to command execution.
	W	MT	MF	SK	1	1	0	0	1	
	W	X	X	X	X	X	HD	US1	US0	
	W				C					
	W				H					
	W				R					
	W				N					
	W				EOT					
Execution	W				GPL					Data compared between the FDD and main system. Status information after command execution. Sector ID information after command execution.
	W				STP					
Result	R				ST 0					
	R				ST 1					
	R				ST 2					
	R				C					
	R				H					
	R				R					
	R				N					
Command	W	Scan High or Equal								Command Codes Sector ID information prior to command execution.
	W	MT	MF	SK	1	1	1	0	1	
	W	X	X	X	X	X	HD	US1	US0	
	W				C					
	W				H					
	W				R					
	W				N					
	W				EOT					
Execution	W				GPL					Data compared between the FDD and main system. Status information after command execution. Sector ID information after command execution.
	W				STP					
Result	R				ST 0					
	R				ST 1					
	R				ST 2					
	R				C					
	R				H					
	R				R					
	R				N					

Phase	R/W	Data Bus D7 D6 D5 D4 D3 D2 D1 D0								Remarks
Command Execution No Result Phase	W	Recalibrate								Command Codes Head retracted to track 0
	W	0	0	0	0	0	1	1	1	
		X	X	X	X	X	0	US1	US0	
Command Result	W	Sense Interrupt Status								Command Codes Status information at the end of seek operation about the FDC
	R	0	0	0	0	1	0	0	0	
	R	ST 0 PCN								
Command No Result Phase	W	Specify								Command Codes
	W	0	0	0	0	0	0	1	1	
	W	SRT ————— HUT ————— HLT ————— ND								
Command Result	W	Sense Drive Status								Command Codes Status information about FDD.
	W	0	0	0	0	0	1	0	0	
	R	X	X	X	X	X	HD	US1	US0	
		ST 3								
Command Execution No Result Phase	W	Seek								Command Codes Head is positioned over proper cylinder on diskette.
	W	0	0	0	0	1	1	1	1	
	W	X	X	X	X	X	HD	US1	US0	
		NCN								
Command Result	W	Invalid Invalid Codes								Invalid command codes (NoOp - FDC goes into standby state). ST 0 = 80.
	R	ST 0								

Bit			Description
No.	Name	Symbol	
D7 D6	Interrupt Code	IC	D7 = 0 and D6 = 0 Normal termination of command (NT). Command was completed and properly executed. D7 = 0 and D6 = 1 Abnormal termination of command (AT). Execution of command was started, but was not successfully completed. D7 = 1 and D6 = 0 Invalid command issue (IC). Command that was issued was never started. D7 = 1 and D6 = 1 Abnormal termination because, during command execution, the ready signal from FDD changed state.
D5	Seek End	SE	When the FDC completes the seek command, this flag is set to 1 (high).
D4	Equipment Check	EC	If a fault signal is received from the FDD, or if the track 0 signal fails to occur after 77 step pulses (recalibrate command), then this flag is set.
D3	Not Ready	NR	When the FDD is in the not-ready state and a read or write command is issued, this flag is set. If a read or write command is issued to side 1 of a single-sided drive, then this flag is set.
D2	Head Address	HD	This flag is used to indicate the state of the head at interrupt.
D1 D0	Unit Select 1 Unit Select 2	US 1 US 2	These flags are used to indicate a drive unit number at interrupt.

Command Status Register 0

Bit			Description
No.	Name	Symbol	
D7	End of Cylinder	EN	When the FDC tries to access a sector beyond the final sector of a cylinder, this flag is set.
D6	—	—	Not used. This bit is always 0 (low).
D5	Data Error	DE	When the FDC detects a CRC error in either the ID field or the data field, this flag is set.
D4	Over Run	OR	If the FDC is not serviced by the main system during data transfers within a certain time interval, this flag is set.
D3	—	—	Not used. This bit is always 0 (low).
D2	No Data	ND	During execution of a read data, write deleted data, or scan command, if the FDC cannot find the sector specified in the ID register, this flag is set. During execution of the read ID command, if the FDC cannot read the ID field without an error, then this flag is set. During the execution of the read a cylinder command, if the starting sector cannot be found, then this flag is set.
D1	Not Writable	NW	During execution of a write data, write deleted data, or format-a-cylinder command, if the FDC detects a write-protect signal from the FDD, then this flag is set.
D0	Missing Address Mark	MA	If the FDC cannot detect the ID address mark, this flag is set. Also, at the same time, the MD (missing address mark in the data field) of status register 2 is set.

Command Status Register 1

Bit			Description
No.	Name	Symbol	
D7	—	—	Not used. This bit is always 0 (low).
D6	Control Mark	CM	During execution of the read data or scan command, if the FDC encounters a sector that contains a deleted data address mark, this flag is set.
D5	Data Error in Data Field	DD	If the FDC detects a CRC error in the data, then this flag is set.
D4	Wrong Cylinder	WC	This bit is related to the ND bit, and when the contents of C on the medium are different from that stored in the ID register, this flag is set.
D3	Scan Equal Hit	SH	During execution of the scan command, if the condition of "equal" is satisfied, this flag is set.
D2	Scan Not Satisfied	SN	During execution of the scan command, if the FDC cannot find a sector on the cylinder that meets the condition, then this flag is set.
D1	Bad Cylinder	BC	This bit is related to the ND bit, and when the contents of C on the medium are different from that stored in the ID register, and the contents of C is FF, then this flag is set.
D0	Missing Address Mark in Data Field	MD	When data is read from the medium, if the FDC cannot find a data address mark or deleted data address mark, then this flag is set.

Command Status Register 2

Bit			Description
No.	Name	Symbol	
D7	Fault	FT	This bit is the status of the fault signal from the FDD.
D6	Write Protected	WP	This bit is the status of the write-protected signal from the FDD.
D5	Ready	RY	This bit is the status of the ready signal from the FDD.
D4	Track 0	TO	This bit is the status of the track 0 signal from the FDD.
D3	Two Side	TS	This bit is the status of the two-side signal from the FDD.
D2	Head Address	HD	This bit is the status of the side-select signal from the FDD.
D1	Unit Select 1	US 1	This bit is the status of the unit-select-1 signal from the FDD.
D0	Unit Select 0	US 0	This bit is the status of the unit-select-0 signal from the FDD.

Command Status Register 3

Programming Summary

FDC Data Register	I/O Address Hex 3F5
FDC Main Status Register	I/O Address Hex 3F4
Digital Output Register	I/O Address Hex 3F2
Bit 0	Drive 00: DR #A 10: DR #C
1	Select 01: DR #B 11: DR #D
2	Not FDC Reset
3	Enable INT & DMA Requests
4	Drive A Motor Enable
5	Drive B Motor Enable
6	Drive C Motor Enable
7	Drive D Motor Enable
All bits cleared with channel reset.	

DPC Registers

FDC Constants (in hex)

N:	02	GPL Format:	05
SC:	08	GPL R/W:	2A
HUT:	F	HLT:	01
SRT:	C		(6 ms track-to-track)

Drive Constants

Head Load	35 ms
Head Settle	15 ms
Motor Start	250 ms

Comments

- Head loads with drive select, wait HD load time before R/W.
- Following access, wait HD settle time before R/W.
- Drive motors should be off when not in use. Only A or B and C or D may run simultaneously. Wait motor start time before R/W.
- Motor must be on for drive to be selected.
- Data errors can occur while using a home television as the system display. Locating the TV too close to the diskette area can cause this to occur. To correct the problem, move the TV away from, or to the opposite side of the system unit.

System I/O Channel Interface

All signals are TTL-compatible:

Most Positive Up Level	5.5 Vdc
Least Positive Up Level	2.7 Vdc
Most Positive Down Level	0.5 Vdc
Least Positive Down Level	-0.5 Vdc

The following lines are used by this adapter.

- +D0-7 (Bidirectional, load: 1 74LS, driver: 74LS 3-state).
These eight lines form a bus by which all commands, status, and data are transferred. Bit 0 is the low-order bit.
- +A0-9 (Adapter input, load: 1 74LS)
These ten lines form an address bus by which a register is selected to receive or supply the byte transferred through lines D0-7. Bit 0 is the low-order bit.
- +AEN (Adapter input, load: 1 74LS)
The content of lines A0-9 is ignored if this line is active.
- IOW (Adapter input, load: 1 74LS)
The content of lines D0-7 is stored in the register addressed by lines A0-9 or DACK2 at the trailing edge of this signal.
- IOR (Adapter input, load: 1 74LS)
The content of the register addressed by lines A0-9 or DACK2 is gated onto lines D0-7 when this line is active.
- DACK2 (Adapter input, load: 2 74LS)
This line being active negates output DRQ2, selects the FDC data register as the source/destination of bus D0-7, and indirectly gates T/C to IRQ6.
- +T/C (Adapter input, load: 4 74LS)
This line and DACK2 being active indicates that the byte of data for which the DMA count was initialized is now being transferred.
- +RESET (Adapter input, load: 1 74LS)
An up level aborts any operation in process and clears the digital output register (DOR).
- +DRQ2 (Adapter output, driver: 74LS 3-state)
This line is made active when the attachment is ready to transfer a byte of data to or from main storage. The line is made inactive by DACK2 becoming active or an I/O read of the FDC data register.

- +IRQ6 (Adapter output, driver: 74LS 3-state)
This line is made active when the FDC has completed an operation. It results in an interrupt to a routine which should examine the FDC result bytes to reset the line and determine the ending condition.

Drive A and B Interface

All signals are TTL-compatible:

Most Positive Up Level	5.5 Vdc
Least Positive Up Level	2.4 Vdc
Most Positive Down Level	0.4 Vdc
Least Positive Down Level	-0.5 Vdc

All adapter outputs are driven by open-collector gates. The drive(s) must provide termination networks to Vcc (except motor enable, which has a 2000-ohm resistor to Vcc).

Each adapter input is terminated with a 150-ohm resistor to Vcc.

Adapter Outputs

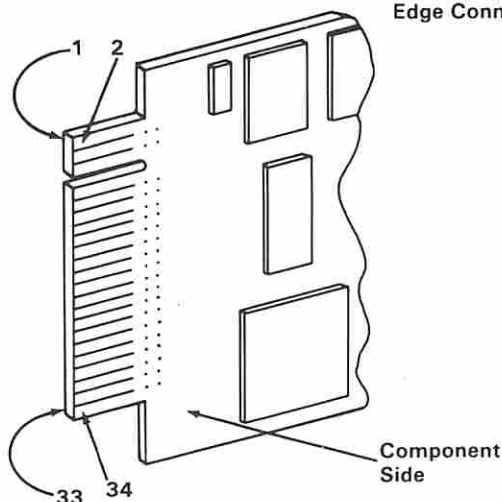
- Drive Select A and B (Driver: 7438)
These two lines are used by drives A and B to degate all drivers to the adapter and receivers from the attachment (except motor enable) when the line associated with a drive is inactive.
- Motor Enable A and B (Driver: 7438)
The drive associated with each of these lines must control its spindle motor such that it starts when the line becomes active and stops when the line becomes inactive.
- Step (Driver: 7438)
The selected drive moves the read/write head one cylinder in or out per the direction line for each pulse present on this line.

- Direction (Driver: 7438)
For each recognized pulse of the step line, the read/write head moves one cylinder toward the spindle if this line is active, and away from the spindle if inactive.
- Head Select (Driver: 7438)
Head 1 (upper head) will be selected when this line is active (low).
- Write Data (Driver: 7438)
For each inactive to active transition of this line while write enable is active, the selected drive causes a flux change to be stored on the diskette.

Adapter Inputs

- Write Enable (Driver: 7438)
The drive disables write current in the head unless this line is active.
- Index
The selected drive supplies one pulse per diskette revolution on this line.
- Write Protect
The selected drive makes this line active if a write-protected diskette is mounted in the drive.
- Track 0
The selected drive makes this line active if the read/write head is over track 0.
- Read Data
The selected drive supplies a pulse on this line for each flux change encountered on the diskette.

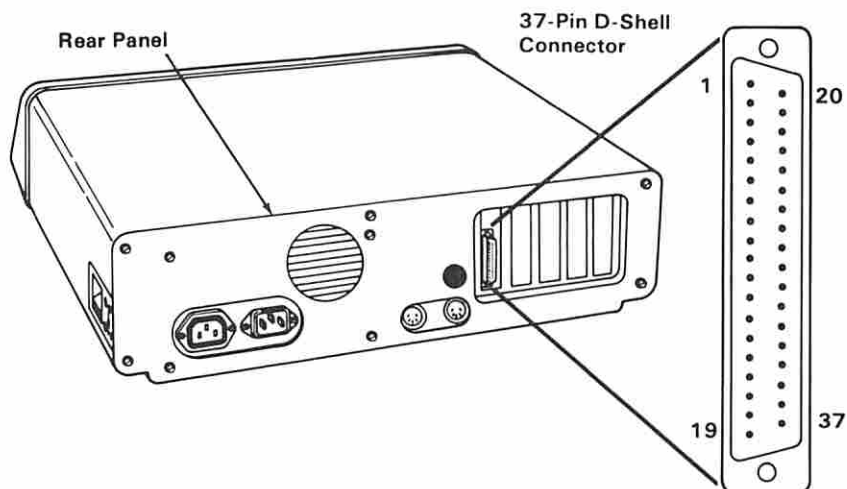
34-Pin Keyed Edge Connector



Note: Lands 1-33 (odd numbers) are on the back of the board. Lands 2-34 (even numbers) are on the front, or component side.

	At Standard TTL Levels	Land Number	
	Ground-Odd Numbers	1-33	
	Unused	2,4,6	
	Index	8	
	Motor Enable A	10	→
	Drive Select B	12	←
	Drive Select A	14	←
	Motor Enable B	16	←
	Direction (Stepper Motor)	18	
Diskette Drives	Step Pulse	20	→
	Write Data	22	
	Write Enable	24	←
	Track 0	26	→
	Write Protect	28	→
	Read Data	30	→
	Select Head 1	32	←
	Unused	34	
			Drive Adapter

Connector Specifications (Part 1 of 2)



At Standard TTL Levels		Pin Number
Unused		1-5
Index		6
Motor Enable C		7
Drive Select D		8
Drive Select C		9
Motor Enable D		10
Direction (Stepper Motor)		11
Step Pulse		12
Write Data		13
Write Enable		14
Track 0		15
Write Protect		16
Read Data		17
Select Head 1		18
Ground		20-37

Connector Specifications (Part 2 of 2)

Notes:

IBM 5-1/4" Diskette Drive

The system unit has space and power for one or two 5-1/4 inch diskette drives. A drive can be single-sided or double-sided with 40 tracks for each side, is fully self-contained, and consists of a spindle drive system, a read positioning system, and a read/write/erase system.

The diskette drive uses modified frequency modulation (MFM) to read and write digital data, with a track-to-track access time of 6 milliseconds.

To load a diskette, the operator raises the latch at the front of the diskette drive and inserts the diskette into the slot. Plastic guides in the slot ensure the diskette is in the correct position. Closing the latch centers the diskette and clamps it to the drive hub. After 250 milliseconds, the servo-controlled dc drive motor starts and drives the hub at a constant speed of 300 rpm. The head positioning system, which consists of a 4-phase stepper-motor and band assembly with its associated electronics, moves the magnetic head so it comes in contact with the desired track of the diskette. The stepper-motor and band assembly uses one-step rotation to cause a one-track linear movement of the magnetic head. No operator intervention is required during normal operation. During a write operation, a 0.013-inch (0.33 millimeter) data track is recorded, then tunnel-erased to 0.012 inch (0.030 millimeter). If the diskette is write-protected, a write-protect sensor disables the drive's circuitry, and an appropriate signal is sent to the interface.

Data is read from the diskette by the data-recovery circuitry, which consists of a low-level read amplifier, differentiator, zero-crossing detector, and digitizing circuits. All data decoding is done by an adapter card.

The diskette drive also has the following sensor systems:

1. The track 00 switch, which senses when the head/carriage assembly is at track 00.
2. The index sensor, which consists of an LED light source and phototransistor. This sensor is positioned so that when an index hole is detected, a digital signal is generated.

3. The write-protect sensor disables the diskette drive's electronics whenever a write-protect tab is applied to the diskette.

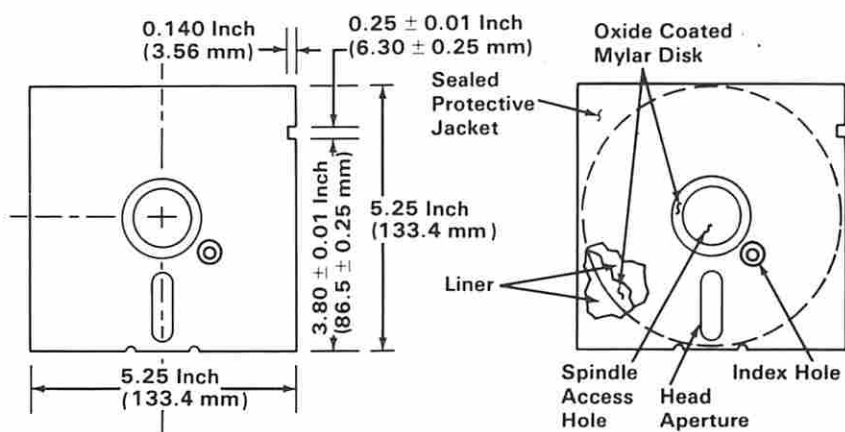
For interface information, refer to "IBM 5-1/4" Diskette Drive Adapter" earlier in this section.

Media	Industry-compatible 5-1/4 inch diskette
Tracks per inch	48
Number of tracks	40
Dimensions	
Height	3.38 inches (85.85 mm)
Width	5.87 inches (149.10 mm)
Depth	8.00 inches (203.2 mm)
Weight	4.50 pounds (2.04 kg)
Temperature	
(Exclusive of media)	
Operating	50°F to 112°F (10°C to 44°C)
Non-operating	-40°F to 140°F (-40°C to 60°C)
Relative humidity	
(Exclusive of media)	
Operating	20% to 80% (non-condensing)
Non-operating	5% to 95% (non-condensing)
Seek Time	6 ms track-to-track
Head Settling Time	15 ms (last track addressed)
Error Rate	1 per 10 ⁹ (recoverable) 1 per 10 ¹² (non-recoverable) 1 per 10 ⁶ (seeks)
Head Life	20,000 hours (normal use)
Media Life	3.0 x 10 ⁶ passes per track
Disk Speed	300 rpm +/- 1.5% (long term)
Instantaneous Speed Variation	+/- 3.0%
Start/Stop Time	250 ms (maximum)
Transfer Rate	250K bits/sec
Recording Mode	MFM
Power	+12 Vdc +/- 0.6 V, 900 mA average +5 Vdc +/- 0.25 V, 600 mA average

Mechanical and Electrical Specifications

Diskettes

The IBM 5-1/4" Diskette Drive uses a standard 5.25-inch (133.4-millimeter) diskette. For programming considerations, single-sided, double-density, soft-sectored diskettes are used for single-sided drives. Double-sided drives use double-sided, double-density, soft-sectored diskettes. The figure below is a simplified drawing of the diskette used with the diskette drive. This recording medium is a flexible magnetic disk enclosed in a protective jacket. The protected disk, free to rotate within the jacket, is continuously cleaned by the soft fabric lining of the jacket during normal operation. Read/write/erase head access is made through an opening in the jacket. Openings for the drive hub and diskette index hole are also provided.



Recording Medium

Notes:

IBM Fixed Disk Drive Adapter

The fixed disk drive adapter attaches to one or two fixed disk drive units, through an internal daisy-chained flat cable (data/control cable). Each system supports a maximum of one fixed disk drive adapter and two fixed disk drives.

The adapter is buffered on the I/O bus and uses the system board direct memory access (DMA) for record data transfers. An interrupt level also is used to indicate operation completion and status conditions that require processor attention.

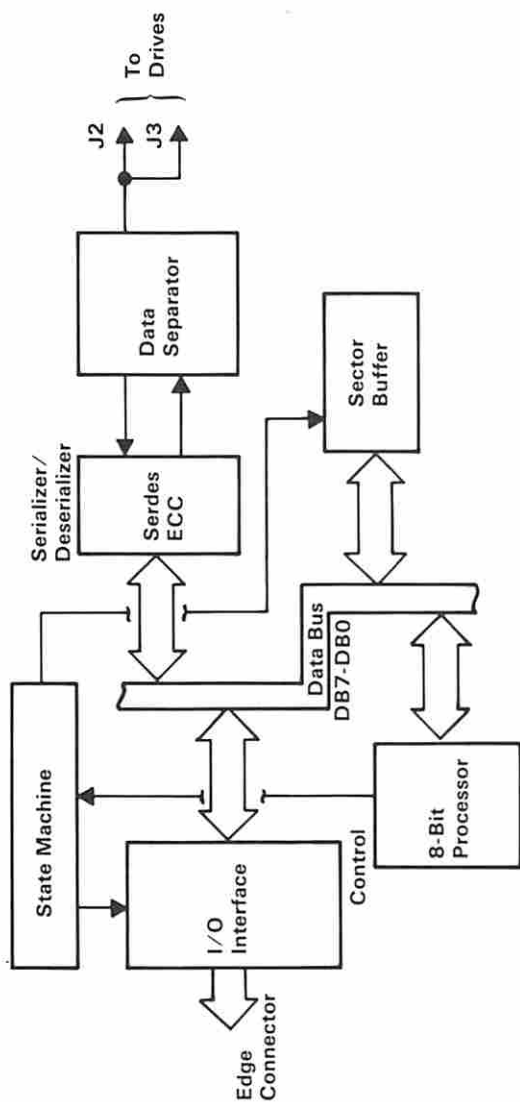
The fixed disk drive adapter provides automatic 11-bit burst error detection and correction in the form of 32-bit error checking and correction (ECC).

The device level control for the fixed disk drive adapter is contained on a ROM module on the adapter. A listing of this device level control can be found in "Appendix A: ROM BIOS Listings."

Warning: The last cylinder on the fixed disk drive is reserved for diagnostic use. Diagnostic write tests will destroy any data on this cylinder.

Fixed Disk Controller

The disk controller has two registers that may be accessed by the main system processor: a status register and a data register. The 8-bit status register contains the status information of the disk controller, and can be accessed at any time. The 8-bit data register (actually consisting of several registers in a stack with only one register presented to the data bus) stores data, commands, parameters, and provides the disk controller's status information. Data bytes are read from, or written to the data register in order to program or obtain the results after a particular command. The status register is a read-only register, and is used to help the transfer of data between the processor and the disk controller. The controller-select pulse is generated by writing to port address hex 322.



Fixed Disk Drive Adapter Block Diagram

Programming Considerations

Status Register

At the end of all commands from the system board, the disk controller returns a completion status byte back to the system board. This byte informs the system unit if an error occurred during the execution of the command. The following shows the format of this byte.

Bit	7	6	5	4	3	2	1	0
	0	0	d	0	0	0	e	0

Bits 0, 1, 2, 3, 4, 6, 7 These bits are set to zero.

Bit 1 When set, this bit shows an error has occurred during command execution.

Bit 5 This bit shows the logical unit number of the drive.

If the interrupts are enabled, the controller sends an interrupt when it is ready to transfer the status byte. Busy from the disk controller is unasserted when the byte is transferred to complete the command.

Sense Bytes

If the status register receives an error (bit 1 is set), then the disk controller requests four bytes of sense data. The format for the four bytes is as follows:

Bits	7	6	5	4	3	2	2	0
Byte 0	Address Valid	0	Error Type		Error Code			
Byte 1	0	0	d	Head Number				
Byte 2	Cylinder High			Sector Number				
Byte 3	Cylinder Low							

Remarks

d = drive

Byte 0	Bits 0, 1, 2, 3	Error code.
Byte 0	Bits 4, 5	Error type.
Byte 0	Bit 6	Set to 0 (spare).
Byte 0	Bit 7	The address valid bit. Set only when the previous command required a disk address, in which case it is returned as a 1; otherwise, it is a 0.

The following disk controller error tables list the error types and error codes found in byte 0:

	Error Type	Error Code	Description
Bits	5 4	3 2 1 0	
	0 0	0 0 0 0	The controller did not detect any error during the execution of the previous operation.
	0 0	0 0 0 1	The controller did not detect an index signal from the drive.
	0 0	0 0 1 0	The controller did not get a seek-complete signal from the drive after a seek operation (for all non-buffered step seeks).
	0 0	0 0 1 1	The controller detected a write fault from the drive during the last operation.
	0 0	0 1 0 0	After the controller selected the drive, the drive did not respond with a ready signal.
	0 0	0 1 0 1	Not used.
	0 0	0 1 1 0	After stepping the maximum number of cylinders, the controller did not receive the track 00 signal from the drive.
	0 0	0 1 1 1	Not used.
	0 0	1 0 0 0	The drive is still seeking. This status is reported by the Test Drive Ready command for an overlap seek condition when the drive has not completed the seek. No time-out is measured by the controller for the seek to complete.

	Error Type	Error Code	Description
Bits	5 4	3 2 1 0	
	0 1	0 0 0 0	ID Read Error: The controller detected an ECC error in the target ID field on the disk.
	0 1	0 0 0 1	Data Error: The controller detected an uncorrectable ECC error in the target sector during a read operation.
	0 1	0 0 1 0	Address Mark: The controller did not detect the target address mark (AM) on the disk.
	0 1	0 0 1 1	Not used.
	0 1	0 1 0 0	Sector Not Found: The controller found the correct cylinder and head, but not the target sector.
	0 1	0 1 0 1	Seek Error: The cylinder or head address (either or both) did not compare with the expected target address as a result of a seek.
	0 1	0 1 1 0	Not used.
	0 1	0 1 1 1	Not used.
	0 1	1 0 0 0	Correctable Data Error: The controller detected a correctable ECC error in the target field.
	0 1	1 0 0 1	Bad Track: The controller detected a bad track flag during the last operation. No retries are attempted on this error.

	Error Type	Error Code	Description
Bits	5 4	3 2 1 0	
	1 0	0 0 0 0	Invalid Command: The controller has received an invalid command from the system unit.
	1 0	0 0 0 1	Illegal Disk Address: The controller detected an address that is beyond the maximum range.

	Error Type	Error Code	Description
Bits	5 4	3 2 1 0	
	1 1	0 0 0 0	RAM Error: The controller detected a data error during the RAM sector-buffer diagnostic test.
	1 1	0 0 0 1	Program Memory Checksum Error: During this internal diagnostic test, the controller detected a program-memory checksum error.
	1 1	0 0 1 0	ECC Polynomial Error: During the controller's internal diagnostic tests, the hardware ECC generator failed its test.

Data Register

The processor specifies the operation by sending the 6-byte device control block (DCB) to the controller. The figure below shows the composition of the DCB, and defines the bytes that make up the DCB.

Bit	7	6	5	4	3	2	1	0
Byte 0	Command Class			Opcode				
Byte 1	0	0	d	Head Number				
Byte 2	Cylinder High		Sector Number					
Byte 3	Cylinder Low							
Byte 4	Interleave or Block Count							
Byte 5	Control Field							

Byte 0 - Bits 7, 6, and 5 identify the class of the command.
Bits 4 through 0 contain the Opcode command.

Byte 1 - Bit 5 identifies the drive number.
Bits 4 through 0 contain the disk head number to be selected.
Bits 6 and 7 are not used.

Byte 2 - Bits 6 and 7 contain the two most significant bits of the cylinder number.
Bits 0 through 5 contain the sector number.

Byte 3 - Bits 0 through 7 are the eight least significant bits of the cylinder number.

Byte 4 - Bits 0 through 7 specify the interleave or block count.

Byte 5 - Bits 0 through 7 contain the control field.

Control Byte

Byte 5 is the control field of the DCB and allows the user to select options for several types of disk drives. The format of this byte is as follows:

Bits	7	6	5	4	3	2	1	0
	r	a	0	0	0	s	s	s

Remarks

r = retries

s = step option

a = retry option on data ECC error

Bit 7 Disables the four retries by the controller on all disk-access commands. Set this bit only during the evaluation of the performance of a disk drive.

Bit 6 If set to 0 during read commands, a reread is attempted when an ECC error occurs. If no error occurs during reread, the command will complete with no error status. If this bit is set to 1, no reread is attempted.

Bits 5, 4, 3 Set to 0.

Bits 2, 1, 0 These bits define the type of drive and select the step option. See the following figure.

Bits	2, 1, 0	
0 0 0	This drive is not specified and defaults to 3 milliseconds per step.	
0 0 1	N/A	
0 1 0	N/A	
0 1 1	N/A	
1 0 0	200 microseconds per step.	
1 0 1	70 microseconds per step (specified by BIOS).	
1 1 0	3 milliseconds per step.	
1 1 1	3 milliseconds per step.	

Command Summary

Command	Data Control Block	Remarks																																																															
Test Drive Ready (Class 0, Opcode 00)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 0	0	0	0	0	0	0	0	0	Byte 1	0	0	d	x	x	x	x	x	d = drive (0 or 1) x = don't care Bytes 2, 3, 4, 5 = don't care																																				
Bit	7	6	5	4	3	2	1	0																																																									
Byte 0	0	0	0	0	0	0	0	0																																																									
Byte 1	0	0	d	x	x	x	x	x																																																									
Recalibrate (Class 0, Opcode 01)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr><tr><td>Byte 5</td><td>r</td><td>0</td><td>0</td><td>0</td><td>0</td><td>s</td><td>s</td><td>s</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 0	0	0	0	0	0	0	0	1	Byte 1	0	0	d	x	x	x	x	x	Byte 5	r	0	0	0	0	s	s	s	d = drive (0 or 1) x = don't care r = retries s = Step Option Bytes 2, 3, 4 = don't care ch = cylinder high																											
Bit	7	6	5	4	3	2	1	0																																																									
Byte 0	0	0	0	0	0	0	0	1																																																									
Byte 1	0	0	d	x	x	x	x	x																																																									
Byte 5	r	0	0	0	0	s	s	s																																																									
Reserved (Class 0, Opcode 02)		This Opcode is not used.																																																															
Request Sense Status (Class 0, Opcode 03)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td></tr><tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 0	0	0	0	0	0	0	1	1	Byte 1	0	0	d	x	x	x	x	x	d = drive (0 or 1) x = don't care Bytes 2, 3, 4, 5 = don't care																																				
Bit	7	6	5	4	3	2	1	0																																																									
Byte 0	0	0	0	0	0	0	1	1																																																									
Byte 1	0	0	d	x	x	x	x	x																																																									
Format Drive (Class 0, Opcode 04)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td colspan="5">Head Number</td></tr><tr><td>Byte 2</td><td>ch</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>Byte 3</td><td colspan="8">Cylinder Low</td></tr><tr><td>Byte 4</td><td>0</td><td>0</td><td>0</td><td colspan="5">Interleave</td></tr><tr><td>Byte 5</td><td>r</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>s</td><td>s</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 0	0	0	0	0	0	1	0	0	Byte 1	0	0	d	Head Number					Byte 2	ch	0	0	0	0	0	0	0	Byte 3	Cylinder Low								Byte 4	0	0	0	Interleave					Byte 5	r	0	0	0	0	0	s	s	d = drive (0 or 1) r = retries s = step option ch = cylinder high Interleave: 1 to 16 for 512-byte sectors.
Bit	7	6	5	4	3	2	1	0																																																									
Byte 0	0	0	0	0	0	1	0	0																																																									
Byte 1	0	0	d	Head Number																																																													
Byte 2	ch	0	0	0	0	0	0	0																																																									
Byte 3	Cylinder Low																																																																
Byte 4	0	0	0	Interleave																																																													
Byte 5	r	0	0	0	0	0	s	s																																																									
Ready Verify (Class 0, Opcode 05)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td></tr><tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td colspan="5">Head Number</td></tr><tr><td>Byte 2</td><td>ch</td><td colspan="7">Sector Number</td></tr><tr><td>Byte 3</td><td colspan="8">Cylinder Low</td></tr><tr><td>Byte 4</td><td colspan="8">Block Count</td></tr><tr><td>Byte 5</td><td>r</td><td>a</td><td>0</td><td>0</td><td>0</td><td>s</td><td>s</td><td>s</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 1	0	0	0	0	0	1	0	1	Byte 1	0	0	d	Head Number					Byte 2	ch	Sector Number							Byte 3	Cylinder Low								Byte 4	Block Count								Byte 5	r	a	0	0	0	s	s	s	d = drive (0 or 1) r = retries s = step option a = retry option on data ECC ch = cylinder high
Bit	7	6	5	4	3	2	1	0																																																									
Byte 1	0	0	0	0	0	1	0	1																																																									
Byte 1	0	0	d	Head Number																																																													
Byte 2	ch	Sector Number																																																															
Byte 3	Cylinder Low																																																																
Byte 4	Block Count																																																																
Byte 5	r	a	0	0	0	s	s	s																																																									

Command	Data Control Block	Remarks																																																															
Format Track (Class 0, Opcode 06)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td></tr><tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td colspan="5">Head Number</td></tr><tr><td>Byte 2</td><td>ch</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>Byte 3</td><td colspan="8">Cylinder Low</td></tr><tr><td>Byte 4</td><td>0</td><td>0</td><td>0</td><td colspan="5">Interleave</td></tr><tr><td>Byte 5</td><td>r</td><td>0</td><td>0</td><td>0</td><td>0</td><td>s</td><td>s</td><td>s</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 0	0	0	0	0	0	1	1	0	Byte 1	0	0	d	Head Number					Byte 2	ch	0	0	0	0	0	0	0	Byte 3	Cylinder Low								Byte 4	0	0	0	Interleave					Byte 5	r	0	0	0	0	s	s	s	d = drive (0 or 1) r = retries s = step option ch =cylinder high Interleave: 1 to 16 for 512-byte sectors
	Bit	7	6	5	4	3	2	1	0																																																								
	Byte 0	0	0	0	0	0	1	1	0																																																								
	Byte 1	0	0	d	Head Number																																																												
	Byte 2	ch	0	0	0	0	0	0	0																																																								
	Byte 3	Cylinder Low																																																															
	Byte 4	0	0	0	Interleave																																																												
Byte 5	r	0	0	0	0	s	s	s																																																									
Format Bad Track (Class 0, Opcode 07)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td></tr><tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td colspan="5">Head Number</td></tr><tr><td>Byte 2</td><td>ch</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>Byte 3</td><td colspan="8">Cylinder Low</td></tr><tr><td>Byte 4</td><td>0</td><td>0</td><td>0</td><td colspan="5">Interleave</td></tr><tr><td>Byte 5</td><td>r</td><td>0</td><td>0</td><td>0</td><td>0</td><td>s</td><td>s</td><td>s</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 0	0	0	0	0	0	1	1	1	Byte 1	0	0	d	Head Number					Byte 2	ch	0	0	0	0	0	0	0	Byte 3	Cylinder Low								Byte 4	0	0	0	Interleave					Byte 5	r	0	0	0	0	s	s	s	d = drive (0 or 1) r = retries s = step option ch = cylinder high Interleave: 1 to 16 for 512-byte sectors
	Bit	7	6	5	4	3	2	1	0																																																								
	Byte 0	0	0	0	0	0	1	1	1																																																								
	Byte 1	0	0	d	Head Number																																																												
	Byte 2	ch	0	0	0	0	0	0	0																																																								
	Byte 3	Cylinder Low																																																															
	Byte 4	0	0	0	Interleave																																																												
Byte 5	r	0	0	0	0	s	s	s																																																									
Read (Class 0, Opcode 08)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td></tr><tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td colspan="5">Head Number</td></tr><tr><td>Byte 2</td><td>ch</td><td colspan="7">Sector Number</td></tr><tr><td>Byte 3</td><td colspan="8">Cylinder Low</td></tr><tr><td>Byte 5</td><td>r</td><td>a</td><td>0</td><td>0</td><td>0</td><td>s</td><td>s</td><td>s</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 0	0	0	0	0	1	0	0	0	Byte 1	0	0	d	Head Number					Byte 2	ch	Sector Number							Byte 3	Cylinder Low								Byte 5	r	a	0	0	0	s	s	s	d = drive (0 or 1) r = retries a = retry option on data ECC error s = step option ch =cylinder high									
	Bit	7	6	5	4	3	2	1	0																																																								
	Byte 0	0	0	0	0	1	0	0	0																																																								
	Byte 1	0	0	d	Head Number																																																												
	Byte 2	ch	Sector Number																																																														
	Byte 3	Cylinder Low																																																															
Byte 5	r	a	0	0	0	s	s	s																																																									
Reserved (Class 0, Opcode 09)		This Opcode is not used																																																															
Write (Class 0, Opcode 0A)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td></tr><tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td colspan="5">Head Number</td></tr><tr><td>Byte 2</td><td>ch</td><td colspan="7">Sector Number</td></tr><tr><td>Byte 3</td><td colspan="8">Cylinder Low</td></tr><tr><td>Byte 4</td><td colspan="8">Block Count</td></tr><tr><td>Byte 5</td><td>r</td><td>0</td><td>0</td><td>0</td><td>0</td><td>s</td><td>s</td><td>s</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 0	0	0	0	0	1	0	1	0	Byte 1	0	0	d	Head Number					Byte 2	ch	Sector Number							Byte 3	Cylinder Low								Byte 4	Block Count								Byte 5	r	0	0	0	0	s	s	s	d = drive (0 or 1) r = retries s = step option ch = cylinder high
	Bit	7	6	5	4	3	2	1	0																																																								
	Byte 0	0	0	0	0	1	0	1	0																																																								
	Byte 1	0	0	d	Head Number																																																												
	Byte 2	ch	Sector Number																																																														
	Byte 3	Cylinder Low																																																															
	Byte 4	Block Count																																																															
Byte 5	r	0	0	0	0	s	s	s																																																									
Seek (Class 0, Opcode 0B)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td><td>1</td></tr><tr><td>Byte 1</td><td>0</td><td>0</td><td>d</td><td colspan="5">Head Number</td></tr><tr><td>Byte 2</td><td>ch</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr><tr><td>Byte 3</td><td colspan="8">Cylinder Low</td></tr><tr><td>Byte 4</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td></tr><tr><td>Byte 5</td><td>r</td><td>0</td><td>0</td><td>0</td><td>0</td><td>s</td><td>s</td><td>s</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 0	0	0	0	0	1	0	1	1	Byte 1	0	0	d	Head Number					Byte 2	ch	0	0	0	0	0	0	0	Byte 3	Cylinder Low								Byte 4	x	x	x	x	x	x	x	x	Byte 5	r	0	0	0	0	s	s	s	d = drive (0 or 1) r = retries s = step option x = don't care ch = cylinder high
	Bit	7	6	5	4	3	2	1	0																																																								
	Byte 0	0	0	0	0	1	0	1	1																																																								
	Byte 1	0	0	d	Head Number																																																												
	Byte 2	ch	0	0	0	0	0	0	0																																																								
	Byte 3	Cylinder Low																																																															
	Byte 4	x	x	x	x	x	x	x	x																																																								
Byte 5	r	0	0	0	0	s	s	s																																																									

Command	Data Control Block	Remarks																		
Initialize Drive Characteristics* (Class 0, Opcode 0C)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 0	0	0	0	0	1	1	0	0	Bytes 1, 2, 3, 4, 5 = don't care
Bit	7	6	5	4	3	2	1	0												
Byte 0	0	0	0	0	1	1	0	0												
Read ECC Burst Error Length (Class 0, Opcode 0D)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td>1</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 0	0	0	0	0	1	1	0	1	Bytes 1, 2, 3, 4, 5 = don't care
Bit	7	6	5	4	3	2	1	0												
Byte 0	0	0	0	0	1	1	0	1												
Read Data from Sector Buffer (Class 0, Opcode 0E)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>0</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 0	0	0	0	0	1	1	1	0	Bytes 1, 2, 3, 4, 5 = don't care
Bit	7	6	5	4	3	2	1	0												
Byte 0	0	0	0	0	1	1	1	0												
Write Data to Sector Buffer (Class 0, Opcode 0F)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 0	0	0	0	0	1	1	1	1	Bytes 1, 2, 3, 4, 5 = don't care
Bit	7	6	5	4	3	2	1	0												
Byte 0	0	0	0	0	1	1	1	1												
RAM Diagnostic (Class 7, Opcode 00)	<table><tr><td>Bit</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr><tr><td>Byte 0</td><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	Bit	7	6	5	4	3	2	1	0	Byte 0	1	1	1	0	0	0	0	0	Bytes 1, 2, 3, 4, 5 = don't care
Bit	7	6	5	4	3	2	1	0												
Byte 0	1	1	1	0	0	0	0	0												
Reserved (Class 7, Opcode 01)		This Opcode is not used																		
Reserved (Class 7, Opcode 02)		This Opcode is not used																		

*Initialize Drive Characteristics: The DCB must be followed by eight additional bytes.

Maximum number of cylinders	(2 bytes)
Maximum number of heads	(1 byte)
Start reduced write current cylinder	(2 bytes)
Start write precompensation cylinder	(2 bytes)
Maximum ECC data burst length	(1 byte)

Command	Data Control Block									Remarks	
Drive Diagnostic (Class 7, Opcode 03)	Bit	7	6	5	4	3	2	1	0	d = drive (0 or 1) s = step option r = retries x = don't care	
	Byte 0	1	1	1	0	0	0	1	1		
	Byte 1	0	0	d	x	x	x	x	x		
	Byte 2	x	x	x	x	x	x	x	x		
	Byte 3	x	x	x	x	x	x	x	x		
	Byte 4	x	x	x	x	x	x	x	x		
	Byte 5	r	0	0	0	0	s	s	s		
Controller Internal Diagnostics (Class 7, Opcode 04)	Bit	7	6	5	4	3	2	1	0	Bytes 1, 2, 3, 4, 5 = don't care	
	Byte 0	1	1	1	0	0	1	0	0		
Read Long* (Class 7, Opcode 05)	Bit	7	6	5	4	3	2	1	0	d = drive (0 or 1) s = step option r = retries ch = cylinder high	
	Byte 0	1	1	1	0	0	1	0	1		
	Byte 1	0	0	d	Head Number						
	Byte 2	ch	Sector Number								
	Byte 3	Cylinder Low									
	Byte 4	Block Count									
	Byte 5	r	0	0	0	0	s	s	s		
Write Long** (Class 7, Opcode 06)	Bit	7	6	5	4	3	2	1	0	d = drive (0 or 1) s = step option r = retries ch = cylinder high	
	Byte 0	1	1	1	0	0	1	1	0		
	Byte 1	0	0	d	Head Number						
	Byte 2	ch	Sector Number								
	Byte 3	Cylinder Low									
	Byte 4	Block Count									
	Byte 5	r	0	0	0	0	s	s	s		

*Returns 512 bytes plus 4 bytes of ECC data per sector.

**Requires 512 bytes plus 4 bytes of ECC data per sector.

Programming Summary

The two least-significant bits of the address bus are sent to the system board's I/O port decoder, which has two sections. One section is enabled by the I/O read signal (-IOR) and the other by the I/O write signal (-IOW). The result is a total of four read/write ports assigned to the disk controller board.

The address enable signal (AEN) is asserted by the system board when DMA is controlling data transfer. When AEN is asserted, the I/O port decoder is disabled.

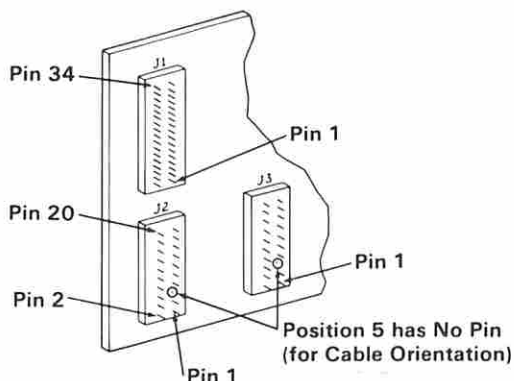
The following figure is a table of the four read/write ports:

R/W	Port Address	Function
Read Write	320 320	Read data (from controller to system unit). Write data (from system unit to controller).
Read Write	321 321	Read controller hardware status. Controller reset.
Read Write	322 322	Reserved. Generate controller-select pulse.
Read Write	323 323	Not used. Write pattern to DMA and interrupt mask register.

System I/O Channel Interface

The following lines are used by the disk controller:

A0 - A19	Positive true 20-bit address. The least-significant 10 bits contain the I/O address within the range of hex 320 to hex 323 when an I/O read or write is executed by the system unit. The full 20 bits are decoded to address the read-only memory (ROM) between the addresses of hex C8000 and C9FFF.
D0 - D7	Positive 8-bit data bus over which data and status information is passed between the system board and the controller.
$\overline{\text{IOR}}$	Negative true signal that is asserted when the system board reads status or data from the controller under either programmed I/O or DMA control.
$\overline{\text{IOW}}$	Negative true signal that is asserted when the system board sends a command or data to the controller under either programmed I/O or DMA control.
AEN	Positive true signal that is asserted when the DMA in the system board is generating the I/O Read (-IOR) or I/O Write (-IOW) signals and has control of the address and data buses.
RESET	Positive true signal that forces the disk controller to its initial power-up condition.
IRQ 5	Positive true interrupt request signal that is asserted by the controller, when enabled to interrupt the system board on the return ending status byte from the controller.
DRQ 3	Positive-true DMA-request signal that is asserted by the controller when data is available for transfer to or from the controller under DMA control. This signal remains active until the system board's DMA channel activates the DMA-acknowledge signal (-DACK 3) in response.
$\overline{\text{DACK 3}}$	This signal is true when negative, and is generated by the system board DMA channel in response to a DMA request (DRQ 3).



Signal	Pin Number
Ground - Odd Numbers	1-33
Reserved	4, 16, 30, 32
-Reduced Write Current	2
-Write Gate	6
-Seek Complete	8
-Track 00	10
-Write Fault	12
-Head Select 2 ⁰	14
-Head Select 2 ¹	18
-Index	20
-Ready	22
-Step	24
-Drive Select 1	26
-Drive Select 2	28
-Direction In	34

Diagram illustrating the pin configurations for connectors J1, J2, and J3. J1 and J2 are 34-pin connectors, while J3 is a 19-pin connector. Pin 1 is indicated for each. A note states "Position 5 has No Pin (for Cable Orientation)".

Signal	Pin Number
Ground	2, 4, 6, 8, 12, 16, 20
Drive Select	1
Reserved	3, 7
Spare	9, 10, 5 (No Pin)
Ground	11
MFM Write Data	13
-MFM Write Data	14
Ground	15
MFM Read Data	17
-MFM Read Data	18
Ground	19

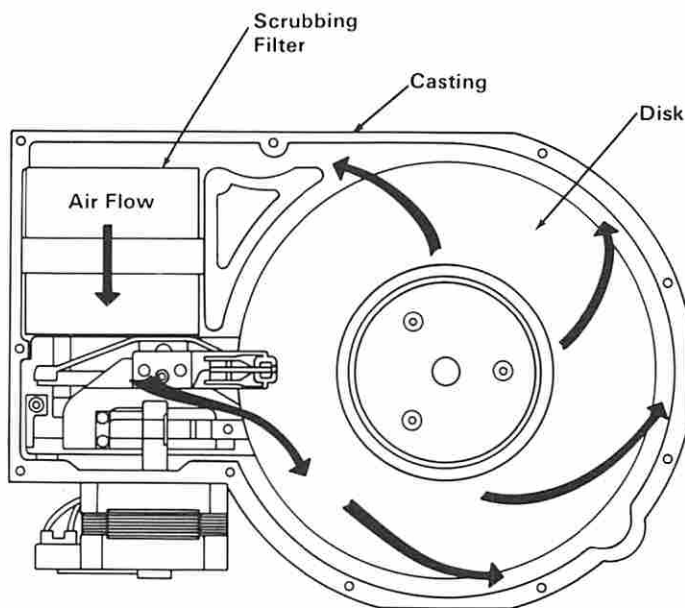
Fixed Disk Adapter Interface Specifications

Notes:

IBM 10MB Fixed Disk Drive

The disk drive is a random-access storage device that uses two non-removable 5-1/4 inch disks for storage. Each disk surface employs one movable head to service 306 cylinders. The total formatted capacity of the four heads and surfaces is 10 megabytes (17 sectors per track with 512 bytes per sector and a total of 1224 tracks).

An impact-resistant enclosure provides mechanical and contamination protection for the heads, actuator, and disks. A self-contained recirculating system supplies clean air through a 0.3-micron filter. Thermal isolation of the stepper and spindle motor assemblies from the disk enclosure results in a very low temperature rise within the enclosure. This isolation provides a greater off-track margin and the ability to perform read and write operations immediately after power-up with no thermal stabilization delay.



Media	Rigid media disk
Number of Tracks	1224
Track Density	345 tracks per inch
Dimensions	
Height	3.25 inches (82.55 mm)
Width	5.75 inches (146.05 mm)
Depth	8.0 inches (203.2 mm)
Weight	4.6 lb (2.08 kg)
Temperature	
Operating	40°F to 122°F (4°C to 50°C)
Non-Operating	-40°F to 140°F (-40°C to 60°C)
Relative Humidity	
Operating	8% to 80% (non-condensing)
Maximum Wet Bulb	78°F (26°C)
Shock	
Operating	10 Gs
Non-Operating	20 Gs
Access Time	3 ms track-to-track
Average Latency	8.33 ms
Error Rates	
Soft Read Errors	1 per 10 ¹⁰ bits read
Hard Read Errors	1 per 10 ¹² bits read
Seek Errors	1 per 10 ⁶ seeks
Design Life	5-years (8,000 hours MTF)
Disk Speed	3600 rpm $\pm$ 1%
Transfer Rate	5.0 M bits/sec
Recording Mode	MFM
Power	+12 Vdc $\pm$ 5% 1.8 A (4.5 A maximum) +5 Vdc $\pm$ 5% 0.7 A (1.0 A maximum)
Maximum Ripple	1% with equivalent resistive load

Mechanical and Electrical Specifications

IBM Memory Expansion Options

Three Memory Expansion Options (32KB, 64KB, and 64/256KB) and two Memory Module Kits (16KB and 64KB) are available for the IBM Personal Computer. Memory expansion is described in the following chart:

	Minimum Memory	Maximum Memory	Number of 16K Memory Module Kits	Number of 64K Memory Module Kits	Memory Module Type
16/64K System Board	16K	64K	1, 2, or 3		16K by 1 Bit, 16 pin
64/256K System Board	64K	256K		1, 2, or 3	64K by 1 Bit, 16 pin
64/256K Memory Option	64K	256K		1, 2, or 3	64K by 1 Bit, 16 pin
32K Memory Option	32K				16K by 1 Bit, 16 pin
64K Memory Option	64K				Stacked 32K by 1 Bit, 18 pin

The system board must be fully populated before any memory expansion options can be installed. An expansion option must be configured to reside at a sequential 32K or 64K memory address boundary within the system address space. This is done by setting the DIP switches on the option.

All memory expansion options are parity checked. If a parity error is detected, a latch is set and an I/O channel check line is activated, indicating an error to the processor.

In addition to the memory modules, the memory expansion options contain the following circuits: bus buffering, dynamic memory timing generation, address multiplexing, and card-select decode logic.

Dynamic-memory refresh timing and address generation are functions that are performed on the system board and made available in the I/O channel for all devices.

To allow the system to address 32K, 64K, or 64/256K memory expansion options, refer to the "System Board and Memory Option Switch Settings" tables, later in this section.

Operating Characteristics

The system board operates at a frequency of 4.77 MHz, which results in a clock cycle of 210 ns.

Normally four clock cycles are required for a bus cycle so that an 840-ns memory cycle time is achieved. Memory-write and memory-read cycles both take four clock cycles, or 840 ns.

General specifications for memory used on all cards are:

	16K by 1 Bit	32K by 1 Bit	64K by 1 Bit
Access	250 ns	250 ns	200 ns
Cycle	410 ns	410 ns	345 ns

Memory Module Description

Both the 32K and the 64K options contain 18 dynamic memory modules. The 32K memory expansion option utilizes 16K by 1 bit modules, and the 64K memory expansion option utilizes 32K by 1 bit modules.

The 64/256K option has four banks of 9 pluggable sockets. Each bank will accept a 64K memory module kit, consisting of 9 (64K by 1) modules. The kits must be installed sequentially into banks 1, 2, and 3. The base 64/256K option comes with modules installed in bank 0, providing 64K of memory. One, two or three 64K bits may be added, upgrading the option to 128K, 192K, or 256K of memory.

The 16K by 1 and the 32K by 1 modules require three voltage levels: +5 Vdc, -5 Vdc, and +12 Vdc. The 64K by 1 modules require only one voltage level of +5 Vdc. All three memory modules require 128 refresh cycles every 2 ns. Absolute maximum access times are:

	16K by 1 Bit	32K by 1 Bit	64K by 1 Bit
From $\overline{\text{RAS}}$	250 ns	250 ns	200 ns
From $\overline{\text{CAS}}$	165 ns	165 ns	115 ns

Pin	16K by 1 Bit Module (used on 32K option and on 16/64K system board)	32K by 1 Bit Module (used on 64K option)	64K by 1 Bit Module (used on 64/256K option and on 64/256K system board)
1	-5 Vdc	-5 Vdc	N/C
2	Data In**	Data In**	Data In***
3	-Write	-Write	-Write
4	-RAS	-RAS 0	-RAS
5	A0	-RAS 1	A0
6	A2	A0	A2
7	A1	A2	A1
8	+12 Vdc	A1	+5 Vdc
9	+5 Vdc	+12 Vdc	A7
10	A5	+5 Vdc	A5
11	A4	A5	A4
12	A3	A4	A3
13	A6	A3	A6
14	Data Out**	A6	Data Out***
15	-CAS	Data Out**	-CAS
16	GND	-CAS 1	GND
17	*	-CAS 0	*
18	*	GND	*

* 16K by 1 and 64K by 1 bit modules have 16 pins.

**Data In and Data Out are tied together (three-state bus).

***Data In and Data Out are tied together on Data Bits 0-7 (three-state bus).

Memory Module Pin Configuration

Switch-Configurable Start Address

Each card has a small DIP module, that contains eight switches. The switches are used to set the card start address as follows:

Number	32K and 64K Options	64/256K Options
1	ON:A 19=0; OFF:A 19=1	ON:A 19=0; OFF:A 19=1
2	ON:A 18=0; OFF:A 18=1	ON:A 18=0; OFF:A 18=1
3	ON:A 17=0; OFF:A 17=1	ON:A 17=0; OFF:A 17=1
4	ON:A 16=0; OFF:A 16=1	ON:A 16=0; OFF:A 16=1
5	ON:A 15=0; OFF:A 15=1 *	ON:Select 64K
6	Not used	ON:Select 128K
7	Not used	ON:Select 192K
8	Used only in 64K RAM Card*	ON:Select 256K

*Switch 8 may be set on the 64K memory expansion option to use only half the memory on the card (that is, 32K). If switch 8 is on, all 64K is accessible. If switch 8 is off, address bit A 15 (as set by switch 5) is used to determine which 32K are accessible, and the 64K option behaves as a 32K option.

DIP Module Start Address

The following method can be used to determine the switch settings for the 32K memory expansion option.

Starting Address = xxxK

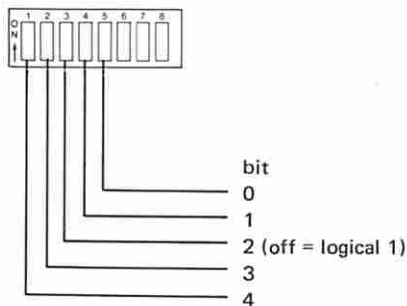
= Decimal value

32K $\overline{\text{xxxK}}$

Convert decimal value to binary

Bit	4	3	2	1	0
Bit value . . .	16	8	4	2	1

Switch



The following method can be used to determine the switch settings for the 64K memory expansion option.

Starting Address = xxxK

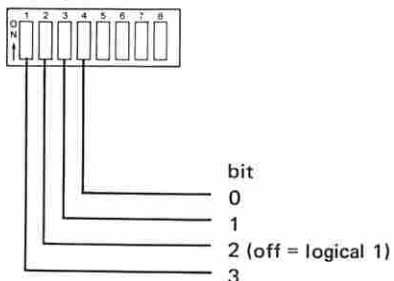
= Decimal value

64K $\overline{\text{xxxK}}$

Convert decimal value to binary

Bit	3	2	1	0
Bit value . . .	8	4	2	1

Switch



The following method can be used to determine the switch settings for the 64/256K memory expansion option.

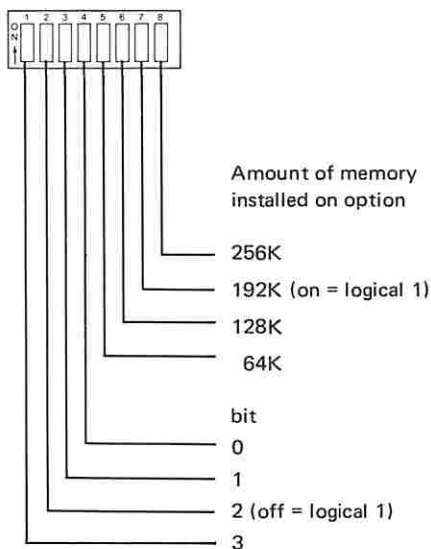
Starting Address = xxxK

64K $\sqrt{\text{xxxK}}$ = Decimal value

Convert decimal value to binary

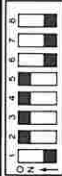
Bit 3 2 1 0
 Bit value . . . 8 4 2 1

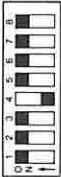
Switch



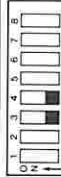
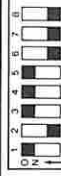
System Board and Memory Option Switch Settings

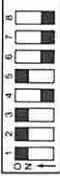
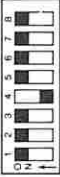
The following system board and memory option switch settings are divided into two groups, the first for the 16/64K system board, and the second for the 64/256K system board. The 16/64K and 64/256K system boards must be populated to their respective maximum of 64K or 256K of RAM before any memory expansion options can be installed.

System Board Switches	Switch 1	Switch 2
		

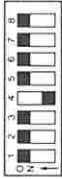
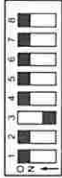
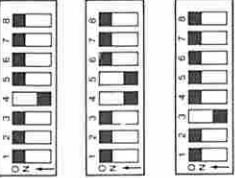
64/256K Options	64K Options	32K Options
1 - 32K option		

96K Total Memory; 32K +(64K on System Board)

System Board Switches	Switch 1	Switch 2
		

64/256K Options	64K Options	32K Options
1 - 64/256K option with 64K installed		
1 - 64K option		
2 - 32K options		

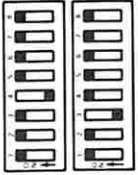
128K Total Memory; 64K +(64K on System Board)

System Board Switches	Switch 1	Switch 2
1 - 64/256K option with 64K installed 1 - 32K option		32K Options
1 - 64K option 1 - 32K option		
3 - 32K options		

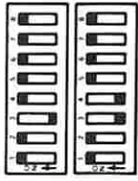
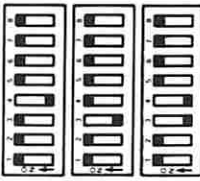
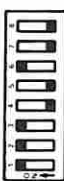
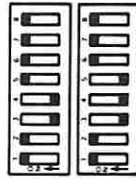
160K Total Memory; 96K + (64K on System Board)

System Board Switches	Switch 1	Switch 2
		
	64-/256K Options	64K Options
1-64-/256K option with 64K installed 1-64K option		
2-64K options		 
1-64-/256K option with 64K installed 2-32K options		 
1-64K option 2-32K options		 
1-64-/256K option with 128K installed		

192K Total Memory; 128K +(64K on System Board)

System Board Switches	Switch 1	Switch 2	
	64/256K Options	54K Options	32K Options
1-64/256K option with 64K installed 1-64K option 1-32K option			
2-64K options 1-32K option			
1-64/256K option with 128K installed 1-32K option			

224K Total Memory; 160K +(64K on System Board)

System Board Switches	Switch 1	Switch 2	
			
	64/256K Options	64K Options	32K Options
1-64/256K option with 192K installed			
1-64/256K option with 128K installed 1-64K option			
1-64/256K option with 64K installed 2-64K options			
3-64K options			
1-64/256K option with 128K installed 2-32K options			

256K Total Memory; 192K +(64K on System Board)

Hardware

1-174 Memory Expansion Options

System Board Switches	Switch 1	Switch 2
-----------------------	----------	----------

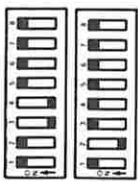
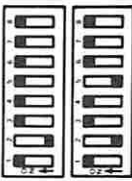


Switch 1

Switch 2

	64 /256K Options	64K Options	32K Options
1-64/256K option with 192K installed 1-32K option			
1-64/256K option with 128K installed 1-64K option 1-32K option			

288K Total Memory: 224 +(64K on System Board)

System Board Switches	Switch 1	Switch 2	
	64/256K Options	64K Options	32K Options
1-64/256K option with 128K installed 2-64K options			
1-64/256K option with 192K installed 1-64K option			
1-64/256K option with 192K installed 2-32K options			
1-64/256K option with 256K installed			

320K Total Memory; 256K + (64K on System Board)

System Board Switches	Switch 1	Switch 2
1-64/256K option with 192K installed 1-64K option 1-32K option		
1-64/256K option with 256K installed 1-32K option		
	64K Options	32K Options
		

352K Total Memory; 288K + (64K on System Board)

System Board Switches	Switch 1	Switch 2	
	64/256K Options	64K Options	32K Options
1-64/256K option with 192K installed 2-64K options			
1-64/256K option with 256K installed 1-64/256K option with 64K installed			
1-64/256K option with 256K installed 1-64K option			
1-64/256K option with 256K installed 2-32K options			

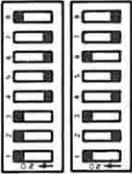
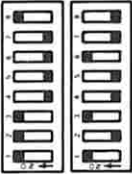
384K Total Memory; 320K +(64K on System Board)

System Board Switches	Switch 1	Switch 2
	64/256K Options	64K Options
1-64/256K option with 256K installed 1-64/256K option with 64K installed 1-32K option	 	
1-64/256K option with 256K installed 1-64K option 1-32K option		

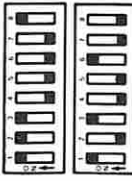
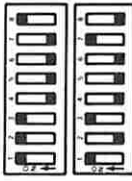
416K Total Memory: 352K +(64K on System Board)

System Board Switches	Switch 1	Switch 2	
	64/256K Options	64K Options	32K Options
1-64/256K option with 256K installed 1-64/256K option with 64K installed 1-64K option			
1-64/256K option with 256K installed 2-64K options			
1-64/256K option with 256K installed 1-64/256K option with 128K installed			

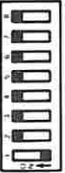
448K Total Memory; 384K +(64K on System Board)

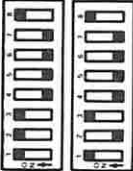
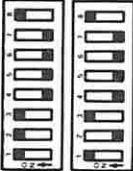
System Board Switches	Switch 1	Switch 2
1-64/256K option with 256K installed 1-64/256K option with 128K installed 1-32K option		
	64/256K Options	64K Options
		

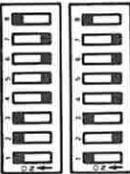
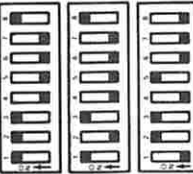
480K Total Memory; 416K +(64K on System Board)

System Board Switches	Switch 1	Switch 2
		
	64/256K Options	64K Options
1-64/256K option with 256K installed 1-64/256K option with 128K installed 1-64K option		
1-64/256K option with 256K installed 1-64/256K option with 192K installed		

512K Total Memory; 448K +(64K on System Board)

System Board Switches	Switch 1	Switch 2
		
	64/256K Options	64K Options
1-64/256K option with 256K installed		
1-64K/256K option with 192K installed		
1-32K option		
544K Total Memory; 480K +(64K on System Board)		
System Board Switches	Switch 1	Switch 2
		
	64/256K Options	64K Options
1-64/256K option with 256K installed		
1-64/256K option with 192K installed		
1-64K option		
2-64/256K options with 256 installed	 	
576K Total Memory; 512K +(64K on System Board)		

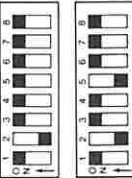
System Board Switches	Switch 1	Switch 2
2-64/256K options with 256K installed 1-32K option		
608K Total Memory; 544K + (64K on System Board)		

System Board Switches	Switch 1	Switch 2	
	64/256K Options	64K Options	32K Options
2-64/256K options with 256K installed 1-64K option			
2-64/256K options with 256K installed 1-64/256K option with 64K installed			

640K Total Memory; 576K +(64K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
1 - 32K option	64/256K Option Card Switches	32K Option Card Switches

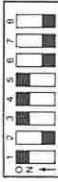
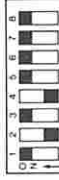
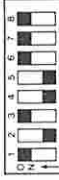
288K Total Memory; 32K + (256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
1 - 64/256K option with 64K installed		
1 - 64K option		
2 - 32K options		

320K Total Memory; 64K +(256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
1 - 64/256K option with 64K installed 1 - 32K option		
1 - 64K option 1 - 32K option		
3 - 32K options		

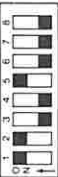
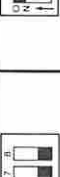
352K Total Memory; 96K +(256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
1 - 64/256K option with 64K installed 1 - 64K option		
2 - 64K options	  	  
1 - 64/256K option with 64K installed 2 - 32K options		 
1 - 64K option 2 - 32K options		 
1 - 64/256K option with 128K installed		

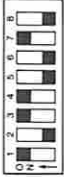
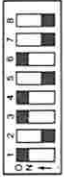
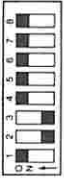
384K Total Memory; 128K + (256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
1 - 64/256K option with 64K installed 1 - 64K option 1 - 32K option		
2 - 64K options 1 - 32K option	 	
1 - 64/256K option with 128K installed 1 - 32K option		

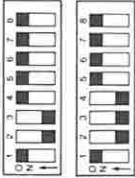
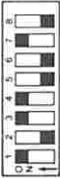
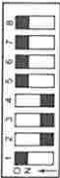
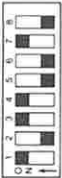
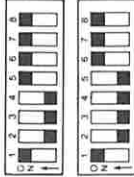
416K Total Memory; 160K +(256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
1 - 64/256K option with 192K installed		
1 - 64/256K option with 128K installed 1 - 64K option		
1 - 64/256K option with 64K installed 2 - 64K options		
3 - 64K options		
1 - 64/256K option with 128K installed 2 - 32K options		

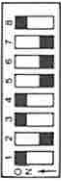
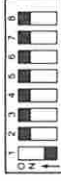
448K Total Memory; 192K +(256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
1 - 64/256K option with 192K installed 1 - 32K option		64K Option Card Switches
1 - 64/256K option with 128K installed 1 - 64K option 1 - 32K option		

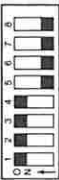
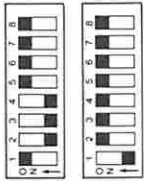
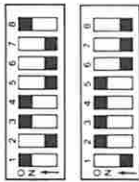
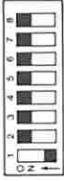
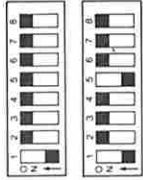
480K Total Memory; 224K +(256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		
1 - 64/256K option with 128K installed 2 - 64K options		
1 - 64/256K option with 192K installed 1 - 64K option		
1 - 64/256K option with 192K installed 2 - 32K options		
1 - 64/256K option with 256K installed		

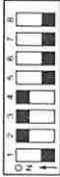
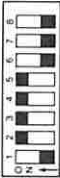
512K Total Memory; 256K + (256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
1 - 64/256K option with 192K installed 1 - 64K option 1 - 32K option		
1 - 64/256K option with 256K installed 1 - 32K option		

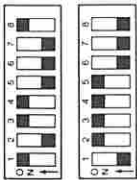
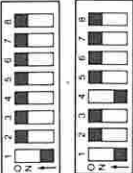
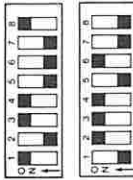
544K Total Memory; 288K +(256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		
1 - 64/256K option with 192K installed 2 - 64K options		
1 - 64/256K option with 256K installed 1 - 64/256K option with 64K installed		
1 - 64/256K option with 256K installed 1 - 64K option		
1 - 64/256K option with 256K installed 2 - 32K options		

576K Total Memory; 320K +(256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
1 - 64/256K option with 256K installed 1 - 64/256K option with 64K installed 1 - 32K option		
	64/256K Option Card Switches	32K Option Card Switches
	 	
1 - 64/256K option with 256K installed 1 - 64K option 1 - 32K option		

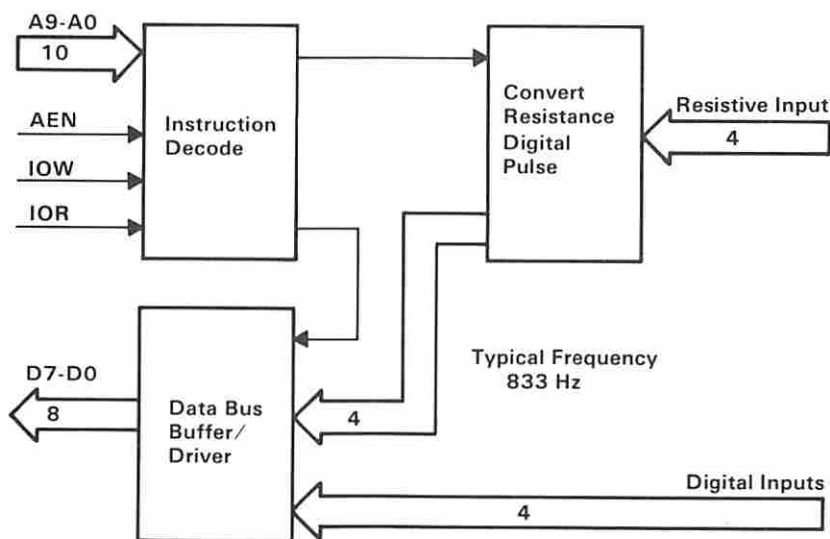
608K Total Memory; 352K + (256K on System Board)

System Board Switches	Switch Block 1	Switch Block 2
		
	64/256K Option Card Switches	61K Option Card Switches
1 - 64/256K option with 256K installed 1 - 64/256K option with 64K installed 1 - 64K option		
1 - 64/256K option with 256K installed 2 - 64K options		
1 - 64/256K option with 256K installed 1 - 64/256K option with 128K installed		

640K Total Memory; 384K +(256K on System Board)

IBM Game Control Adapter

The game control adapter allows up to four paddles or two joy sticks to be attached to the system. This card fits into one of the system board's or expansion board's expansion slots. The game control interface cable attaches to the rear of the adapter. In addition, four inputs for switches are provided. Paddle and joy stick positions are determined by changing resistive values sent to the adapter. The adapter plus system software converts the present resistive value to a relative paddle or joy stick position. On receipt of an output signal, four timing circuits are started. By determining the time required for the circuit to time-out (a function of the resistance), the paddle position can be determined. This adapter could be used as a general purpose I/O card with four analog (resistive) inputs plus four digital input points.



Game Control Adapter Block Diagram

Functional Description

Address Decode

The select on the game control adapter is generated by two 74LS138s as an address decoder. AEN must be inactive while the address is hex 201 in order to generate the select. The select allows a write to fire the one-shots or a read to give the values of the trigger buttons and one-shot outputs.

Data Bus Buffer/Driver

The data bus is buffered by a 74LS244 buffer/driver. For an In from address hex 201, the game control adapter will drive the data bus; at all other times, the buffer is left in the high impedance state.

Trigger Buttons

The trigger button inputs are read by an In from address hex 201. A trigger button is on each joy stick or paddle. These values are seen on data bits 7 through 4. These buttons default to an open state and are read as "1." When a button is pressed, it is read as "0." Software should be aware that these buttons are not debounced in hardware.

Joy Stick Positions

The joy stick position is indicated by a potentiometer for each coordinate. Each potentiometer has a range from 0 to 100 k-ohms that varies the time constant for each of the four one-shots. As this time constant is set at different values, the output of the one-shot will be of varying durations.

All four one-shots are fired at once by an Out to address hex 201. All four one-shot outputs will go true after the fire pulse and will remain high for varying times depending on where each potentiometer is set.

These four one-shot outputs are read by an In from address hex 201 and are seen on data bits 3 through 0.

I/O Channel Description

A9-A0:	Address lines 9 through 0 are used to address the game control adapter.
D7-D0:	Data lines 7 through 0 are the data bus.
IOR, IOW:	I/O read and I/O write are used when reading from or writing to an adapter (In, Out).
AEN:	When active, the adapter must be inactive and the data bus driver inactive.
+5 Vdc:	Power for the game control adapter.
GND:	Common ground.
A19-A10:	Unused.
MEMR, MEMW:	Unused.
DACK0-DACK3:	Unused.
IRQ7-IRQ2:	Unused.
DRQ3-DRQ1:	Unused.
ALE, T/C:	Unused.
CLK, OSC:	Unused.
I/O CH CK	Unused.
I/O CH RDY:	Unused.
RESET DRV:	Unused.
-5 Vdc, +12 Vdc, -12 Vdc:	Unused.

Interface Description

The game control adapter has eight input lines, four of which are digital inputs and 4 of which are resistive inputs. The inputs are read with one In from address hex 201.

The four digital inputs each have a 1 k-ohm pullup resistor to +5 Vdc. With no drives on these inputs, a 1 is read. For a 0 reading, the inputs must be pulled to ground.

The four resistive pullups, measured to +5 Vdc, will be converted to a digital pulse with a duration proportional to the resistive load, according to the following equation:

$$\text{Time} = 24.2 \mu\text{sec} + 0.011 (r) \mu\text{sec}$$

The user must first begin the conversation by an Out to address hex 201. An In from address hex 201 will show the digital pulse go high and remain high for the duration according to the resistance value. All four bits (bit 3-bit 0) function in the same manner; their digital pulse will all go high simultaneously and will reset independently according to the input resistance value.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Digital Inputs				Resistive Inputs			

The typical input to the game control adapter is a set of joy sticks or game paddles.

The joy sticks will typically be a set of two (A and B). These will have one or two buttons each with two variable resistances each, with a range from 0 to 100 k-ohms. One variable resistance will indicate the X-coordinate and the other variable resistance will indicate the Y-coordinate. This should be attached to give the following input data:

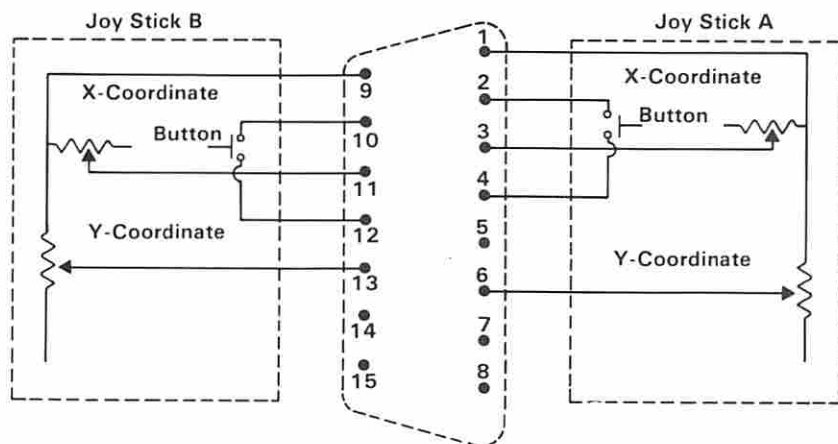
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
B-#2	B-#1	A-#2	A-#1	B-Y	B-X	A-Y	A-X
Button	Button	Button	Button	Coordinate	Coordinate	Coordinate	Coordinate

The game paddles will have a set of two (A and B) or four (A, B, C, and D) paddles. These will have one button each and one variable resistance each, with a range from 0 to 100 k-ohms. This should be attached to give the following input data:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
D	C	B	A	D	C	B	A
Button	Button	Button	Button	Coordinate	Coordinate	Coordinate	Coordinate

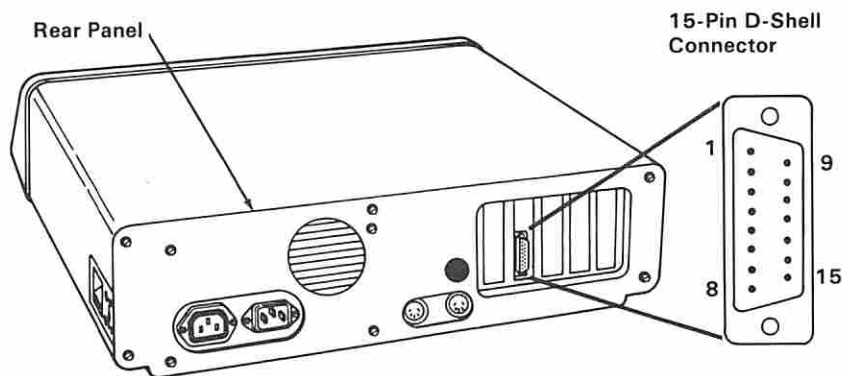
Refer to "Joy Stick Schematic Diagram" for attaching game controllers.

15-Pin Male D-Shell Connector

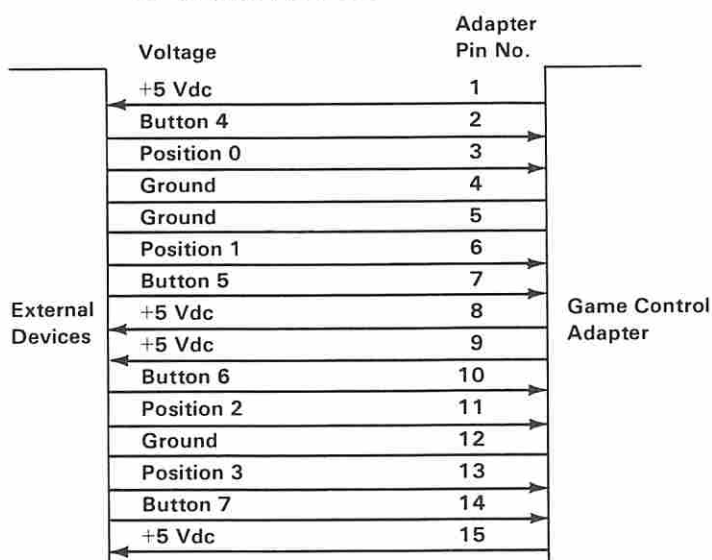


Note: Potentiometer for X- and Y-Coordinates has a range of 0 to 100 k-ohms. Button is normally open; closed when pressed.

Joy Stick Schematic Diagram



At Standard TTL Levels



Connector Specifications

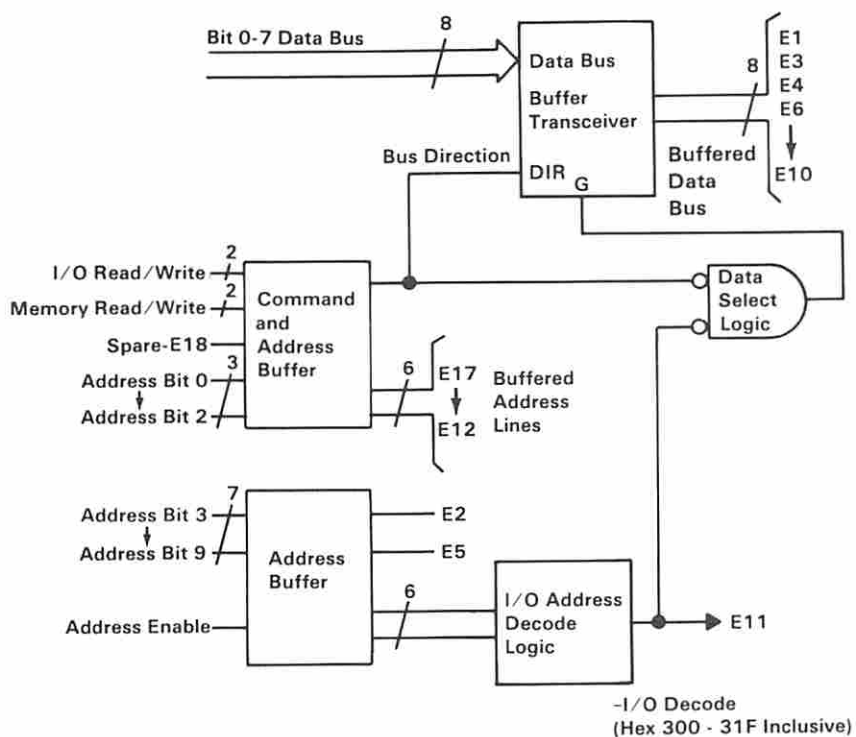
IBM Prototype Card

The prototype card is 4.2 inches (106.7 millimeters) high by 13.2 inches (335.3 millimeters) long and plugs into an expansion unit or system unit expansion slot. All system control signals and voltage requirements are provided through a 2 by 31 position card-edge tab.

The card contains a voltage bus (+5 Vdc) and a ground bus (0 Vdc). Each bus borders the card, with the voltage bus on the back (pin side) and the ground bus on the front (component side). A system interface design is also provided on the prototype card.

The prototype card can also accommodate a D-shell connector if it is needed. The connector size can range from a 9 to a 37 position connector.

Note: Install all components on the component side of the prototype card. The total width of the card including components should not exceed 0.500 inch (12.7 millimeters). If these specifications are not met, components on the prototype card may touch other cards plugged into adjacent slots.



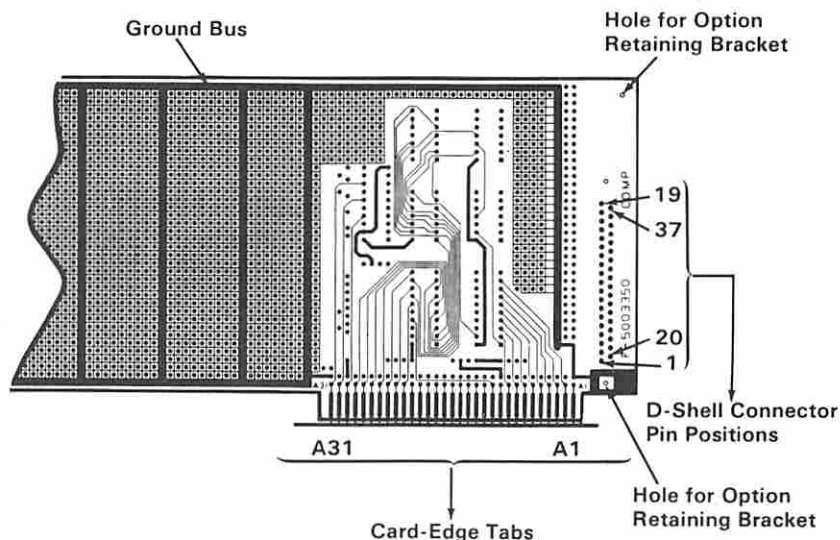
Prototype Card Block Diagram

I/O Channel Interface

The prototype card has two layers screened onto it (one on the front and one on the back). It also has 3,909 plated through-holes that are 0.040 inch (10.1 millimeters) in size and have a 0.060 inch (1.52 millimeters) pad, which is located on a 0.10 inch (2.54 millimeters) grid. There are 37 plated through-holes that are 0.048 inch (1.22 millimeters) in size. These holes are located at the rear of the card (viewed as if installed in the machine). These 37 holes are used for a 9 to 37 position D-shell connector. The card also has 5 holes that are 0.125 inch (3.18 millimeters) in size. One hole is located just above the two rows of D-shell connector holes, and the other four are located in the corners of the board (one in each corner).

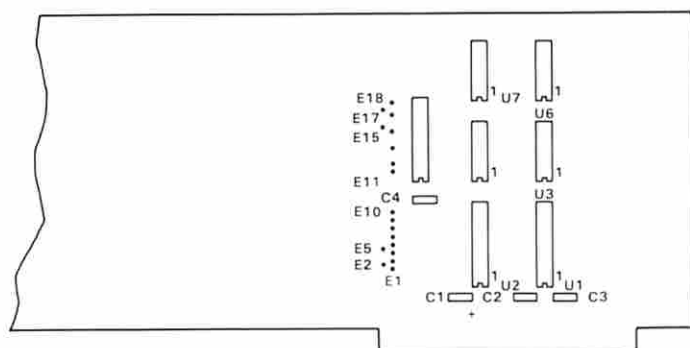
Prototype Card Layout

The component side has the ground bus [0.05 inch (1.27 millimeters) wide] screened on it and card-edge tabs that are labeled A1 through A31.



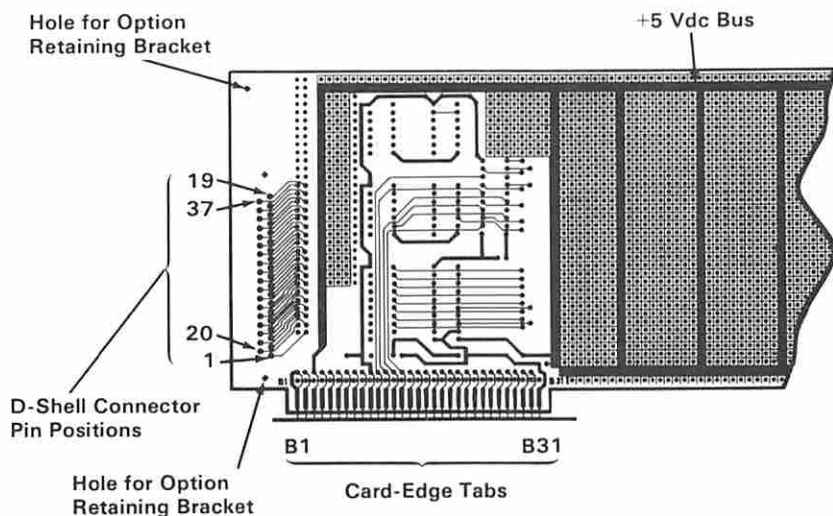
Component Side

The component side also has a silk screen printed on it that is used as a component guide for the I/O interface.



Component Side

The pin side has a +5 Vdc bus [0.05 inch (1.27 millimeters) wide] screened onto it and card-edge tabs that are labeled B1 through B31.



Pin Side

Each card-edge tab is connected to a plated through-hole by a 0.012-inch (0.3-millimeter) land. There are three ground tabs connected to the ground bus by three 0.012-inch (0.3-millimeter) lands. Also, there are two +5-Vdc tabs connected to the voltage bus by two 0.012-inch (0.3-millimeter) lands.

For additional interfacing information, refer to "I/O Channel Description" and "I/O Channel Diagram" in this manual. Also, the "Prototype Card Interface Logic Diagram" is in Appendix D of this manual. If the recommended interface logic is used, the list of TTL type numbers listed below will help you select the necessary components.

Component	TTL Number	Description
U1	74LS245	Octal Bus Transceiver
U2, U5	74LS244	Octal Buffers Line Driver/Line Receivers
U4	74LS04	Hex Inverters
U3	74LS08	Quadruple 2 - Input Positive - AND Gate
U6	74LS02	Quadruple 2 - Input Positive - NOR Gate
U7	74LS21	Dual 4 - Input Positive - AND Gate
C1		10.0 μ F Tantalum Capacitor
C2, C3, C4		0.047 μ F Ceramic Capacitor

System Loading and Power Limitations

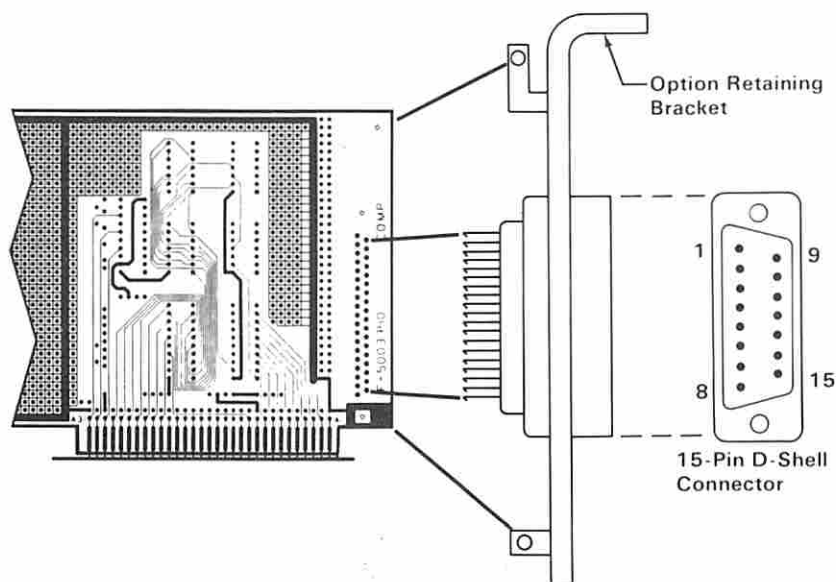
Because of the number of options that may be installed in the system, the I/O bus loading should be limited to one Schottky TTL load. If the interface circuitry on the card is used, then this requirement is met.

Refer to the power supply information in this manual for the power limitations to be observed.

Prototype Card External Interface

If a connector is required for the card function, then you should purchase one of the recommended connectors (manufactured by Amp) or equivalent listed below:

Connector Size	Part Number (Amp)
9-pin D-shell (Male)	205865-1
9-pin D-shell (Female)	205866-1
15-pin D-shell (Male)	205867-1
15-pin D-shell (Female)	205868-1
25-pin D-shell (Male)	205857-1
25-pin D-shell (Female)	205858-1
37-pin D-shell (Male)	205859-1
37-pin D-shell (Female)	205860-1



Component Side

IBM Asynchronous Communications Adapter

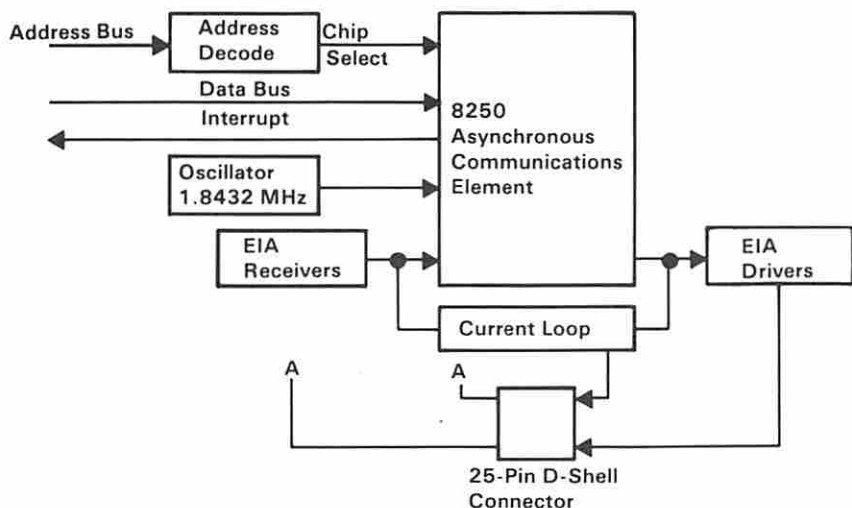
The asynchronous communications adapter system control signals and voltage requirements are provided through a 2 by 31 position card-edge tab. Two jumper modules are provided on the adapter. One jumper module selects either RS-232C or current-loop operation. The other jumper module selects one of two addresses for the adapter, so two adapters may be used in one system.

The adapter is fully programmable and supports asynchronous communications only. It will add and remove start bits, stop bits, and parity bits. A programmable baud rate generator allows operation from 50 baud to 9600 baud. Five, six, seven or eight bit characters with 1, 1-1/2, or 2 stop bits are supported. A fully prioritized interrupt system controls transmit, receive, error, line status and data set interrupts. Diagnostic capabilities provide loopback functions of transmit/receive and input/output signals.

The heart of the adapter is a INS8250 LSI chip or functional equivalent. Features in addition to those listed above are:

- Full double buffering eliminates need for precise synchronization.
- Independent receiver clock input.
- Modem control functions: clear to send (CTS), request to send (RTS), data set ready (DSR), data terminal ready (DTR), ring indicator (RI), and carrier detect.
- False-start bit detection.
- Line-break generation and detection.

All communications protocol is a function of the system microcode and must be loaded before the adapter is operational. All pacing of the interface and control signal status must be handled by the system software. The following figure is a block diagram of the asynchronous communications adapter.



Asynchronous Communications Adapter Block Diagram

Modes of Operation

The different modes of operation are selected by programming the 8250 asynchronous communications element. This is done by selecting the I/O address (hex 3F8 to 3FF primary, and hex 2F8 to 2FF secondary) and writing data out to the card. Address bits A0, A1, and A2 select the different registers that define the modes of operation. Also, the divisor latch access bit (bit 7) of the line control register is used to select certain registers.

I/O Decode (in Hex)		Register Selected	DLAB State
Primary Adapter	Alternate Adapter		
3F8	2F8	TX Buffer	DLAB=0 (Write)
3F8	2F8	RX Buffer	DLAB=0 (Read)
3F8	2F8	Divisor Latch LSB	DLAB=1
3F9	2F9	Divisor Latch MSB	DLAB=1
3F9	2F9	Interrupt Enable Register	
3FA	2FA	Interrupt Identification Registers	
3FB	2FB	Line Control Register	
3FC	2FC	Modem Control Register	
3FD	2FD	Line Status Register	
3FE	2FE	Modem Status Register	

I/O Decodes

Hex Address 3F8 to 3FF and 2F8 to 2FF											DLAB	Register
A9	A8	A7	A6	A5	A4	A3	A2	A1	A0			
1	1/0	1	1	1	1	1	x	x	x			
							0	0	0	0		Receive Buffer (read), Transmit Holding Reg. (write)
							0	0	1	0		Interrupt Enable
							0	1	0	x		Interrupt Identification
							0	1	1	x		Line Control
							1	0	0	x		Modem Control
							1	0	1	x		Line Status
							1	1	0	x		Modem Status
							1	1	1	x		None
							0	0	0	1		Divisor Latch (LSB)
							0	0	1	1		Divisor Latch (MSB)

Note: Bit 8 will be logical 1 for the adapter designated as primary or a logical 0 for the adapter designated as alternate (as defined by the address jumper module on the adapter).

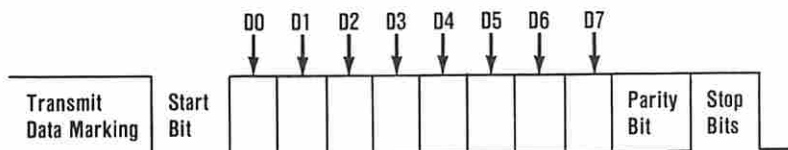
A2, A1 and A0 bits are "don't cares" and are used to select the different register of the communications chip.

Address Bits

Interrupts

One interrupt line is provided to the system. This interrupt is IRQ4 for a primary adapter or IRQ3 for an alternate adapter, and is positive active. To allow the communications card to send interrupts to the system, bit 3 of the modem control register must be set to 1 (high). At this point, any interrupts allowed by the interrupt enable register will cause an interrupt.

The data format will be as follows:



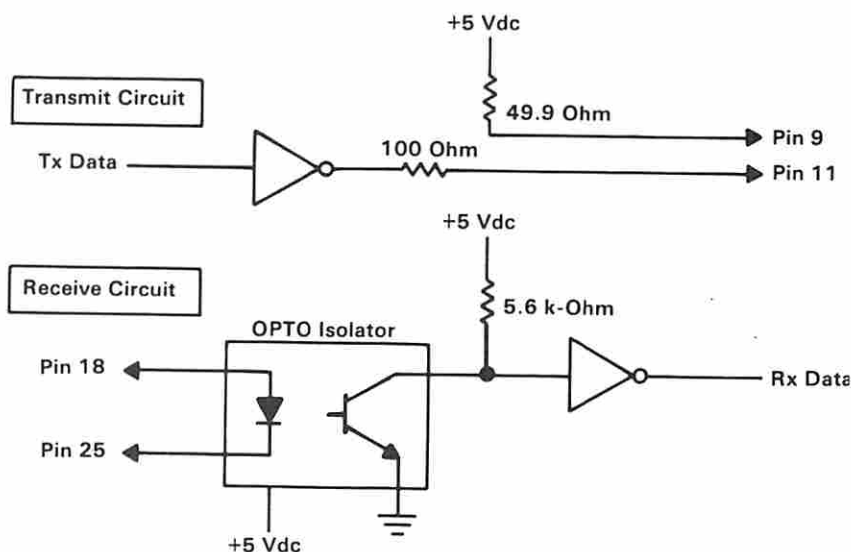
Data bit 0 is the first bit to be transmitted or received. The adapter automatically inserts the start bit, the correct parity bit if programmed to do so, and the stop bit (1, 1-1/2, or 2 depending on the command in the line-control register).

Interface Description

The communications adapter provides an EIA RS-232C-like interface. One 25-pin D-shell, male type connector is provided to attach various peripheral devices. In addition, a current loop interface is also located in this same connector. A jumper block is provided to manually select either the voltage interface, or the current loop interface.

The current loop interface is provided to attach certain printers provided by IBM that use this particular type of interface.

- Pin 18 + receive current loop data
- Pin 25 - receive current loop return
- Pin 9 + transmit current loop return
- Pin 11 - transmit current loop data



Current Loop Interface

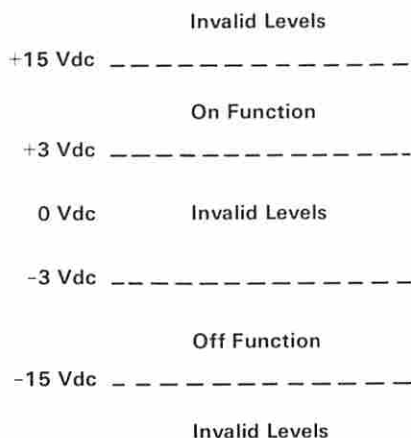
The voltage interface is a serial interface. It supports certain data and control signals, as listed below.

Pin 2	Transmitted Data
Pin 3	Received Data
Pin 4	Request to Send
Pin 5	Clear to Send
Pin 6	Data Set Ready
Pin 7	Signal Ground
Pin 8	Carrier Detect
Pin 20	Data Terminal Ready
Pin 22	Ring Indicator

The adapter converts these signals to/from TTL levels to EIA voltage levels. These signals are sampled or generated by the communications control chip. These signals can then be sensed by the system software to determine the state of the interface or peripheral device.

Voltage Interchange Information

Interchange Voltage	Binary State	Signal Condition	Interface Control Function
Positive Voltage =	Binary (0)	= Spacing	=On
Negative Voltage =	Binary (1)	= Marking	=Off



The signal will be considered in the "marking" condition when the voltage on the interchange circuit, measured at the interface point, is more negative than -3 Vdc with respect to signal ground. The signal will be considered in the "spacing" condition when the voltage is more positive than +3 Vdc with respect to signal ground. The region between +3 Vdc and -3 Vdc is defined as the transition region, and considered an invalid level. The voltage that is more negative than -15 Vdc or more positive than +15 Vdc will also be considered an invalid level.

During the transmission of data, the "marking" condition will be used to denote the binary state "1" and "spacing" condition will be used to denote the binary state "0."

For interface control circuits, the function is "on" when the voltage is more positive than +3 Vdc with respect to signal ground and is "off" when the voltage is more negative than -3 Vdc with respect to signal ground.

INS8250 Functional Pin Description

The following describes the function of all INS8250 input/output pins. Some of these descriptions reference internal circuits.

Note: In the following descriptions, a low represents a logical 0 (0 Vdc nominal) and a high represents a logical 1 (+2.4 Vdc nominal).

Input Signals

Chip Select ($\overline{CS0}$, $\overline{CS1}$, $\overline{CS2}$), Pins 12-14: When $\overline{CS0}$ and $\overline{CS1}$ are high and $\overline{CS2}$ is low, the chip is selected. Chip selection is complete when the decoded chip select signal is latched with an active (low) address strobe ($\overline{ADS}$) input. This enables communications between the INS8250 and the processor.

Data Input Strobe ($\overline{DISTR}$, $\overline{DISTR}$) Pins 22 and 21: When $\overline{DISTR}$ is high or $\overline{DISTR}$ is low while the chip is selected, allows the processor to read status information or data from a selected register of the INS8250.

Note: Only an active $\overline{DISTR}$ or $\overline{DISTR}$ input is required to transfer data from the INS8250 during a read operation. Therefore, tie either the $\overline{DISTR}$ input permanently low or the $\overline{DISTR}$ input permanently high, if not used.

Data Output Strobe ($\overline{DOSTR}$, $\overline{DOSTR}$), Pins 19 and 18: When $\overline{DOSTR}$ is high or $\overline{DOSTR}$ is low while the chip is selected, allows the processor to write data or control words into a selected register of the INS8250.

Note: Only an active $\overline{DOSTR}$ or $\overline{DOSTR}$ input is required to transfer data to the INS8250 during a write operation. Therefore, tie either the $\overline{DOSTR}$ input permanently low or the $\overline{DOSTR}$ input permanently high, if not used.

Address Strobe ($\overline{ADS}$), Pin 25: When low, provides latching for the register select ($A0$, $A1$, $A2$) and chip select ($\overline{CS0}$, $\overline{CS1}$, $\overline{CS2}$) signals.

Note: An active $\overline{ADS}$ input is required when the register select ($A0$, $A1$, $A2$) signals are not stable for the duration of a read or write operation. If not required, tie the $\overline{ADS}$ input permanently low.

Register Select (A0, A1, A2), Pins 26-28: These three inputs are used during a read or write operation to select an INS8250 register to read or write to as indicated in the table below. Note that the state of the divisor latch access bit (DLAB), which is the most significant bit of the line control register, effects the selection of certain INS8250 registers. The DLAB must be set high by the system software to access the baud generator divisor latches.

DLAB	A2	A1	A0	Register
0	0	0	0	Receiver Buffer (Read), Transmitter Holding Register (Write)
0	0	0	1	Interrupt Enable
X	0	1	0	Interrupt Identification (Read Only)
X	0	1	1	Line Control
X	1	0	0	Modem Control
X	1	0	1	Line Status
X	1	1	0	Modem Control Status
X	1	1	1	None
1	0	0	0	Divisor Latch (Least Significant Bit)
1	0	0	1	Divisor Latch (Most Significant Bit)

Master Reset (MR), Pin 35: When high, clears all the registers (except the receiver buffer, transmitter holding, and divisor latches), and the control logic of the INS8250. Also, the state of various output signals ($\overline{\text{SOUT}}$, $\overline{\text{INTRPT}}$, $\overline{\text{OUT 1}}$, $\overline{\text{OUT 2}}$, $\overline{\text{RTS}}$, $\overline{\text{DIR}}$) are affected by an active MR input. Refer to the "Asynchronous Communications Reset Functions" table.

Receiver Clock (RCLK), Pin 9: This input is the 16 x baud rate clock for the receiver section of the chip.

Serial Input (SIN), Pin 10: Serial data input from the communications link (peripheral device, modem, or data set).

Clear to Send ($\overline{\text{CTS}}$), Pin 36: The $\overline{\text{CTS}}$ signal is a modem control function input whose condition can be tested by the processor by reading bit 4 (CTS) of the modem status register. Bit 0 (DCTS) of the modem status register indicates whether the CTS input has changed state since the previous reading of the modem status register.

Note: Whenever the CTS bit of the modem status register changes state, an interrupt is generated if the modem status interrupt is enabled.

Data Set Ready ($\overline{\text{DSR}}$), Pin 37: When low, indicates that the modem or data set is ready to establish the communications link and transfer data with the INS8250. The $\overline{\text{DSR}}$ signal is a modem-control function input whose condition can be tested by the processor by reading bit 5 (DSR) of the modem status register. Bit 1 (DDSR) of the modem status register indicates whether the $\overline{\text{DSR}}$ input has changed since the previous reading of the modem status register.

Note: Whenever the DSR bit of the modem status register changes state, an interrupt is generated if the modem status interrupt is enabled.

Received Line Signal Detect ($\overline{\text{RLSD}}$), Pin 38: When low, indicates that the data carrier had been detected by the modem or data set. The $\overline{\text{RLSD}}$ signal is a modem-control function input whose condition can be tested by the processor by reading bit 7 (RLSD) of the modem status register. Bit 3 (DRLSD) of the modem status register indicates whether the $\overline{\text{RLSD}}$ input has changed state since the previous reading of the modem status register.

Note: Whenever the RLSD bit of the modem status register changes state, an interrupt is generated if the modem status interrupt is enabled.

Ring Indicator ($\overline{RI}$), Pin 39: When low, indicates that a telephone ringing signal has been received by the modem or data set. The $\overline{RI}$ signal is a modem-control function input whose condition can be tested by the processor by reading bit 6 (RI) of the modem status register. Bit 2 (TERI) of the modem status register indicates whether the $\overline{RI}$ input has changed from a low to high state since the previous reading of the modem status register.

Note: Whenever the RI bit of the modem status register changes from a high to a low state, an interrupt is generated if the modem status interrupt is enabled.

VCC, Pin 40: +5 Vdc supply.

VSS, Pin 20: Ground (0 Vdc) reference.

Output Signals

Data Terminal Ready ($\overline{DTR}$), Pin 33: When low, informs the modem or data set that the INS8250 is ready to communicate. The DTR output signal can be set to an active low by programming bit 0 (DTR) of the modem control register to a high level. The $\overline{DTR}$ signal is set high upon a master reset operation.

Request to Send ($\overline{RTS}$), Pin 32: When low, informs the modem or data set that the INS8250 is ready to transmit data. The $\overline{RTS}$ output signal can be set to an active low by programming bit 1 (RTS) of the modem control register. The $\overline{RTS}$ signal is set high upon a master reset operation.

Output 1 ($\overline{OUT\ 1}$), Pin 34: User-designated output that can be set to an active low by programming bit 2 ($\overline{OUT\ 1}$) of the modem control register to a high level. The $\overline{OUT\ 1}$ signal is set high upon a master reset operation.

Output 2 ($\overline{OUT\ 2}$), Pin 31: User-designated output that can be set to an active low by programming bit 3 ($\overline{OUT\ 2}$) of the modem control register to a high level. The $\overline{OUT\ 2}$ signal is set high upon a master reset operation.

Chip Select Out (CSOUT), Pin 24: When high, indicates that the chip has been selected by active CS0, CS1, and CS2 inputs. No data transfer can be initiated until the CSOUT signal is a logical 1.

Driver Disable (DDIS), Pin 23: Goes low whenever the processor is reading data from the INS8250. A high-level DDIS output can be used to disable an external transceiver (if used between the processor and INS8250 on the D7-D0 data bus) at all times, except when the processor is reading data.

Baud Out (BAUDOUT), Pin 15: 16 x clock signal for the transmitter section of the INS8250. The clock rate is equal to the main reference oscillator frequency divided by the specified divisor in the baud generator divisor latches. The BAUDOUT may also be used for the receiver section by typing this output to the RCLK input of the chip.

Interrupt (INTRPT), Pin 30: Goes high whenever any one of the following interrupt types has an active high condition and is enabled through the IER: receiver error flag, received data available, transmitter holding register empty, or modem status. The INTRPT Signal is reset low upon the appropriate interrupt service or a master reset operation.

Serial Output (SOUT), Pin 11: Composite serial data output to the communications link (peripheral, modem, or data set). The SOUT signal is set to the marking (logical 1) state upon a master reset operation.

Input/Output Signals

Data (D7-D0) Bus, Pins 1-8: This bus comprises eight tri-state input/output lines. The bus provides bidirectional communications between the INS8250 and the processor. Data, control words, and status information are transferred via the D7-D0 Data bus.

External Clock Input/Output (XTAL1, XTAL2), Pins 16 and 17: These two pins connect the main timing reference (crystal or signal clock) to the INS8250.

Programming Considerations

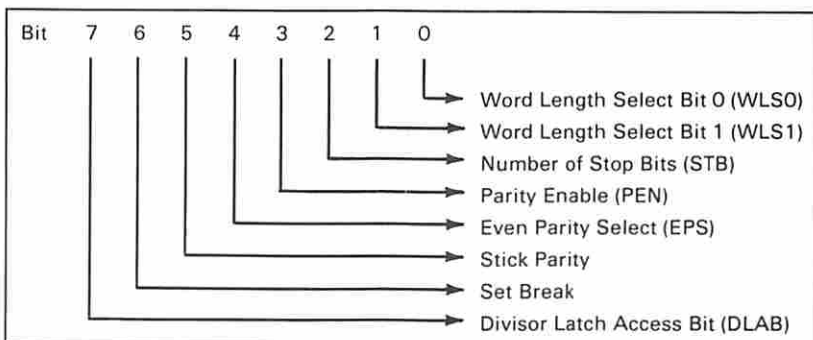
The INS8250 has a number of accessible registers. The system programmer may access or control any of the INS825 registers through the processor. These registers are used to control INS8250 operations and to transmit and receive data. A table listing and description of the accessible registers follows.

Register/Signal	Reset Control	Reset State
Interrupt Enable Register	Master Reset	All Bits Low (0-3 Forced and 4-7 Permanent)
Interrupt Identification Register	Master Reset	Bit 0 is High, Bits 1 and 2 Low Bits 3-7 are Permanently Low
Line Control Register	Master Reset	All Bits Low
Modem Control Register	Master Reset	All Bits Low
Line Status Register	Master Reset	Except Bits 5 and 6 are High
Modem Status Register	Master Reset	Bits 0-3 Low Bits 4-7 - Input Signal
SOUT	Master Reset	High
INTRPT (RCVR Errors)	Read LSR/MR	Low
INTRPT (RCVR Data Ready)	Read RBR/MR	Low
INTRPT (RCVR Data Ready)	Read IIR/Write THR/MR	Low
INTRPT (Modem Status Changes)	Read MSR/MR	Low
OUT 2	Master Reset	High
RTS	Master Reset	High
DTR	Master Reset	High
OUT 1	Master Reset	High

Asynchronous Communications Reset Functions

Line-Control Register

The system programmer specifies the format of the asynchronous data communications exchange through the line-control register. In addition to controlling the format, the programmer may retrieve the contents of the line-control register for inspection. This feature simplifies system programming and eliminates the need for separate storage in system memory of the line characteristics. The contents of the line-control register are indicated and described below.



Line-Control Register (LCR)

Bits 0 and 1: These two bits specify the number of bits in each transmitted or received serial character. The encoding of bits 0 and 1 is as follows:

Bit 1	Bit 0	Word Length
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

Bit 2: This bit specifies the number of stop bits in each transmitted or received serial character. If bit 2 is a logical 0, one stop bit is generated or checked in the transmit or receive data, respectively. If bit 2 is logical 1 when a 5-bit word length is selected through bits 0 and 1, 1-1/2 stop bits are generated or checked. If bit 2 is logical 1 when either a 6-, 7-, or 8-bit word length is selected, two stop bits are generated or checked.

Bit 3: This bit is the parity enable bit. When bit 3 is a logical 1, a parity bit is generated (transmit data) or checked (receive data) between the last data word bit and stop bit of the serial data. (The parity bit is used to produce an even or odd number of 1's when the data word bits and the parity bit are summed.)

Bit 4: This bit is the even parity select bit. When bit 3 is a logical 1 and bit 4 is a logical 0, an odd number of logical 1's is transmitted or checked in the data word bits and parity bit. When bit 3 is a logical 1 and bit 4 is a logical 1, an even number of bits is transmitted or checked.

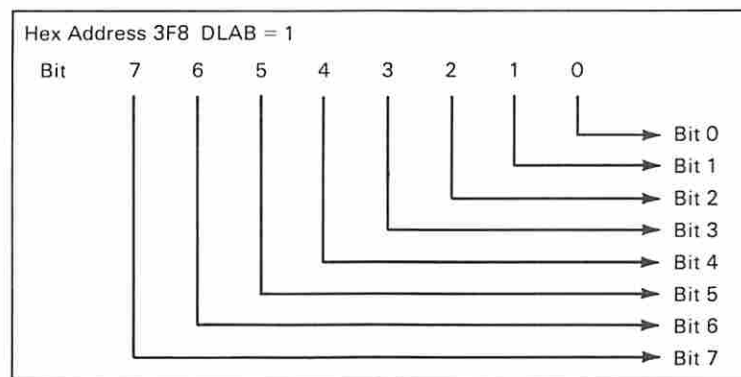
Bit 5: This bit is the stick parity bit. When bit 3 is a logical 1 and bit 5 is a logical 1, the parity bit is transmitted and then detected by the receiver as a logical 0 if bit 4 is a logical 1, or as a logical 1 if bit 4 is a logical 0.

Bit 6: This bit is the set break control bit. When bit 6 is a logical 1, the serial output (SOUT) is forced to the spacing (logical 0) state and remains there regardless of other transmitter activity. The set break is disabled by setting bit 6 to a logical 0. This feature enables the processor to alert a terminal in a computer communications system.

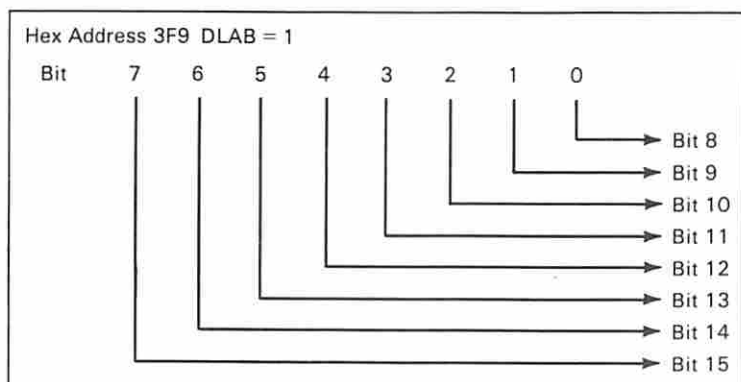
Bit 7: This bit is the divisor latch access bit (DLAB). It must be set high (logical 1) to access the divisor latches of the baud rate generator during a read or write operation. It must be set low (logical 0) to access the receiver buffer, the transmitter holding register, or the interrupt enable register.

Programmable Baud Rate Generator

The INS8250 contains a programmable baud rate generator that is capable of taking the clock input (1.8432 MHz) and dividing it by any divisor from 1 to $(2^{16}-1)$. The output frequency of the baud generator is $16 \times$ the baud rate [divisor $\# = (\text{frequency input}) / (\text{baud rate} \times 16)$]. Two 8-bit latches store the divisor in a 16-bit binary format. These divisor latches must be loaded during initialization in order to ensure desired operation of the baud rate generator. Upon loading either of the divisor latches, a 16-bit baud counter is immediately loaded. This prevents long counts on initial load.



Divisor Latch Least Significant Bit (DLL)



Divisor Latch Most Significant Bit (DLM)

The following figure illustrates the use of the baud rate generator with a frequency of 1.8432 MHz. For baud rates of 9600 and below, the error obtained is minimal.

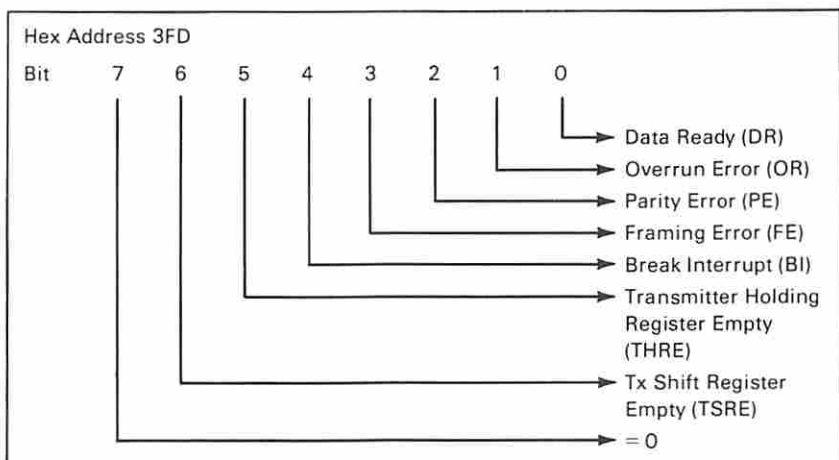
Note: The maximum operating frequency of the baud generator is 3.1 MHz. In no case should the data rate be greater than 9600 baud.

Desired Baud Rate	Divisor Used to Generate 16x Clock		Percent Error Difference Between Desired and Actual
	(Decimal)	(Hex)	
50	2304	900	—
75	1536	600	—
110	1047	417	0.026
134.5	857	359	0.058
150	768	300	—
300	384	180	—
600	192	0C0	—
1200	96	060	—
1800	64	040	—
2000	58	03A	0.69
2400	48	030	—
3600	32	020	—
4800	24	018	—
7200	16	010	—
9600	12	00C	—

Baud Rate at 1.843 MHz

Line Status Register

This 8-bit register provides status information on the processor concerning the data transfer. The contents of the line status register are indicated and described below:



Line Status Register (LSR)

Bit 0: This bit is the receiver data ready (DR) indicator. Bit 0 is set to a logical 1 whenever a complete incoming character has been received and transferred into the receiver buffer register. Bit 0 may be reset to a logical 0 either by the processor reading the data in the receiver buffer register or by writing a logical 0 into it from the processor.

Bit 1: This bit is the overrun error (OE) indicator. Bit 1 indicates that data in the receiver buffer register was not read by the processor before the next character was transferred into the receiver buffer register, thereby destroying the previous character. The OE indicator is reset whenever the processor reads the contents of the line status register.

Bit 2: This bit is the parity error (PE) indicator. Bit 2 indicates that the received data character does not have the correct even or odd parity, as selected by the even parity-select bit. The PE bit is set to a logical 1 upon detection of a parity error and is reset to a logical 0 whenever the processor reads the contents of the line status register.

Bit 3: This bit is the framing error (FE) indicator. Bit 3 indicates that the received character did not have a valid stop bit. Bit 3 is set to a logical 1 whenever the stop bit following the last data bit or parity is detected as a zero bit (spacing level).

Bit 4: This bit is the break interrupt (BI) indicator. Bit 4 is set to a logical 1 whenever the received data input is held in the spacing (logical 0) state for longer than a full word transmission time (that is, the total time of start bit + data bits + parity + stop bits).

Note: Bits 1 through 4 are the error conditions that produce a receiver line status interrupt whenever any of the corresponding conditions are detected.

Bit 5: This bit is the transmitter holding register empty (THRE) indicator. Bit 5 indicates that the INS8250 is ready to accept a new character for transmission. In addition, this bit causes the INS8250 to issue an interrupt to the processor when the transmit holding register empty interrupt enable is set high. The THRE bit is set to a logical 1 when a character is transferred from the transmitter holding register into the transmitter shift register. The bit is reset to logical 0 concurrently with the loading of the transmitter holding register by the processor.

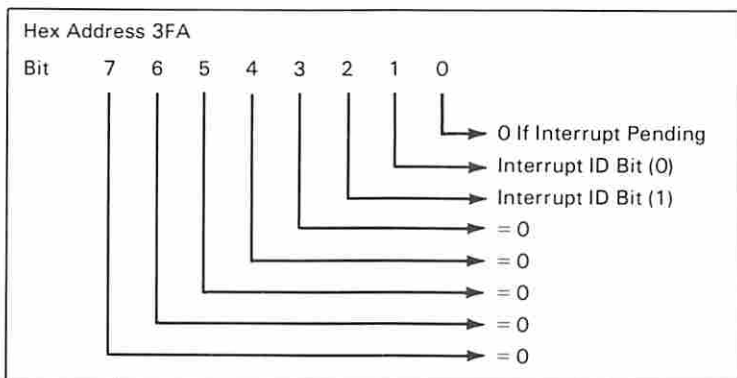
Bit 6: This bit is the transmitter shift register empty (TSRE) indicator. Bit 6 is set to a logical 1 whenever the transmitter shift register is idle. It is reset to logical 0 upon a data transfer from the transmitter holding register to the transmitter shift register. Bit 6 is a read-only bit.

Bit 7: This bit is permanently set to logical 0.

Interrupt Identification Register

The INS8250 has an on-chip interrupt capability that allows for complete flexibility in interfacing to all the popular microprocessors presently available. In order to provide minimum software overhead during data character transfers, the INS8250 prioritizes interrupts into four levels: receiver line status (priority 1), received data ready (priority 2), transmitter holding register empty (priority 3), and modem status (priority 4).

Information indicating that a prioritized interrupt is pending and the type of prioritized interrupt is stored in the interrupt identification register. Refer to the "Interrupt Control Functions" table. The interrupt identification register (IIR), when addressed during chip-select time, freezes the highest priority interrupt pending, and no other interrupts are acknowledged until that particular interrupt is serviced by the processor. The contents of the IIR are indicated and described below.



Interrupt Identification Register (IIR)

Bit 0: This bit can be used in either a hard-wired prioritized or polled environment to indicate whether an interrupt is pending and the IIR contents may be used as a pointer to the appropriate interrupt service routine. When bit 0 is a logical 1, no interrupt is pending and polling (if used) is continued.

Bits 1 and 2: These two bits of the IIR are used to identify the highest priority interrupt pending as indicated in the "Interrupt Control Functions" table.

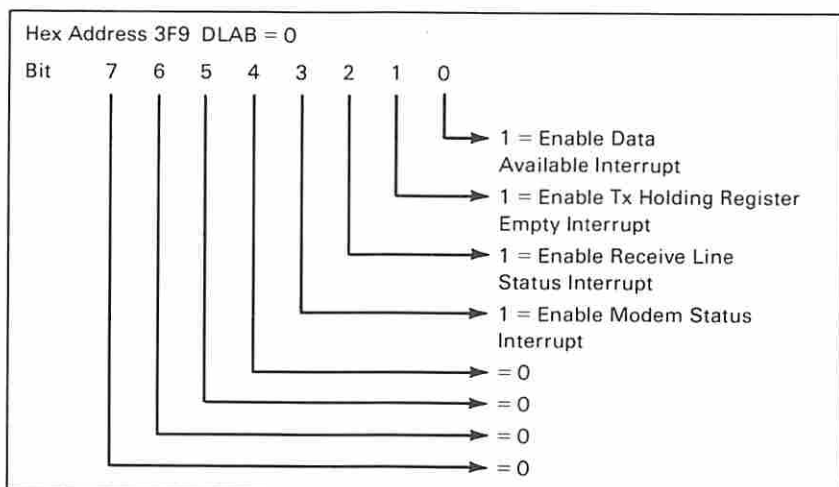
Bits 3 through 7: These five bits of the IIR are always logical 0.

Interrupt ID Register			Interrupt Set and Reset Functions			
Bit 2	Bit 1	Bit 0	Priority Level	Interrupt Type	Interrupt Source	Interrupt Reset Control
0	0	1	—	None	None	—
1	1	0	Highest	Receiver Line Status	Overrun Error or Parity Error or Framing Error or Break Interrupt	Reading the Line Status Register
1	0	0	Second	Received Data Available	Receiver Data Available	Reading the Receiver Buffer Register
0	1	0	Third	Transmitter Holding Register Empty	Transmitter Holding Register Empty	Reading the IIR Register (if source of interrupt) or Writing into the Transmitter Holding Register
0	0	0	Fourth	Modem Status	Clear to Send or Data Set Ready or Ring Indicator or Received Line Signal Direct	Reading the Modem Status Register

Interrupt Control Functions

Interrupt Enable Register

This eight-bit register enables the four types of interrupt of the INS8250 to separately activate the chip interrupt (INTRPT) output signal. It is possible to totally disable the interrupt system by resetting bits 0 through 3 of the interrupt enable register. Similarly, by setting the appropriate bits of this register to a logical 1, selected interrupts can be enabled. Disabling the interrupt system inhibits the interrupt identification register and the active (high) INTRPT output from the chip. All other system functions operate in their normal manner, including the setting of the line status and modem status registers. The contents of the interrupt enable register are indicated and described below:



Interrupt Enable Register (IER)

Bit 0: This bit enables the received data available interrupt when set to logical 1.

Bit 1: This bit enables the transmitter holding register empty interrupt when set to logical 1.

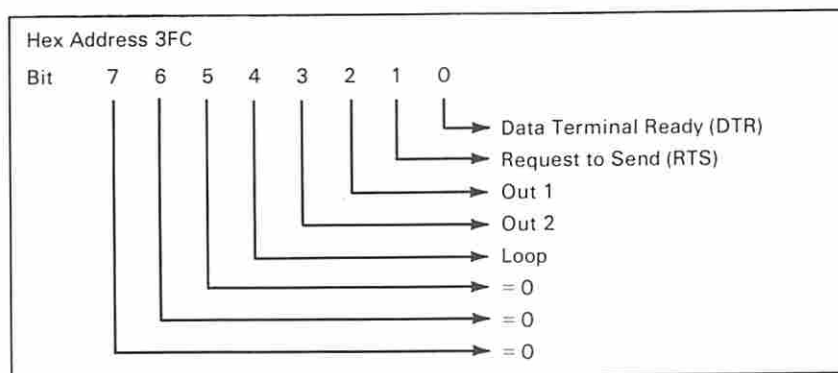
Bit 2: This bit enables the receiver line status interrupt when set to logical 1.

Bit 3: This bit enables the modem status interrupt when set to logical 1.

Bits 4 through 7: These four bits are always logical 0.

Modem Control Register

This eight-bit register controls the interface with the modem or data set (or a peripheral device emulating a modem). The contents of the modem control register are indicated and described below:



Modem Control Register (MCR)

Bit 0: This bit controls the data terminal ready ($\overline{\text{DTR}}$) output. When bit 0 is set to a logical 1, the $\overline{\text{DTR}}$ output is forced to a logical 0. When bit 0 is reset to a logical 0, the $\overline{\text{DTR}}$ output is forced to a logical 1.

Note: The $\overline{\text{DTR}}$ output of the INS8250 may be applied to an EIA inverting line driver (such as the DS1488) to obtain the proper polarity input at the succeeding modem or data set.

Bit 1: This bit controls the request to send ($\overline{\text{RTS}}$) output. Bit 1 affects the $\overline{\text{RTS}}$ output in a manner identical to that described above for bit 0.

Bit 2: This bit controls the output 1 ($\overline{\text{OUT 1}}$) signal, which is an auxiliary user-designated output. Bit 2 affects the $\overline{\text{OUT 1}}$ output in a manner identical to that described above for bit 0.

Bit 3: This bit controls the output 2 ($\overline{\text{OUT 2}}$) signal, which is an auxiliary user-designated output. Bit 3 affects the $\overline{\text{OUT 2}}$ output in a manner identical to that described above for bit 0.

Bit 4: This bit provides a loopback feature for diagnostic testing of the INS8250. When bit 4 is set to logical 1, the following occurs: the transmitter serial output (SOUT) is set to the marking (logical 1) state; the receiver serial input (SIN) is disconnected; the output of the transmitter shift register is "looped back" into the receiver shift register input; the four modem control inputs ($\overline{\text{CTS}}$, $\overline{\text{DSR}}$, $\overline{\text{RLSD}}$, AND $\overline{\text{RI}}$) are disconnected; and the four modem control outputs ($\overline{\text{DTR}}$, $\overline{\text{RTS}}$, $\overline{\text{OUT 1}}$, and $\overline{\text{OUT 2}}$) are internally connected to the four modem control inputs. In the diagnostic mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit- and receive-data paths of the INS8250.

In the diagnostic mode, the receiver and transmitter interrupts are fully operational. The modem control interrupts are also operational but the interrupts' sources are now the lower four bits of the modem control register instead of the four modem control inputs. The interrupts are still controlled by the interrupt enable register.

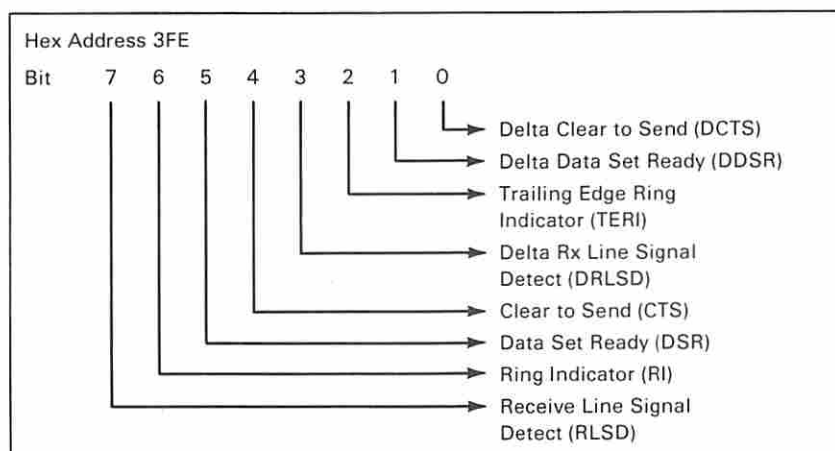
The INS8250 interrupt system can be tested by writing into the lower four bits of the modem status register. Setting any of these bits to a logical 1 generates the appropriate interrupt (if enabled). The resetting of these interrupts is the same as in normal INS8250 operation. To return to normal operation, the registers must be reprogrammed for normal operation and then bit 4 of the modem control register must be reset to logical 0.

Bits 5 through 7: These bits are permanently set to logical 0.

Modem Status Register

This eight-bit register provides the current state of the control lines from the modem (or peripheral device) to the processor. In addition to this current-state information, four bits of the modem status register provide change information. These bits are set to a logical 1 whenever a control input from the modem changes state. They are reset to logical 0 whenever the processor reads the modem status register.

The content of the modem status register are indicated and described below:



Modem Status Register (MSR)

Bit 0: This bit is the delta clear to send (DCTS) indicator. Bit 0 indicates that the $\overline{\text{CTS}}$ input to the chip has changed state since the last time it was read by the processor.

Bit 1: This bit is the delta data set ready (DDSR) indicator. Bit 1 indicates that the $\overline{\text{DSR}}$ input to the chip has changed state since the last time it was read by the processor.

Bit 2: This bit is the trailing edge of ring indicator (TERI) detector. Bit 2 indicates that the $\overline{\text{RI}}$ input to the chip has changed from an On (logical 1) to an Off (logical 0) condition.

Bit 3: This bit is the delta received line signal detector (DRLSD) indicator. Bit 3 indicates that the $\overline{\text{RLSD}}$ input to the chip has changed state.

Note: Whenever bit 0, 1, 2, or 3 is set to a logical 1, a modem status interrupt is generated.

Bit 4: This bit is the complement of the clear to send ($\overline{\text{CTS}}$) input. If bit 4 (loop) of the MCR is set to a logical 1, this is equivalent to RTS in the MCR.

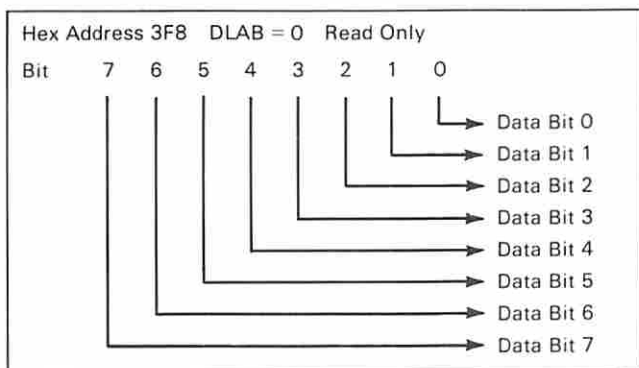
Bit 5: This bit is the complement of the data set ready ($\overline{\text{DSR}}$) input. If bit 4 of the MCR is set to a logical 1, this bit is equivalent to DTR in the MCR.

Bit 6: This bit is the complement of the ring indicator ($\overline{\text{RI}}$) input. If bit 4 of the MCR is set to a logical 1, this bit is equivalent to OUT 1 in the MCR.

Bit 7: This bit is the complement of the received line signal detect ($\overline{\text{RLSD}}$) input. If bit 4 of the MCR is set to a logical 1, this bit is equivalent to OUT 2 of the MCR.

Receiver Buffer Register

The receiver buffer register contains the received character as defined below:

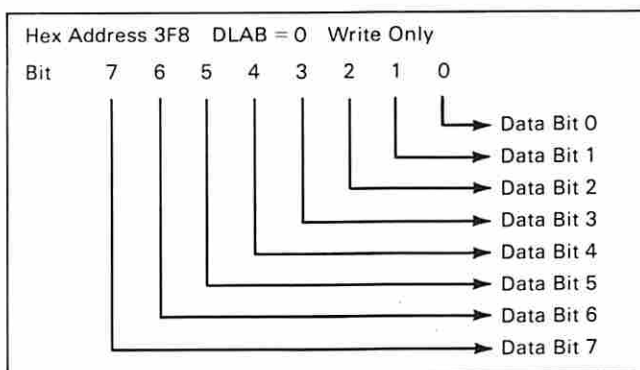


Receiver Buffer Register (RBR)

Bit 0 is the least significant bit and is the first bit serially received.

Transmitter Holding Register

The transmitter holding register contains the character to be serially transmitted and is defined below:



Transmitter Holding Register (THR)

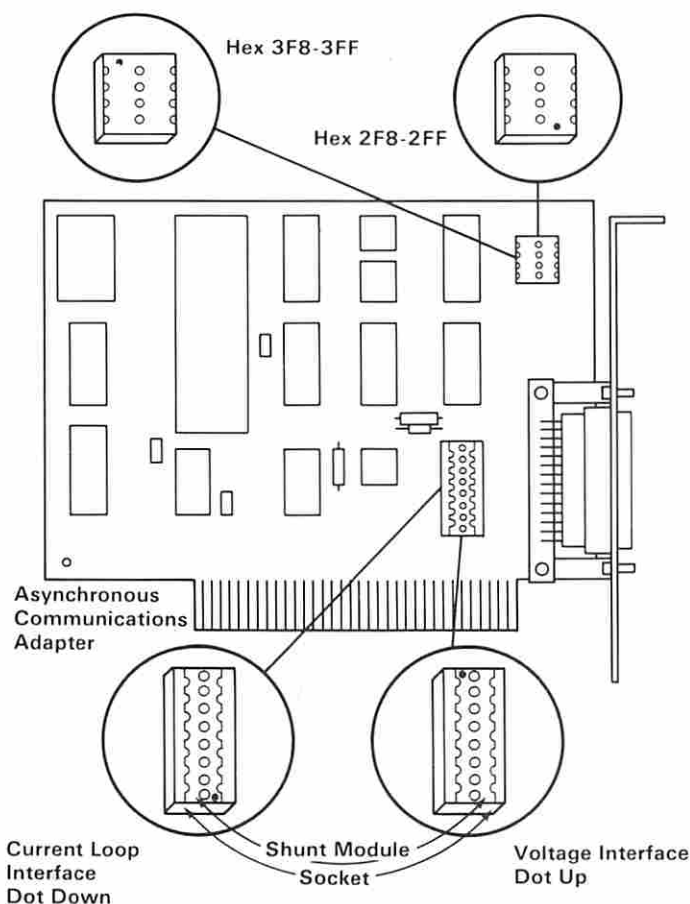
Bit 0 is the least significant bit and is the first bit serially transmitted.

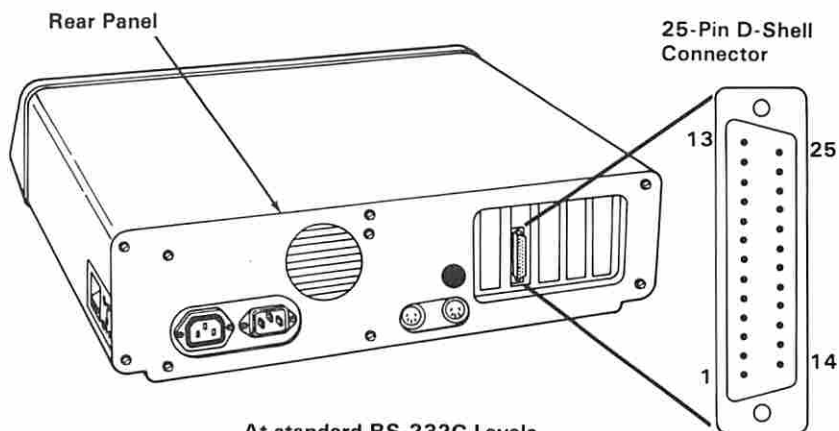
Selecting the Interface Format and Adapter Address

The voltage or current loop interface and adapter address are selected by plugging the programmed shunt modules with the locator dots up or down. See the figure below for the configurations.

Module Position
for Primary Asynchronous
Adapter

Module Position
for Alternate Asynchronous
Adapter





At standard RS-232C Levels
(with exception of current loops)

External Device	Description	Pin	Asynchronous Communications Adapter (RS-232C)
	NC	1	
	Transmitted Data	2	
←	Received Data	3	→
←	Request to Send	4	→
	Clear to Send	5	
	Data Set Ready	6	
	Signal Ground	7	
	Received Line Signal Detector	8	
←	+Transmit Current Loop Data	9	→
	NC	10	
←	-Transmit Current Loop Data	11	→
	NC	12	
	NC	13	
	NC	14	
	NC	15	
	NC	16	
	NC	17	
	+Receive Current Loop Data	18	→
	NC	19	
←	Data Terminal Ready	20	→
	NC	21	
	Ring Indicator	22	
	NC	23	
	NC	24	
	-Receive Current Loop Return	25	→

Note: To avoid inducing voltage surges on interchange circuits, signals from interchange circuits shall be used to drive inductive devices, such as relay coils.

Connector Specifications

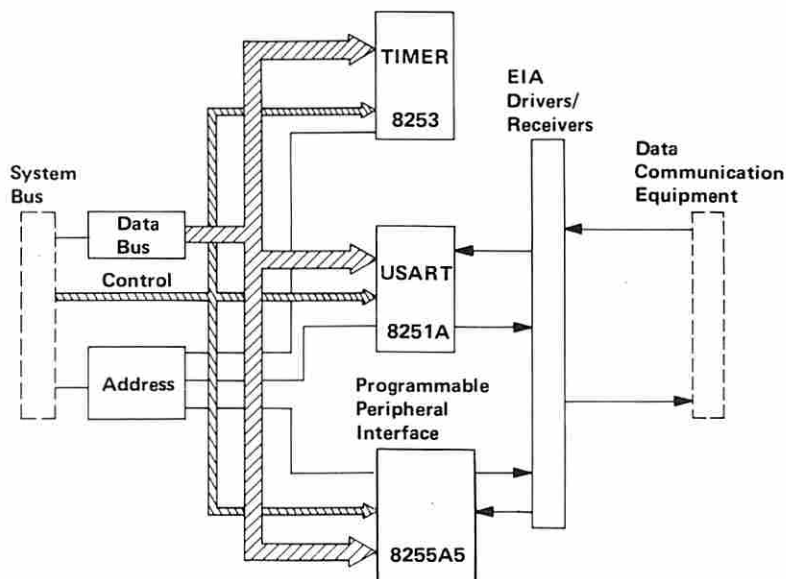
Notes:

Binary Synchronous Communications Adapter

The binary synchronous communications (BSC) adapter is a 4-inch high by 7.5-inch wide card that provides an RS232C-compatible communication interface for the IBM Personal Computer. All system control, voltage, and data signals are provided through a 2- by 31-position card-edge tab. External interface is in the form of EIA drivers and receivers connected to an RS232C, standard 25-pin, D-shell connector.

The adapter is programmed by communication software to operate in binary synchronous mode. Maximum transmission rate is 9600 bits per second (bps). The heart of the adapter is an Intel 8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART). An Intel 8255A-5 programmable peripheral interface (PPI) is also used for an expanded modem interface, and an Intel 8253-5 programmable interval timer provides time-outs and generates interrupts.

The following is a block diagram of the BSC adapter.



BSC Adapter Block Diagram

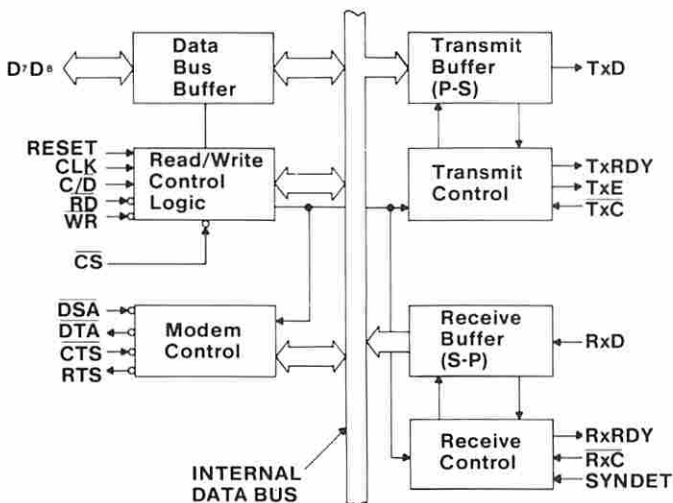
Functional Description

8251A Universal Synchronous/Asynchronous Receiver/Transmitter

The 8251A operational characteristics are programmed by the system unit's software, and it can support virtually any form of synchronous data technique currently in use. In the configuration being described, the 8251A is used for IBM's binary synchronous communications (BSC) protocol in half-duplex mode.

Operation of the 8251A is started by programming the communications format, then entering commands to tell the 8251A what operation is to be performed. In addition, the 8251A can pass device status to the system unit by doing a Status Read operation. The sequence of events to accomplish this are mode instruction, command instruction, and status read. Mode instruction must follow a master reset operation. Commands can be issued in the data block at any time during operation of the 8251A.

A block diagram of the 8251A follows:



8251A Block Diagram

Data Bus Buffer

The system unit's data bus interfaces the 8251A through the data bus buffer. Data is transferred or received by the buffer upon execution of input or output instructions from the system unit. Control words, command words, and status information are also transferred through the data bus buffer.

Read/Write Control Logic

The read/write control logic controls the transfer of information between the system unit and the 8251A. It consists of pins designated as RESET, CLK, WR, RD, C/D, and CS.

RESET: The Reset pin is gated by Port B, bit 4 of the 8255, and performs a master reset of the 8251A. The minimum reset pulse width is 6 clock cycles. Clock-cycle duration is determined by the oscillator speed of the processor.

CLK (Clock): The clock generates internal device timing. No external inputs or outputs are referenced to CLK. The input is the system board's bus clock of 4.77 MHz.

WR (Write): An input to WR informs the 8251A that the system unit is writing data or control words to it. The input is the WR signal from the system-unit bus.

RD (Read): An input to RD informs the 8251A that the processing unit is reading data or status information from it. The input is the RD signal from the system-unit bus.

C/D (Control/Data): An input on this pin, in conjunction with the WR and RD inputs, informs the 8251A that the word on the data bus is either a data character, a control word, or status information. The input is the low-order address bit from the system board's address bus.

CS (Chip Select): A low on the input selects the 8251A. No reading or writing will occur unless the device is selected. An input is decoded at the adapter from the address information on the system-unit bus.

Modem Control

The 8251A has the following input and output control signals which are used to interface the transmission equipment selected by the user.

DSR (Data Set Ready): The DSR input port is a general-purpose, 1-bit, inverting input port. The 8251A can test its condition with a Status Read operation.

CTS (Clear to Send): A low on this input enables the 8251A to transfer serial data if the TxEnable bit in the command byte is set to 1. If either a TxEnable off or CTS off condition occurs while the transmitter is in operation, the transmitter will send all the data in the USART that was written prior to the TxDisable command, before shutting down.

DTR (Data Terminal Ready): The DTR output port is a general-purpose, 1-bit, inverting output port. It can be set low by programming the appropriate bit in the command instruction word.

RTS (Request to Send): The RTS output signal is a general-purpose, 1-bit, inverting output port. It can be set low by programming the appropriate bit in the Command Instruction word.

Transmitter Buffer

The transmitter buffer accepts parallel data from the data-bus buffer, converts it to a serial bit stream, and inserts the appropriate characters or bits for the BSC protocol. The output from the transmit buffer is a composite serial stream of data on the falling edge of Transmit Clock. The transmitter will begin transferring data upon being enabled, if CTS = 0 (active). The transmit data (TxD) line will be set in the marking state upon receipt of a master reset, or when transmit enable/CTS is off and the transmitter is empty (TxEmpty).

Transmitter Control

Transmitter Control manages all activities associated with the transfer of serial data. It accepts and issues the following signals, both externally and internally, to accomplish this function:

TxRDY (Transmitter Ready): This output signals the system unit that the transmitter is ready to accept a data character. The TxRDY output pin is used as an interrupt to the system unit (Level 4) and is masked by turning off Transmit Enable. TxRDY is automatically reset by the leading edge of a WR input signal when a data character is loaded from the system unit.

TxE (Transmitter Empty): This signal is used only as a status register input.

TxC (Transmit Clock): The Transmit Clock controls the rate at which the character is to be transmitted. In synchronous mode, the bit-per-second rate is equal to the TxC frequency. The falling edge of TxC shifts the serial data out of the 8251A.

Receiver Buffer

The receiver accepts serial data, converts it to parallel format, checks for bits or characters that are unique to the communication technique, and sends an "assembled" character to the system unit. Serial data input is received on the Rx D (Receive Data) pin, and is clocked in on the rising edge of Rx C (Receive Clock).

Receiver Control

This control manages all receiver-related activities. The parity-toggle and parity-error flip-flop circuits are used for parity-error detection, and set the corresponding status bit.

RxRDY (Receiver Ready): This output indicates that the 8251A has a character that is ready to be received by the system unit. RxRDY is connected to the interrupt structure of the system unit (Interrupt Level 3). With Receive Enable off, RxRDY is masked and held in the reset mode. To set RxRDY, the receiver must be enabled, and a character must finish assembly and be transferred to the data output register. Failure to read the received character from the RxRDY output register before the assembly of the next Rx Data character will set an overrun-condition error, and the previous character will be lost.

RxC (Receiver Clock): The receiver clock controls the rate at which the character is to be received. The bit rate is equal to the actual frequency of RxC.

SYNDET (Synchronization Detect): This pin is used for synchronization detection and may be used as either input or output, programmable through the control word. It is reset to output-mode-low upon reset. When used as an output (internal synchronization mode), the SYNDET pin will go to 1 to indicate that the 8251A has found the synchronization character in the receive mode. If the 8251A is programmed to use double synchronization characters (bisynchronization, as in this application), the SYNDET pin will go to 1 in the middle of the last bit of the second synchronization character. SYNDET is automatically reset for a Status Read operation.

8255A-5 Programmable Peripheral Interface

The 8255A-5 is used on the BSC adapter to provide an expanded modem interface and for internal gating and control functions. It has three 8-bit ports, which are defined by the system during initialization of the adapter. All levels are considered plus active unless otherwise indicated. A detailed description of the ports is in "Programming Considerations" in this section.

8253-5 Programmable Interval Timer

The 8253-5 is driven by a divided-by-two system-clock signal. Its outputs are used as clocking signals and to generate inactivity timeout interrupts. These level 4 interrupts occur when either of the timers reaches its programmed terminal counts. The 8253-5 has the following outputs:

Timer 0: Not used for synchronous-mode operation.

Timer 1: Connected to port A, bit 7 of the 8255 and Interrupt Level 4.

Timer 2: Connected to port A, bit 6 of the 8255 and Interrupt Level 4.

Operation

The complete functional definition of the BSC adapter is programmed by the system software. Initialization and control words are sent out by the system to initialize the adapter and program the communications format in which it operates. Once programmed, the BSC Adapter is ready to perform its communication functions.

Transmit

In synchronous transmission, the TxD output is continuously at a mark level until the system sends its first character, which is a synchronization character to the 8251A. When the CTS line goes on, the first character is serially transmitted. All bits are shifted out on the falling edge of TxC. When the 8251A is ready to receive another character from the system for transmission, it raises TxRDY, which causes a level-4 interrupt.

Once transmission has started, the data stream at the TxD output must continue at the TxC rate. If the system does not provide the 8251A with a data character before the 8251A transmit buffers become empty, the synchronization characters will be automatically inserted in the TxD data stream. In this case, the TxE bit in the status register is raised high to signal that the 8251A is empty and that synchronization characters are being sent out. (Note that this TxE bit is in the status register, and is not the TxE pin on the 8251A). TxE does not go low when SYNC is being shifted out. The TxE status bit is internally reset by a data character being written to the 8251A.

Receive

In synchronous reception, the 8251A will achieve character synchronization, because the hardware design of the BSC adapter is intended for internal synchronization. Therefore, the SYNDET pin on the 8251A is not connected to the adapter circuits. For internal synchronization, the Enter Hunt command should be included in the first command instruction word written. Data on the RxD pin is then sampled in on the rising edge of RxC. The content of the RxD buffer is compared at every bit boundary with the first SYNC character until a match occurs. Because the 8251A has been programmed for two synchronization characters (bisynchronization), the next received character is also compared. When both SYNC characters have been detected, the 8251A ends the hunt mode and is in character synchronization. The SYNDET bit in the status register (not the SYNDET pin) is then set high, and is reset automatically by a Status Read.

Once synchronization has occurred, the 8251A begins to assemble received data bytes. When a character is assembled and ready to be transferred to memory from the 8251A, it raises RxDY, causing an interrupt level 3 to the system.

If the system has not fetched a previous character by the time another received character is assembled (and an interrupt-level 3 issued by the adapter), the old character will be overwritten, and the overrun error flag will be raised. All error flags can be reset by an error reset operation.

Programming Considerations

Before starting data transmission or reception, the BSC adapter is programmed by the system unit to define control and gating ports, timer functions and counts, and the communication environment in which it is to operate.

Typical Programming Sequence

The 8255A-5 programmable peripheral interface (PPI) is initialized for the proper mode by selecting address hex 3A3 and writing the control word. This defines port A as an input, port B as an output for modem control and gating, and port C for 4-bit input and 4-bit output. The bit descriptions for the 8255A-5 are shown in the following figures. Using an output to port C, the adapter is then set to wrap mode, disallow interrupts, and gate external clocks (address=3A2H, data=0DH). The adapter is now isolated from the communication interface, and initialization continues.

Through bit 4 of 8255 Port B, the 8251A reset pin is brought high, held, then dropped. This resets the internal registers of the 8251A.

8255 Port A Assignments Input Port	Address: hex 3A0 for BSC hex 380 for Alternate BSC
	0 = Ring Indicate is on from Interface 0 = Data Carrier Defect is on from Interface - Oscillating = Transmit Clock Active 0 = Clear-to-Send is on from Interface - Oscillating = Receive Clock Active 1 = TxRDY Active 1 = Timer 2 Output Active 1 = Timer 1 Output Active
8255 Port B Assignments Output Port	Address: hex 3A1 for BSC hex 381 for Alternate BSC
	0 = Turn on Data Signal Rate Selector 0 = Turn on Select Standby 0 = Turn on Test 1 = Not Used 1 = Reset 8251A 1 = Gate Timer 2 1 = Gate Timer 1 1 = Gate Timers 1 and 2 to Interrupt Level 4
8255 Port C Assignments	Address: hex 3A2 for BSC hex 382 for Alternate BSC
	1 = Gate Internal Clock (Output Bit) 1 = Gate External Clock (Output Bit) 1 = Electronic Wrap (Output Bit) 0 = Enable Timer 1 and 2, Interrupt 6 and Receive Interrupt 3 - Oscillating = Receive Data (Input Bit) - Oscillating = Timer 0 Output (Input Bit) 0 = Test Indicate Active (Input Bit) 0 = BSC Adapter

The 8253-5 programmable interval timer is used in the synchronous mode to provide inactivity time-outs to interrupt the system unit after a preselected period of time has elapsed from the start of a communication operation. Counter 0 is not used for synchronous operation. Counters 1 and 2 are connected to interrupt-level 4, and are programmed to terminal-count values, which will provide the desired time delay before a level-4 interrupt is generated. These interrupts will indicate to the system software that a predetermined period of time has elapsed without a TxRDY (level 4) or RxRDY (level 3) interrupt being sent to the system unit.

The modes for each counter are programmed by selecting each timer-register address and writing the correct control word for counter operation to the adapter. The mode for counters 1 and 2 is set to 0. The terminal-count values are loaded using control-word bits D4 and D5 to select "load." The 8253-5 Control Word format is shown in the following chart.

Control Word Format				Address hex 3A7			
D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
SC1	SC0	RL1	RL0	M2	M1	M0	BCD

Definition of Control

SC — Select Counter:

SC1	SC0	
0	0	Select Counter 0
0	1	Select Counter 1
1	0	Select Counter 2
1	1	Illegal

RL — Read/Load:

RL1	RL0	
0	0	Counter Latching operation
1	0	Read/Load most significant byte only
0	1	Read/Load least significant byte only
1	1	Read/Load least significant byte first, then most significant byte

M — Mode:

M2	M1	M0	
0	0	0	Mode 0 Terminal Count Interrupt

BCD:

0	Binary Counter 16-bits
1	Binary Coded Decimal (BCD) Counter (4 Decades)

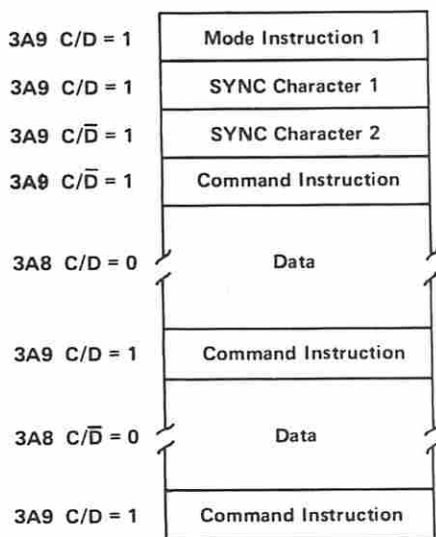
8253-5 Control Word Format

8251A Programming Procedures

After the support devices on the BSC adapter are programmed, the 8251A is loaded with a set of control words that define the communication environment. The control words are split into two formats, mode instruction, and command instruction.

Both the mode and command instructions must conform to a specified sequence for proper device operation. The mode instruction must be inserted immediately after a reset operation, before using the 8251A for data communications. The required synchronization characters for the defined communication technique are next loaded into the 8251A (usually hex 32 for BSC). All control words written to the 8251A after the mode instruction will load the command instruction. Command instructions can be written to the 8251A at any time in the data block anytime during the operation of the 8251A. To return to the mode instruction format, the master reset bit in the command instruction word can be set to start an internal reset operation which automatically places the 8251A back into the mode instruction format. Command instructions must follow the mode instructions or synchronization characters.

The following diagram is a typical data block, showing the mode instruction and command instruction.

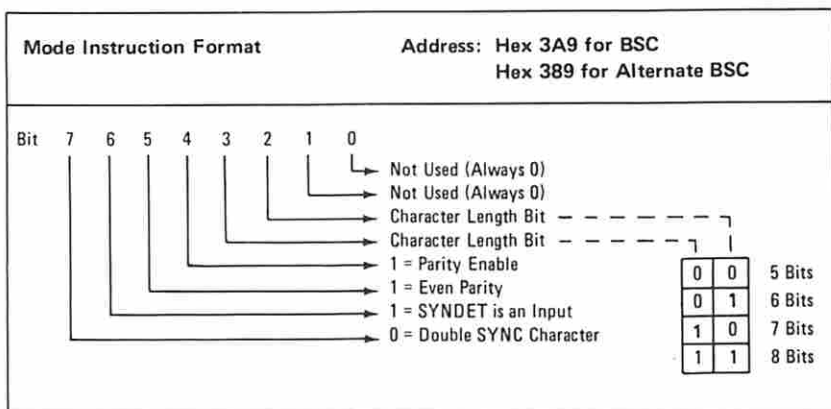


Typical Data Block

Mode Instruction Definition

The mode instruction defines the general operational characteristics of the 8251A. It follows a reset operation (internal or external). Once the mode instruction has been written to the 8251A by the system unit, synchronization characters or command instructions may be written to the device.

The following figure shows the format for the mode instruction.



Bit 0 Not used; always 0

Bit 1 Not used; always 0

Bit 2 and Bit 3 These two bits are used together to define the character length. With 0 and 1 as inputs on bits 2 and 3, character lengths of 5, 6, 7, and 8 bits can be established, as shown in the preceding figure.

Bit 4 In the synchronous mode, parity is enabled from this bit. A 1 on this bit sets parity enable.

Bit 5 The parity generation/check is set from this bit. For BSC, even parity is used by having bit 5 = 1.

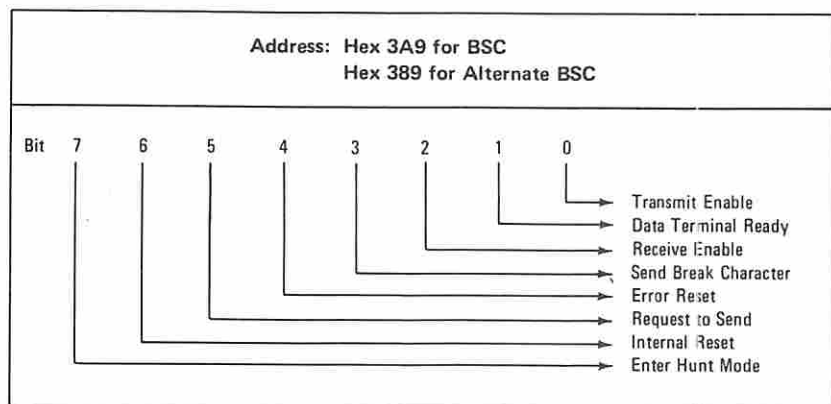
Bit 6 External synchronization is set by this bit. A 1 on this bit establishes synchronization detection as an input.

Bit 7 This bit establishes the mode of character synchronization. A 0 is set on this bit to give double character synchronization.

Command-Instruction Format

The command-instruction format defines a status word that is used to control the actual operation of the 8251A. Once the mode instruction has been written to the 8251A, and SYNC characters loaded, all further "Control Writes" to I/O address hex 3A9 or hex 389 will load a command instruction.

Data is transferred by accessing two I/O ports on the 8251A, ports 3A8 and 388. A byte of data can be read from port 3A8 and can be written to port 388.



Command Instruction Format

- Bit 0** The Transmit Enable bit sets the function of the 8251A to either enabled (1) or disabled (0).
- Bit 1** The Data Terminal Ready bit, when set to 1 will force the data terminal output to 0. This is a one-bit inverting output port.
- Bit 2** The Receive Enable bit sets the function to either enable the bit (1), or to disable the bit (0).
- Bit 3** The Send Break Character bit is set to 0 for normal BSC operation.
- Bit 4** The Error Reset bit is set to 1 to reset error flags from the command instruction.
- Bit 5** A 1 on the Request to Send bit will set the output to 0. This is a one-bit inverting output port.

- Bit 6** The Internal Reset bit when set to 1 returns the 8251A to mode-instruction format.
- Bit 7** The Enter Hunt bit is set to 1 for BSC to enable a search for synchronization characters.

Status Read Definition

In telecommunication systems, the status of the active device must often be checked to determine if errors or other conditions have occurred that require the processor's attention. The 8251A has a status read facility that allows the system software to read the status of the device at anytime during the functional operation. A normal read command is issued by the processor with I/O address hex 3A9 for BSC, and hex 389 for Alternate BSC to perform a status read operation.

The format for a status read word is shown in the figure below. Some of the bits in the status read format have the same meanings as external output pins so the 8251A can be used in a completely polled environment or in an interrupt-driven environment.

Address: Hex 3A9 for BSC Hex 389 for Alternate BSC	
Bit	
0	→ TxRDY (See Note Below)
1	→ RxRDY
2	→ TxEmpty
3	→ Parity Error (PE Flag On when a Parity Error Occurs)
4	→ Overrun Error (OE Flag On when Overrun Error Occurs)
5	→ Framing Error (Not Used for Synchronous Communications)
6	→ SYNDET
7	→ Data Set Ready (Indicates that DSR is at 0 Level)
Note: TxRDY status bit does not have the same meaning as the 8251A TxRDY output pin. The former is not conditioned by CTS and TxEnable. The latter is conditioned by both CTS and TxEnable.	

Status Read Format

- Bit 0 See the Note in the preceding figure.
- Bit 1 An output on this bit means a character is ready to be received by the computer's 8088 microprocessor.
- Bit 2 A 1 on this bit indicates the 8251A has no characters to transmit.
- Bit 3 The Parity Error bit sets a flag when errors are detected. It is reset by the error reset in the command instruction.
- Bit 4 This bit sets a flag when the computers 8088 microprocessor does not read a character before another one is presented. The 8251A operation is not inhibited by this flag, but the overrun character will be lost.
- Bit 5 Not used
- Bit 6 SYNDET goes to 1 when the synchronization character is found in receive mode. For BSC, SYNDET goes high in the middle of the last bit of the second synchronization character.
- Bit 7 The Data Set Ready bit is a one bit inverting input. It is used to check modem conditions, such as data-set ready.

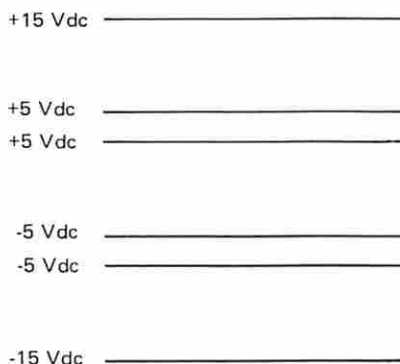
Interface Signal Information

The BSC adapter conforms to interface signal levels standardized by the Electronics Industry Association (EIA) RS232C Standard. These levels are shown in the following figure.

Additional lines, not standardized by the EIA, are pins 11, 18, and 25 on the interface connector. These lines are designated as Select Standby, Test, and Test Indicate. Select Standby is used to support the switched network backup facility of a modem that provides this option. Test and Test Indicate support a modem wrap function on modems that are designated for business-machine, controlled-modem wraps.

Driver

EIA RS232C/CCITT V24-V28 Signal Levels



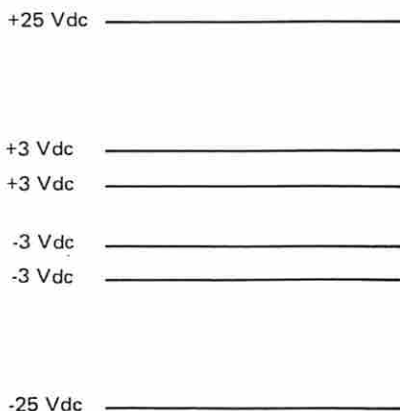
Active/Data = 0

Invalid Level

Inactive/Data = 1

Receiver

EIA RS232C/CCITT V24-V28 Signal Levels



Active/Data = 0

Invalid Level

Inactive/Data = 1

Interface Voltage Levels

Interrupt Information

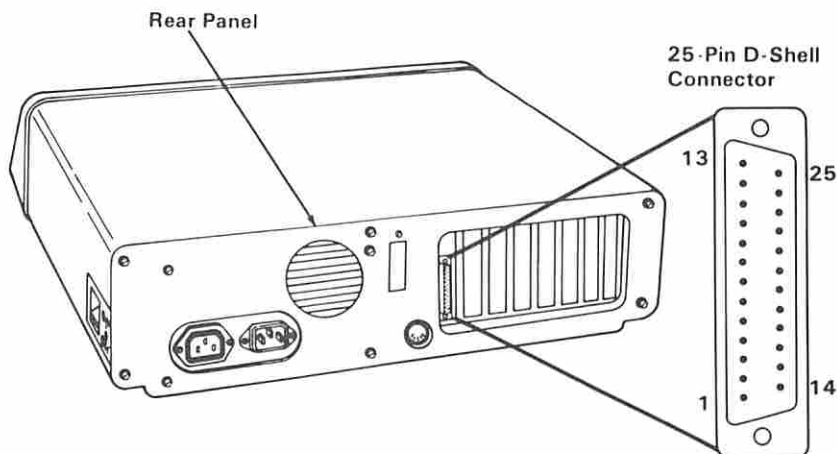
Interrupt Level 4: Transmitter Ready
Counter 1
Counter 2

Interrupt Level 3: Receiver Ready

The following chart is a device address summary for the primary and alternate modes of the binary synchronous communications adapter.

Hex Address		Device	Register Name	Function
Primary	Alternate			
3A0	380	8255	Port A Data	Internal/External Sensing
3A1	381	8255	Port B Data	External Modem Interface
3A2	382	8255	Port C Data	Internal Control
3A3	383	8255	Mode Set	8255 Mode Initialization
3A4	384	8253	Counter 0 LSB	Not Used in Synch Mode
3A4	384	8253	Counter 0 MSB	Not Used in Synch Mode
3A5	385	8253	Counter 1 LSB	Inactivity Time-Outs
3A5	385	8253	Counter 1 MSB	Inactivity Time-Outs
3A6	386	8253	Counter 2 LSB	Inactivity Time-Outs
3A6	386	8253	Counter 2 MSB	Inactivity Time-Outs
3A7	387	8253	Mode Register	8253 Mode Set
3A8	388	8251	Data Select	Data
3A9	389	8251	Command/Status	Mode/Command USART Status

Device Address Summary



Signal Name — Description		Pin
External Device	No Connection	1
	Transmitted Data	2
	Received Data	3
	Request to Send	4
	Clear to Send	5
	Data Set Ready	6
	Signal Ground	7
	Received Line Signal Detector	8
	No Connection	9
	No Connection	10
	Select Standby*	11
	No Connection	12
	No Connection	13
	No Connection	14
	Transmitter Signal Element Timing	15
	No Connection	16
	Receiver Signal Element Timing	17
	Test (IBM Modems Only)*	18
	No Connection	19
	Data Terminal Ready	20
	No Connection	21
	Ring Indicator	22
	Data Signal Rate Selector	23
	No Connection	24
	Test Indicate (IBM Modems Only)*	25
		Binary Synchronous Communications Adapter

*Not standardized by EIA (Electronics Industry Association).

Connector Specifications

Notes:

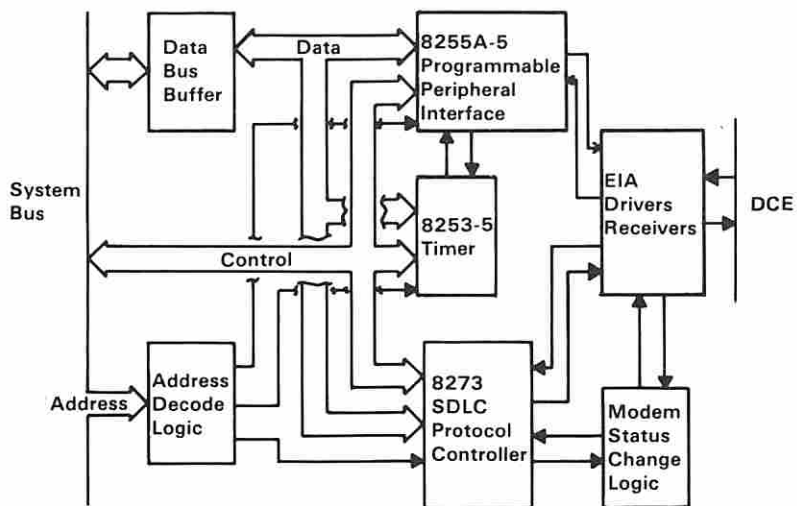
IBM Synchronous Data Link Control (SDLC) Communications Adapter

The SDLC communications adapter system control, voltage, and data signals are provided through a 2 x 31 position card edge tab. Modem interface is in the form of EIA drivers and receivers connecting to an RS-232C standard 25-pin, D-shell, male connector.

The adapter is programmed by communications software to operate in a half-duplex synchronous mode. Maximum transmission rate is 9600 bits per second, as generated by the attached modem or other data communication equipment.

The SDLC adapter utilizes an Intel 8273 SDLC protocol controller and an Intel 8255A-5 programmable peripheral interface for an expanded external modem interface. An Intel 8253 programmable interval timer is also provided to generate timing and interrupt signals. Internal test loop capability is provided for diagnostic purposes.

The figure below is a block diagram of the SDLC communications adapter.



SDLC Communications Adapter Block Diagram

The 8273 SDLC protocol control module has the following key features:

- Automatic frame check sequence generation and checking
- Automatic zero bit insertion and deletion.
- TTL compatibility.
- Dual internal processor architecture, allowing frame level command structure and control of data channel with minimal system processor intervention.

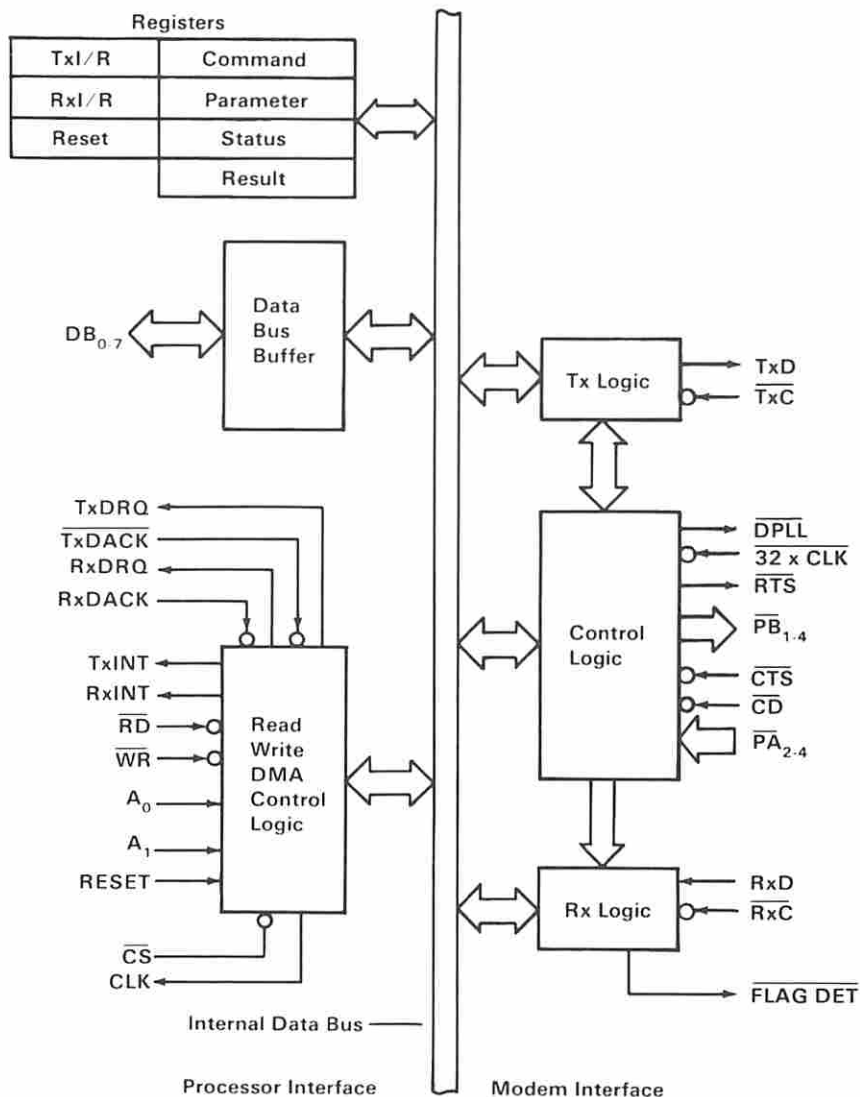
The 8273 SDLC protocol controller operations, whether transmission, reception, or port read, are each comprised of three phases:

Command	Commands and/or parameters for the required operation are issued by the processor.
Execution	Executes the command, manages the data link, and may transfer data to or from memory utilizing direct memory access (DMA), thus freezing the processor except for minimal interruptions.
Result	Returns the outcome of the command by returning interrupt results.

Support of the controller operational phases is through internal registers and control blocks of the 8273 controller.

8273 Protocol Controller Structure

The 8273 module consists of two major interfaces: the processor interface and the modem interface. A block diagram of the 8273 protocol controller module follows.



8273 SDLC Protocol Control Block Diagram

Processor Interface

The processor interface consists of four major blocks: the control/read/write logic (C/R/W), internal registers, data transfer logic, and data bus buffers.

Control/Read/Write Logic

The control/read/write logic is used by the processor to issue commands to the 8273. Once the 8273 receives and executes a command, it returns the results using the C/R/W logic. The logic is supported by seven registers which are addressed by A0, A1, RD, and WR, in addition to CS. A0 and A1 are the two low-order bits of the adapter address-byte. RD and WR are the processor read and write signals present on the system control bus. CS is the chip select, also decoded by the adapter address logic. The table below shows the address of each register using the C/R/W logic.

Address Inputs		Control Inputs			Register
A0	A1	CS	WR	RD	
0	0	0	0	1	Command
0	0	0	1	0	Status
0	1	0	0	1	Parameter
0	1	0	1	0	Result
1	0	0	0	1	Reset
1	0	0	1	0	Txl/R
1	1	0	0	1	None
1	1	0	1	0	Rxl/R

8273 SDLC Protocol Controller Register Selection

8273 Control/Read/Write Registers

Command	Operations are initialized by writing the appropriate command byte into this register.
Status	This register provides the general status of the 8273. The status register supplies the processor/adaptor handshaking necessary during various phases of the 8273 operation.
Parameter	Additional information that is required to process the command is written into this register. Some commands require more than one parameter.
Immediate Result (Result)	Commands that execute immediately produce a result byte in this register, to be read by the processor.
Transmit Interrupt Results (TxI/R)	Results of transmit operations are passed to the processor from this register. This result generates an interrupt to the processor when the result becomes available.
Receiver Interrupt Results (Rx/I/R)	Results of receive operations are passed to the processor from this register. This result generates an interrupt to the processor when the result becomes available.
Reset	This register provides a software reset function for the 8273.

The other elements of the C/R/W logic are the interrupt lines (RxINT and TxINT). Interrupt priorities are listed in the "Interrupt Information" table in this section. These lines signal the processor that either the transmitter or the receiver requires service (results should be read from the appropriate register), or a data transfer is required. The status of each interrupt line is also reflected by a bit in the status register, so non-interrupt driven operation is also possible by the communication software examining these bits periodically.

Data Interfaces

The 8273 supports two independent data interfaces through the data transfer logic: received data and transmitted data. These interfaces are programmable for either DMA or non-DMA data transfers. Speeds below 9600 bits-per-second may or may not require DMA, depending on the task load and interrupt response time of the processor. The processor DMA controller is used for management of DMA data transfer timing and addressing. The 8273 handles the transfer requests and actual counts of data-block lengths. DMA level 1 is used to transmit and receive data transfers. Dual DMA support is not provided.

Elements of Data Transfer Interface

TxDQ/RxDQ	This line requests a DMA to or from memory and is asserted by the 8273.
TxDACK/RxDACK	This line notifies the 8273 that a request has been granted and provides access to data regions. This line is returned by the DMA controller (DACK1 on the system unit control bus is connected to TxDACK/RxDACK on the 8273).
RD (Read)	This line indicates data is to be read from the 8273 and placed in memory. It is controlled by the processor DMA controller.
WR (Write)	This line indicates if data is to be written to the 8273 from memory and is controlled by the processor DMA controller.

To request a DMA transfer, the 8273 raises the DMA request line. Once the DMA controller obtains control of the system bus, it notifies the 8273 that the DRQ is granted by returning DACK, and WR or RD, for a transmit or receive operation, respectively. The DACK and WR or RD signals transfer data between the 8273 and memory, independent of the 8273 chip-select pin (CS). This "hard select" of data into the transmitter or out of the receiver alleviates the need for the normal transmit and receive data registers, addressed by a combination of address lines, CS, and WR or RD.

Modem Interface

The modem interface of the 8273 consists of two major blocks: the modem control block and the serial data timing block.

Modem Control Block

The modem control block provides both dedicated and user-defined modem control function. EIA inverting drivers and receivers are used to convert TTL levels to EIA levels.

Port A is a modem control input port. Bits PA0 and PA1 have dedicated functions.

8273 Port A (Modem Control Input Port)									
Bit PA	7	6	5	4	3	2	1	0	
									PA0 Clear to Send
									PA1 Carrier Detect
									PA2 Data Set Ready
									PA3 CTS Change
									PA4 DSR Change
									Not Used

Bit PA0 This bit reflects the logical state of the clear to send (CTS) pin. The 8273 waits until CTS is active before it starts transmitting a frame. If CTS goes inactive while transmitting, the frame is aborted and the processor is interrupted. A CTS failure will be indicated in the appropriate interrupt-result register.

Bit PA1 This bit reflects the logical state of the carrier detect pin (CD). CD must be active in sufficient time for reception of a frame's address field. If CD is lost (goes inactive) while receiving a frame, an interrupt is generated with a CD failure result.

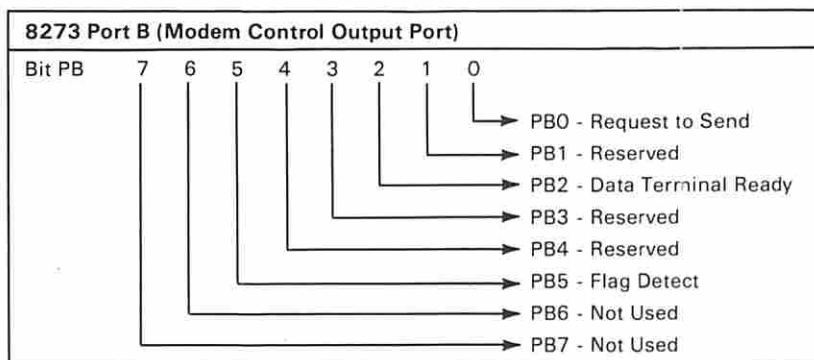
Bit PA2 This bit is a sense bit for data set ready (DSR).

Bit PA3 This bit is a sense bit to detect a change in CTS.

Bit PA4 This bit is a sense bit to detect a change in data set ready.

Bits PA5 to PA7 These bits are not used and each is read as a 1 for a read port A command.

Port B is a modem control output port. Bits PB0 and PB5 are dedicated function pins.



Bit PB0 This bit represents the logical state of request to send (RTS). This function is handled automatically by the 8273.

Bit PB1 Reserved.

Bit PB2 Used for data terminal ready.

Bit PB3 Reserved.

Bit PB4 Reserved.

Bit PB5 This bit reflects the state of the flag detect pin. This pin is activated whenever an active receiver sees a flag character.

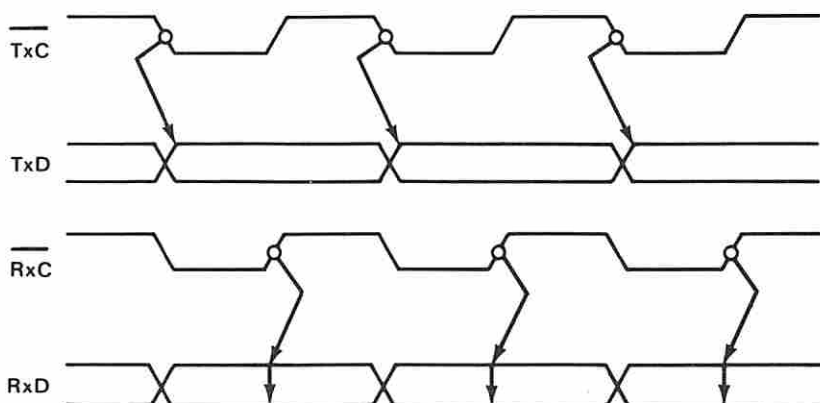
Bit PB6 Not used.

Bit PB7 Not used.

Serial Data Timing Block

The serial data timing block is comprised of two sections: the serial data logic and the digital phase locked loop (DPLL).

Elements of the serial data logic section are the data pins TxD (transmitted data output) and RxD (received data input), and the respective clocks. The leading edge of TxC generates new transmitted data and the trailing edge of RxC is used to capture the received data. The figure below shows the timing for these signals.



8273 SDLC Protocol Controller Transmit/Receive Timing

The digital phase locked loop provided on the 8273 controller module is utilized to capture looped data in proper synchronization during wrap operations performed by diagnostics.

8255A-5 Programmable Peripheral Interface

The 8255A-5 contains three 8-bit ports. Descriptions of each bit of these ports are as follows:

8255A-5 Port A Assignments*								Hex Address 380
Bit	7	6	5	4	3	2	1	0
								0 = Ring Indicator is on from Interface
								0 = Data Carrier Detect is on from Interface
								Oscillating = Transmit Clock Active
								0 = Clear to Send is on from Interface
								Oscillating = Receive Clock Active
								1 = Modem Status Changed
								1 = Timer 2 Output Active
								1 = Timer 1 Output Active

*Port A is defined as an input port

8255A-5 Port B Assignments*								Hex Address 381
Bit	7	6	5	4	3	2	1	0
								0 = Turn On Data Signal Rate Select at Modem Interface
								0 = Turn On Select Standby at Modem Interface
								0 = Turn On Test
								1 = Reset Modem Status Changed Logic
								1 = Reset 8273
								1 = Gate Timer 2
								1 = Gate Timer 1
								1 = Enable Level 4 Interrupt

*Port B is defined as an output port

8255A-5 Port C Assignments*								Hex Address 382
Bit	7	6	5	4	3	2	1	0
								1 = Gate Internal Clock (Output Bit)
								1 = Gate External Clock (Output Bit)
								1 = Electronic Wrap (Output Bit)
								0 = Gate Interrupts 3 and 4 (Output Bit)
								Oscillating = Receive Data (Input Bit)
								Oscillating = Timer 0 Output (Input bit)
								0 = Test Indicate Active (Input Bit)
								Not Used

*Port C is defined for internal control and gating functions. It has three input and four output bits. The four output bits are defined during initialization, but only three are used.

8253-5 Programmable Interval Timer

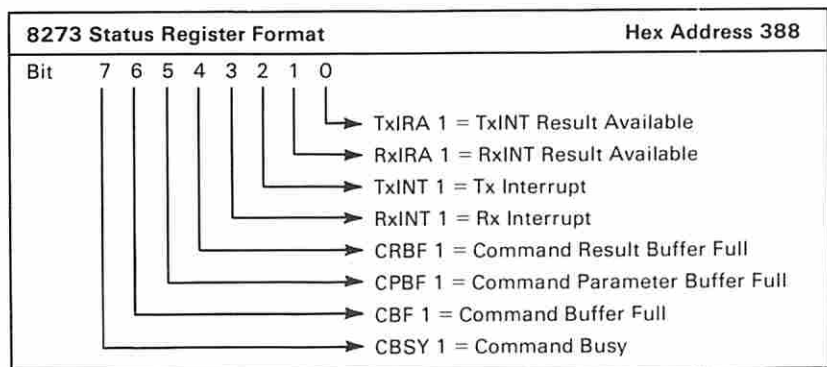
The 8253-5 is driven by a processor clock signal divided by two. It has the following output:

- Timer 0 Programmed to generate a square wave signal, used as an input to timer 2. Also connected to 8253 port C, bit 5.
- Timer 1 Connected to 8255 port A, bit 7, and interrupt level 4.
- Timer 2 Connected to 8255 port A, bit 6, and interrupt 4.

Programming Considerations

The software aspects of the 8273 involve the communication of both commands from the processor to the 8273 and the return of results of those commands from the 8273 to the processor. Due to the internal processor architecture of the 8273, this system unit/8273 communication is basically a form of interprocessor communication, and must be considered when programming for the SDLC communications adapter.

The protocol for this interprocessor communication is implemented through use of handshaking supplied in the 8273 status register. The bit definitions of this register are shown below.



- Bit 0** This bit is the transmitter interrupt result available (TxIRA) bit. This bit is set when the 8273 places an interrupt-result byte in the TxI/R register, and reset when the processor reads the TxI/R register.
- Bit 1** This bit is the receiver interrupt result available (RxIRA) bit. It is the corresponding result-available bit for the receiver. It is set when the 8273 places an interrupt-result byte in the RxI/R register and reset when the processor reads the register.
- Bit 2** This bit is the transmitter interrupt (TxINT) bit and reflects the state of the TxINT pin. TxINT is set by the 8273 whenever the transmitter needs servicing, and reset when the processor reads the result or performs the data transfer.

- Bit 3 This bit is the receiver interrupt (RxINT) bit and is identical to the TxINT, except action is initiated based on receiver interrupt-sources.
- Bit 4 This bit is the command result buffer full (CRBF) bit. It is set when the 8273 places a result from an immediate-type command in the result register, and reset when the processor reads the result or performs the data transfer.
- Bit 5 This bit is the command parameter buffer full (CPBF) bit and indicates that the parameter register contains a parameter. It is set when the processor deposits a parameter in the parameter register, and reset when the 8273 accepts the parameter.
- Bit 6 This bit is the command buffer full (CBF) bit and, when set, it indicates that a byte is present in the command register. This bit is normally not used.
- Bit 7 This bit is the command busy (CBSY) bit and indicates when the 8273 is in the command phase. It is set when the processor writes a command into the command register, starting the command phase. It is reset when the last parameter is deposited in the parameter register and accepted by the 8273, completing the command phase.

Initializing the Adapter (Typical Sequence)

Before initialization of the 8273 protocol controller, the support devices on the card must be initialized to the proper modes of operation.

Configuration of the 8255A-5 programmable peripheral interface is accomplished by selecting the mode-set address for the 8255 (see the "SDLC Communications Adapter Device Addresses" table later in this section) and writing the appropriate control word to the device (hex 98) to set ports A, B, and C to the modes described previously in this section.

Next, a bit pattern is output to port C which disallows interrupts, sets wrap mode on, and gates the external clock pins (address = hex 382, data = hex 0D). The adapter is now isolated from the communications interface.

Using bit 4 of port B, the 8273 reset line is brought high, held, and then dropped. This resets the internal registers of the 8273.

The 8253-5's counter 1 and 2 terminal-count values are now set to values which will provide the desired time delay before a level 4 interrupt is generated. These interrupts may be used to indicate to the communication software that a pre-determined period of time has elapsed without a result interrupt (interrupt level 3). The terminal count-values for these counters are set for any time delay which the programmer requires. Counter 0 is also set at this time to mode 3 (generates square wave signal, used to drive counter 2 input).

To setup the counter modes, the address for the 8253 counter-mode register is selected (see the "SDLC Communications Adapter Device Addresses" table, later in this section), and the control word for each individual counter is written to the device separately. The control-word format and bit definitions for the 8253 are shown below. Note that the two most-significant bits of the control word select each individual counter, and each counter mode is defined separately.

Once the support devices have been initialized to the proper modes and the 8273 has been reset, the 8273 protocol controller is ready to be configured for the operating mode that defines the communications environment in which it will be used.

Control Word Format

D ₇	D ₆	D ₅	D ₄	D ₃	D ₂	D ₁	D ₀
SC1	SC0	RL1	RL0	M2	M1	M0	BCD

Definitions of Control**SC - Select Counter:**

SC1	SC0	
0	0	Select Counter 0
0	1	Select Counter 1
1	0	Select Counter 2
1	1	Illegal

RL - Read/Load:

RL1	RL0	
0	0	Counter Latching operation
1	0	Read/Load most significant byte (MSB)
0	1	Read/Load least significant byte (LSB)
1	1	Read/Load least significant byte first, then most significant byte.

M - Mode:

M2	M1	M0	Mode
0	0	0	Mode 0
0	0	1	Mode 1
X	1	0	Mode 2
X	1	1	Mode 3
1	0	0	Mode 4
1	0	1	Mode 5

BCD:

0	Binary Counter 16-bits
1	Binary Coded Decimal (BCD) Counter (4 Decades)

8253-5 Programmable Interval Timer Control Word

Initialization/Configuration Commands

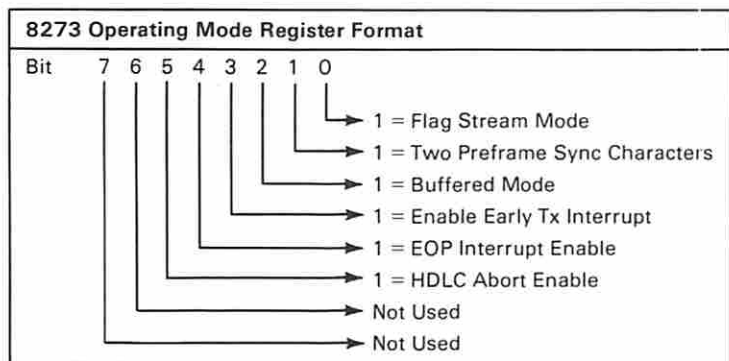
The initialization/configuration commands manipulate internal registers of the 8273, which define operating modes. After chip reset, the 8273 defaults to all 1's in the mode registers. The initialization/configuration commands either set or reset specified bits in the registers depending on the type of command. One parameter is required with the commands. The parameter is actually the bit pattern (mask) used by the set or reset command to manipulate the register bits.

Set commands perform a logical OR operation of the parameter (mask) of the internal register. This mask contains 1's where register bits are to be set. Zero (0's) in the mask cause no change to the corresponding register bit.

Reset commands perform a logical AND operation of the parameter (mask) and internal register. The mask 0 is reset to register bit, and 1 to cause no change.

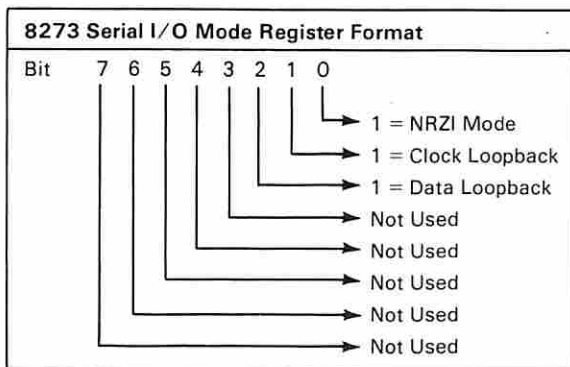
The following are descriptions of each bit of the operating, serial I/O, one-bit delay, and data transfer mode registers.

Operating Mode Register



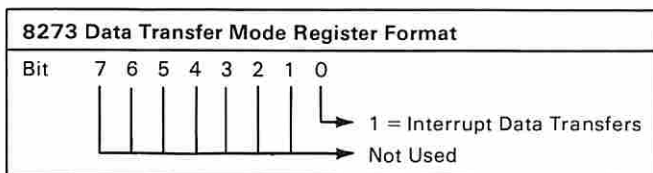
- Bit 0 If bit 0 is set to a 1, flags are sent immediately if the transmitter was idle when the bit was set. If a transmit or transmit-transparent command was active, flags are sent immediately after transmit completion. This mode is ignored if loop transmit is active or the one-bit-delay mode register is set for one-bit delay. If bit 0 is reset (to 0), the transmitter sends idles on the next character boundary if idle or, after transmission is complete, if the transmitter was active at bit-0 reset time.
- Bit 1 If bit 1 is set to a 1, the 8273 sends two characters before the first flag of a frame. These characters are hex 00 if NRZI is set or hex 55 if NRZI is not set. (See "Serial I/O Mode Register," for NRZI encoding mode format.)
- Bit 2 If bit 2 is set to a 1, the 8273 buffers the first two bytes of a received frame (the bytes are not passed to memory). Resetting this bit (to 0) causes these bytes to be passed to and from memory.
- Bit 3 This bit indicates to the 8273 when to generate an end-of-frame interrupt. If bit 3 is set, an early interrupt is generated when the last data character has been passed to the 8273. If the processor responds to the early interrupt with another transmit command before the final flag is sent, the final-flag interrupt will not be generated and a new frame will begin when the current frame is complete. Thus, frames may be sent separated by a single flag. A reset condition causes an interrupt to be generated only following a final flag.
- Bit 4 This is the EOP-interrupt-mode function and is not used on the SDLC communications adapter. This bit should always be in the reset condition.
- Bit 5 This bit is always reset for SDLC operation, which causes the 8273 protocol controller to recognize eight ones (0 1 1 1 1 1 1 1) as an abort character.

Serial I/O Mode Register



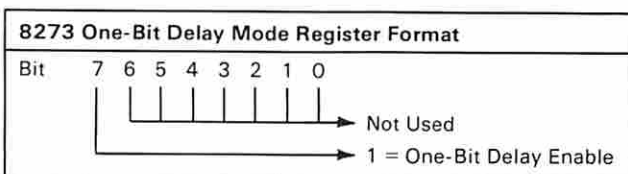
- Bit 0 Set to 1, this bit specifies NRZI encoding and decoding. Resetting this bit specifies that transmit and receive data be treated as a normal positive-logic bit stream.
- Bit 1 When bit 1 is set to 1, the transmit clock is internally routed to the receive-clock circuitry. It is normally used with the loopback bit (bit 2). The reset condition causes the transmit and receive clocks to be routed to their respective 8273 I/O pins.
- Bit 2 When bit 2 is set, the transmitted data is internally routed to the received data circuitry. The reset condition causes the transmitted and received data to be routed to their respective 8273 I/O pins.

Data Transfer Mode Register



When the data transfer mode register is set, the 8273 protocol controller will interrupt when data bytes are required for transmission, or are available from a reception. If a transmit or receive interrupt occurs and the status register indicates that there is no transmit or receive interrupt result, the interrupt is a transmit or receive data request, respectively. Reset of this register causes DMA requests to be performed with no interrupts to the processor.

One-Bit Delay Mode Register



When one-bit delay is set, the 8273 retransmits the received data stream one-bit delayed. Reset of this bit stops the one-bit delay mode.

The table below is a summary of all set and reset commands associated with the 8273 mode registers. The set or reset mask used to define individual bits is treated as a single parameter. No result or interrupt is generated by the 8273 after execution of these commands.

Register	Command	Hex Code	Parameter
One-Bit Delay Mode	Set	A4	Set Mask
	Reset	64	Reset Mask
Data Transfer Mode	Set	97	Set Mask
	Reset	57	Reset Mask
Operating Mode	Set	91	Set Mask
	Reset	51	Reset Mask
Serial I/O Mode	Set	A0	Set Mask
	Reset	60	Reset Mask

8273 SDLC Protocol Controller Mode Register Commands

Command Phase

Although the 8273 is a full duplex device, there is only one command register. Thus, the command register must be used for only one command sequence at a time and the transmitter and receiver may never be simultaneously in a command phase.

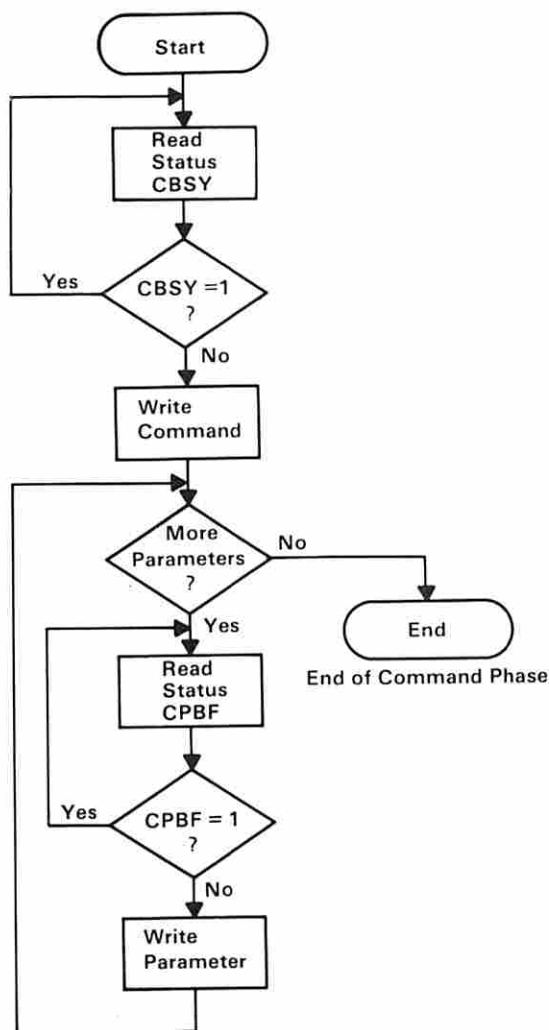
The system software starts the command phase by selecting the 8273 command register address and writing a command byte into the register. The following table lists command and parameter information for the 8273 protocol controller. If further information is required by the 8273 prior to execution of the command, the system software must write this information into the parameter register.

Command Description	Command (Hex)	Parameter	Results	Result Port	Completion Interrupt
Set One-Bit Delay	A4	Set Mask	None	—	No
Reset One-Bit Delay	64	Reset Mask	None	—	No
Set Data Transfer Mode	97	Set Mask	None	—	No
Reset Data Transfer Mode	57	Reset Mask	None	—	No
Set Operating Mode	91	Set Mask	None	—	No
Reset Operating Mode	51	Reset Mask	None	—	No
Set Serial I/O Mode	A0	Set Mask	None	—	No
Reset Serial I/O Mode	60	Reset Mask	None	—	No
General Receive	C0	80,81	RIC,R0,R1,A,C	RXI/R	Yes
Selective Receive	C1	80,81,A1,A2	RIC,R0,R1,A,C	RXI/R	Yes
Receive Disable	C5	None	None	—	No
Transmit Frame	C8	L0,L1,A,C	TIC	TXI/R	Yes
Transmit Transparent	C9	L0,L1	TIC	TXI/R	Yes
Abort Transmit Frame	CC	None	TIC	TXI/R	Yes
Abort Transmit Transparent	CD	None	TIC	TXI/R	Yes
Read Port A	22	None	Port Value	Result	No
Read Port B	23	None	Port Value	Result	No
Set Port B Bit	A3	Set Mask	None	—	No
Reset Port B Bit	63	Reset Mask	None	—	No

8273 Command Summary Key

- B0** — Least significant byte of the receiver buffer length.
B1 — Most significant byte of the receiver buffer length.
L0 — Least significant byte of the Tx frame length.
L1 — Most significant byte of the Tx frame length.
A1 — Receive frame address match field one.
A2 — Receive frame address match field two.
A — Address field of received frame. If non-buffered mode is specified, this result is not provided.
C — Control field of received frame. If non-buffered mode is specified, this result is not provided.
RXI/R — Receive interrupt result register.
TXI/R — Transmit interrupt result register.
R0 — Least significant byte of the length of the frame received.
R1 — Most significant byte of the length of the frame received.
RIC — Receiver interrupt result code.
TIC — Transmitter interrupt result code.

A flowchart of the command phase is shown below. Handshaking of the command and parameter bytes is accomplished by the CBSY and CPBF bits of the status register. A command may not be written if the 8273 is busy (CBSY = 1). The original command will be overwritten if a second command is issued while CBSY = 1. The flowchart also indicates a parameter buffer full check. The processor must wait until CPBF = 0 before writing a parameter to the parameter register. Previous parameters are overwritten and lost if a parameter is written while CPBF = 1.



Execution Phase

During the execution phase, the operation specified by the command phase is performed. If DMA is utilized for data transfers, no processor involvement is required.

For interrupt-driven transfers the 8273 raises the appropriate INT pin (TxINT or RxINT). When the processor responds to the interrupt, it must determine the cause by examining the status register and the associated IRA (interrupt result available) bit of the status register. If IRA = 0, the interrupt is a data transfer request. If IRA = 1, an operation is complete and the associated interrupt result register must be read to determine completion status.

Result Phase

During the result phase, the 8273 notifies the processor of the outcome of a command execution. This phase is initiated by either a successful completion or error detection during execution.

Some commands such as reading or writing the I/O ports provide immediate results. These results are made available to the processor in the 8273 result register. Presence of a valid immediate result is indicated by the CRBF (command result buffer full) bit of the status register.

Non-immediate results deal with the transmitter and receiver. These results are provided in the TxI/R (transmit interrupt result) or RxI/R (receiver interrupt result) registers, respectively. The 8273 notifies the processor that a result is available with the TxIRA and RxIRA bits of the status register. Results consist of one-byte result interrupt code indicating the condition for the interrupt and, if required, one or more bytes supplying additional information. The "Result Code Summary" table later in this section provides information on the format and decode of the transmitter and receiver results.

The following are typical frame transmit and receive sequences. These examples assume DMA is utilized for data transfer operations.

Transmit

Before a frame can be transmitted, the DMA controller is supplied, by the communication software, the starting address for the desired information field. The 8273 is then commanded to transmit a frame (by issuing a transmit frame command).

After a command, but before transmission begins, the 8273 needs some more information (parameters). Four parameters are required for the transmit frame command; the frame address field byte, the frame control field byte, and two bytes which are the least significant and most significant bytes of the information field byte length. Once all four parameters are loaded, the 8273 makes RTS (request to send) active and waits for CTS (clear to send) to go active from the modem interface. Once CTS is active, the 8273 starts the frame transmission. While the 8273 is transmitting the opening flag, address field, and control field, it starts making transmitter DMA requests. These requests continue at character (byte) boundaries until the pre-loaded number of bytes of information field have been transmitted. At this point, the requests stop, the FCS (frame check sequence) and closing flag are transmitted, and the TxINT line is raised, signaling the processor the frame transmission is complete and the result should be read. Note that after the initial command and parameter loading, no processor intervention was required (since DMA is used for data transfers) until the entire frame was transmitted.

General Receive

Receiver operation is very similar. Like the initial transmit sequence, the processor's DMA controller is loaded with a starting address for a receive data buffer and the 8273 is commanded to receive. Unlike the transmitter, there are two different receive commands; a general receive, where all received frames are transferred to memory, and selective receive, where only frames having an address field matching one of two preprogrammed 8273 address fields are transferred to memory. (This example covers a general receive operation.) After the receive command, two parameters are required before the receiver becomes active; the least significant and most significant bytes of the receiver buffer length. Once these bytes are loaded,

the receiver is active and the processor may return to other tasks. The next frame appearing at the receiver input is transferred to memory using receiver DMA requests. When the closing flag is received, the 8273 checks the FCS and raises its RxINT line. The processor can then read the results, which indicate if the frame was error-free or not. (If the received frame had been longer than the pre-loaded buffer length, the processor would have been notified of that occurrence earlier with a receiver error interrupt. Like the transmit example, after the initial command, the processor is free for other tasks until a frame is completely received.

Selective Receive

In selective receive, two parameters (A1 and A2) are required in addition to those for general receive. These parameters are two address match bytes. When commanded to selective receive, the 8273 passes to memory or the processor only those frames having an address field matching either A1 or A2. This command is usually used for secondary stations with A1 designating the secondary address and A2 being the "all parties" address. If only one match byte is needed, A1 and A2 should be equal. As in general receive, the 8273 counts the incoming data bytes and interrupts the processor if the received frame is larger than the preset receive buffer length.

Result Code Summary

	Hex Code	Result	Status After Interrupt
T r a n s m i t	0C	Early Transmit Interrupt	Transmitter Active
	0D	Frame Transmit Complete	Idle or Flags
	0E	DMA Underrun	Abort
	0F	Clear to Send Error	Abort
	10	Abort Complete	Idle or Flags
R e c e i v e	X0	A1 Match or General Receive	Active
	X1	A2 Match	Active
	03	CRC Error	Active
	04	Abort Detected	Active
	05	Idle Detected	Disabled
	06	EOP Detected	Disabled
	07	Frame Less Than 32 Bits	Active
	08	DMA Overrun	Disabled
	09	Memory Buffer Overflow	Disabled
	0A	Carrier Detect Failure	Disabled
	0B	Receiver Interrupt Overrun	Disabled
Note: X decodes to number of bits in partial byte received.			

The first two codes in the receive result code table result from the error free reception of a frame. Since SDLC allows frames of arbitrary length (>32 bits), the high order bits of the receive result report the number of valid received bits in the last received information field byte. The chart below shows the decode of this receive result bit.

X	Bits Received in Last Byte
E	All Eight Bits of Last Byte
0	Bit0 Only
8	Bit1-Bit0
4	Bit2-Bit0
C	Bit3-Bit0
2	Bit4-Bit0
A	Bit5-Bit0
6	Bit6-Bit0

Address and Interrupt Information

The following tables provide address and interrupt information for the SDLC adapter:

Hex Code	Device	Register Name	Function
380	8255	Port A Data	Internal/External Sensing
381	8255	Port B Data	External Modem Interface
382	8255	Port C Data	Internal Control
383	8255	Mode Set	8255 Mode Initialization
384	8253	Counter 0 LSB	Square Wave Generator
384	8253	Counter 0 MSB	Square Wave Generator
385	8253	Counter 1 LSB	Inactivity Time-Outs
385	8253	Counter 1 MSB	Inactivity Time-Outs
386	8253	Counter 2 LSB	Inactivity Time-Outs
386	8253	Counter 2 MSB	Inactivity Time-Outs
387	8253	Mode Register	8253 Mode Set
388	8273	Command/Status	Out=Command In=Status
389	8273	Parameter/Result	Out=Parameter In=Status
38A	8273	Transmit INT Status	DMA/INT
38B	8273	Receive INT Status	DMA/INT
38C	8273	Data	DPC (Direct Program Control)

SDLC Communications Adapter Device Addresses

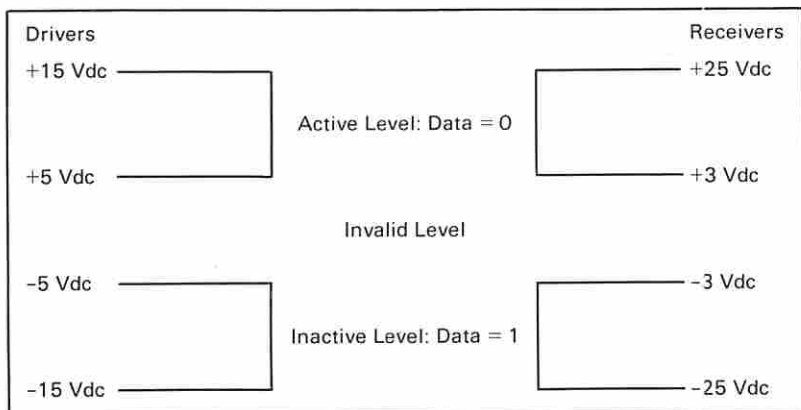
Interrupt Level 3	Transmit/Receive Interrupt
Interrupt Level 4	Timer 1 Interrupt Timer 2 Interrupt Clear to Send Changed Data Set Ready Changed
DMA Level One is used for Transmit and Receive	

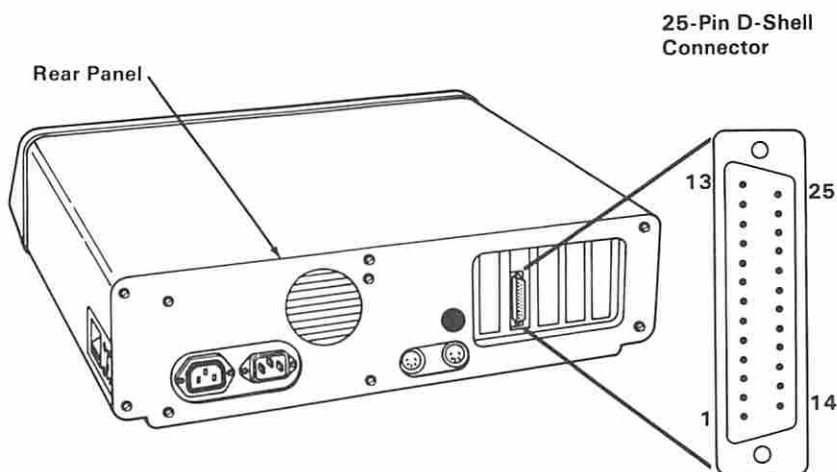
Interrupt Information

Interface Information

The SDLC communications adapter conforms to interface signal levels standardized by the Electronics Industries Association RS-232C Standard. These levels are shown in the figure below.

Additional lines used but not standardized by EIA are pins 11, 18, and 25. These lines are designated as select standby, test and test indicate, respectively. Select Standby is used to support the switched network backup facility of a modem providing this option. Test and test indicate support a modem wrap function on modems which are designed for business machine controlled modem wraps. Two jumpers on the adapter (P1 and P2) are used to connect test and test indicate to the interface, if required (see Appendix D for these jumpers).





Signal Name — Description		Pin
External Device	No Connection	1
	Transmitted Data ←	2
	Received Data →	3
	Request to Send ←	4
	Clear to Send →	5
	Data Set Ready →	6
	Signal Ground	7
	Received Line Signal Detector →	8
	No Connection	9
	No Connection	10
	Select Standby* ←	11
	No Connection	12
	No Connection	13
	No Connection	14
	Transmitter Signal Element Timing →	15
	No Connection	16
	Receiver Signal Element Timing →	17
	Test (IBM Modems Only)* ←	18
	No Connection	19
	Data Terminal Ready ←	20
	No Connection	21
	Ring Indicator →	22
	Data Signal Rate Selector ←	23
	No Connection	24
	Test Indicate (IBM Modems Only)* →	25

Synchronous
Data Link
Control
Communications
Adapter

*Not standardized by EIA (Electronics Industry Association).

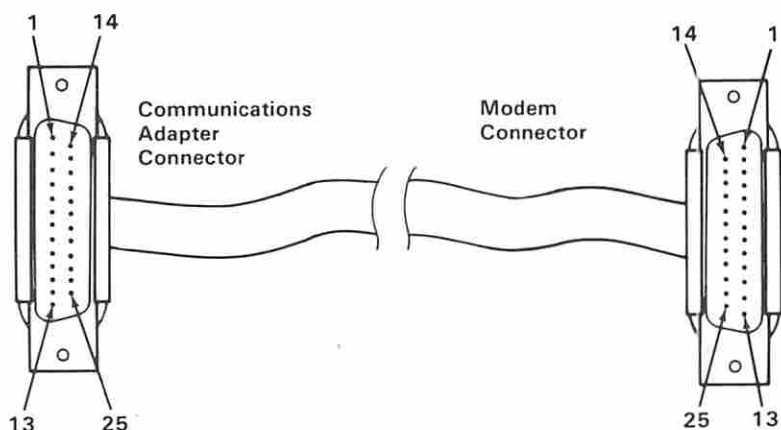
Notes:

IBM Communications Adapter Cable

The IBM Communications Adapter Cable is a ten foot cable for connection of an IBM communications adapter to a modem or other RS-232C DCE (data communications equipment). It is fully shielded and provides a high quality, low noise channel for interface between the communications adapter and DCE.

The connector ends are 25-pin D-shell connectors. All pin connections conform with the EIA RS-232C standard. In addition, connection is provided on pins 11, 18 and 25. These pins are designated as select standby, test and test indicate, respectively, on some modems. Select standby is used to support the switched network backup facility, if applicable. Test and test indicate support a modem wrap function on modems designed for business machine controlled modem wraps.

The IBM Communications Adapter Cable connects the following pins on the 25-pin D-shell connectors.



Communications Adapter Connector Pin #	Name	Modem Connector Pin #
NC	Outer Cable Shield	1
2	Transmitted Data	2
3	Received Data	3
4	Request to Send	4
5	Clear to Send	5
6	Data Set Ready	6
7	Signal Ground (Inner Lead Shields)	7
8	Received Line Signal Detector	8
NC		NC
NC		NC
11	Select Standby	11
NC		NC
NC		NC
NC		NC
15	Transmitter Signal Element Timing	15
NC		NC
17	Receiver Signal Element Timing	17
18	Test	18
NC		NC
20	Data Terminal Ready	20
NC		NC
22	Ring Indicator	22
23	Data Signal Rate Selector	23
NC		NC
25	Test Indicate	25

Connector Specifications

1-258 Communications Cable

SECTION 2: ROM BIOS AND SYSTEM USAGE

ROM BIOS	2-2
Keyboard Encoding and Usage	2-11
BIOS Cassette Logic	2-21

ROM BIOS

The basic input/output system (BIOS) resides in ROM on the system board and provides device level control for the major I/O devices in the system. Additional ROM modules may be located on option adapters to provide device level control for that option adapter. BIOS routines enable the assembly language programmer to perform block (disk and diskette) or character-level I/O operations without concern for device address and operating characteristics. System services, such as time-of-day and memory size determination, are provided by the BIOS.

The goal is to provide an operational interface to the system and relieve the programmer of the concern about the characteristics of hardware devices. The BIOS interface insulates the user from the hardware, thus allowing new devices to be added to the system, yet retaining the BIOS level interface to the device. In this manner, user programs become transparent to hardware modifications and enhancements.

The IBM Personal Computer MACRO Assembler manual and the IBM Personal Computer Disk Operating System (DOS) manual provide useful programming information related to this section. A complete listing of the BIOS is given in Appendix A.

Use of BIOS

Access to BIOS is through the 8088 software interrupts. Each BIOS entry point is available through its own interrupt, which can be found in the "8088 Software Interrupt Listing."

The software interrupts, hex 10 through hex 1A, each access a different BIOS routine. For example, to determine the amount of memory available in the system,

INT 12H

will invoke the BIOS routine for determining memory size and will return the value to the caller.

Parameter Passing

All parameters passed to and from the BIOS routines go through the 8088 registers. The prolog of each BIOS function indicates the registers used on the call and the return. For the memory size example, no parameters are passed. The memory size, in 1K byte increments, is returned in the AX register.

If a BIOS function has several possible operations, the AH register is used at input to indicate the desired operation. For example, to set the time of day, the following code is required:

```
MOV  AH,1           ;function is to set time of day.
MOV  CX,HIGH_COUNT ;establish the current time.
MOV  DX,LOW_COUNT
INT  1AH           ;set the time.
```

To read the time of day:

```
MOV  AH,0           ;function is to read time of day.
INT  1AH           ;read the timer.
```

Generally, the BIOS routines save all registers except for AX and the flags. Other registers are modified on return only if they are returning a value to the caller. The exact register usage can be seen in the prolog of each BIOS function.

Address (Hex)	Interrupt Number	Name	BIOS Entry
0-3	0	Divide by Zero	D_EOI
4-7	1	Single Step	D_EOI
8-B	2	Nonmaskable	NMI_INT
C-F	3	Breakpoint	D_EOI
10-13	4	Overflow	D_EOI
14-17	5	Print Screen	PRINT_SCREEN
18-1B	6	Reserved	D_EOI
1D-1F	7	Reserved	D_EOI
20-23	8	Time of Day	TIMER_INT
24-27	9	Keyboard	KB_INT
28-2B	A	Reserved	D_EOI
2C-2F	B	Communications	D_EOI
30-33	C	Communications	D_EOI
34-37	D	Disk	D_EOI
38-3B	E	Diskette	DISK_INT
3C-3F	F	Printer	D_EOI
40-43	10	Video	VIDEO_IO
44-47	11	Equipment Check	EQUIPMENT
48-4B	12	Memory	MEMORY_SIZE_DETERMINE
4C-4F	13	Diskette/Disk	DISKETTE_IO
50-53	14	Communications	RS232_IO
54-57	15	Cassette	CASSETTE_IO
58-5B	16	Keyboard	KEYBOARD_IO
5C-5F	17	Printer	PRINTER_IO
60-63	18	Resident BASIC	F600:0000
64-67	19	Bootstrap	BOOT_STRAP
68-6B	1A	Time of Day	TIME_OF_DAY
6C-6F	1B	Keyboard Break	DUMMY_RETURN
70-73	1C	Timer Tick	DUMMY_RETURN
74-77	1D	Video Initialization	VIDEO_PARAMS
78-7B	1E	Diskette Parameters	DISK_BASE
7C-7F	1F	Video Graphics Chars	0

8088 Software Interrupt Listing

Vectors with Special Meanings

Interrupt Hex 1B - Keyboard Break Address

This vector points to the code to be exercised when the Ctrl and Break keys are pressed on the keyboard. The vector is invoked while responding to the keyboard interrupt, and control should be returned through an IRET instruction. The power-on routines initialize this vector to point to an IRET instruction, so that nothing will occur when the Ctrl and Break keys are pressed unless the application program sets a different value.

Control may be retained by this routine, with the following problems. The Break may have occurred during interrupt processing, so that one or more End of Interrupt commands must be sent to the 8259 controller. Also, all I/O devices should be reset in case an operation was underway at that time.

Interrupt Hex 1C - Timer Tick

This vector points to the code to be executed on every system-clock tick. This vector is invoked while responding to the timer interrupt, and control should be returned through an IRET instruction. The power-on routines initialize this vector to point to an IRET instruction, so that nothing will occur unless the application modifies the pointer. It is the responsibility of the application to save and restore all registers that will be modified.

Interrupt Hex 1D - Video Parameters

This vector points to a data region containing the parameters required for the initialization of the 6845 on the video card. Note that there are four separate tables, and all four must be reproduced if all modes of operation are to be supported. The power-on routines initialize this vector to point to the parameters contained in the ROM video routines.

Interrupt Hex 1E - Diskette Parameters

This vector points to a data region containing the parameters required for the diskette drive. The power-on routines initialize the vector to point to the parameters contained in the ROM diskette routine. These default parameters represent the specified values for any IBM drives attached to the machine. Changing this parameter block may be necessary to reflect the specifications of the other drives attached.

Interrupt Hex 1F - Graphics Character Extensions

When operating in the graphics modes of the IBM Color/Graphics Monitor Adapter (320 by 200 or 640 by 200), the read/write character interface will form the character from the ASCII code point, using a set of dot patterns. The dot patterns for the first 128 code points are contained in ROM. To access the second 128 code points, this vector must be established to point at a table of up to 1K bytes, where each code point is represented by eight bytes of graphic information. At power-on, this vector is initialized to 000:0, and it is the responsibility of the user to change this vector if the additional code points are required.

Interrupt Hex 40 - Reserved

When an IBM Fixed Disk Drive Adapter is installed, the BIOS routines use interrupt hex 40 to revector the diskette pointer.

Interrupt Hex 41 - Fixed Disk Parameters

This vector points to a data region containing the parameters required for the fixed disk drive. The power-on routines initialize the vector to point to the parameters contained in the ROM disk routine. These default parameters represent the specified values for any IBM Fixed Disk Drives attached to the machine. Changing this parameter block may be necessary to reflect the specifications of the other fixed disk drives attached.

Other Read/Write Memory Usage

The IBM BIOS routines use 256 bytes of memory starting at absolute hex 400 to hex 4FF. Locations hex 400 to 407 contain the base addresses of any RS-232C cards attached to the system. Locations hex 408 to 40F contain the base addresses of the printer adapter.

Memory locations hex 300 to 3FF are used as a stack area during the power-on initialization, and bootstrap, when control is passed to it from power-on. If the user desires the stack in a different area, the area must be set by the application.

Address (Hex)	Interrupt (Hex)	Function
80-83	20	DOS Program Terminate
84-87	21	DOS Function Call
88-8B	22	DOS Terminate Address
8C-8F	23	DOS Ctrl Break Exit Address
90-93	24	DOS Fatal Error Vector
94-97	25	DOS Absolute Disk Read
98-9B	26	DOS Absolute Disk Write
9C-9F	27	DOS Terminate, Fix In Storage
A0-FF	28-3F	Reserved for DOS
100-17F	40-5F	Reserved
180-19F	60-67	Reserved for User Software Interrupts
1A0-1FF	68-7F	Not Used
200-217	80-85	Reserved by BASIC
218-3C3	86-F0	Used by BASIC Interpreter while BASIC is running
3C4-3FF	F1-FF	Not Used

BASIC and DOS Reserved Interrupts

Address (Hex)	Mode	Function
400-48F 490-4EF 4F0-4FF	ROM BIOS	See BIOS Listing Reserved Reserved as Intra-Application Communication Area for any application
500-5FF 500	DOS	Reserved for DOS and BASIC Print Screen Status Flag Store 0-Print Screen Not Active or Successful Print Screen Operation 1-Print Screen In Progress 255-Error Encountered during Print Screen Operation
504	DOS	Single Drive Mode Status Byte
510-511	BASIC	BASIC's Segment Address Store
512-515	BASIC	Clock Interrupt Vector Segment: Offset Store
516-519	BASIC	Break Key Interrupt Vector Segment: Offset Store
51A-51D	BASIC	Disk Error Interrupt Vector Segment: Offset Store

Reserved Memory Locations

If you do DEF SEG (Default workspace segment):

	Offset (Hex Value)	Length
Line number of current line being executed	2E	2
Line number of last error	347	2
Offset into segment of start of program text	30	2
Offset into segment of start of variables (end of program text 1-1)	358	2
Keyboard buffer contents if 0-no characters in buffer if 1-characters in buffer	6A	1
Character color in graphics mode Set to 1, 2, or 3 to get text in colors 1 to 3. Do not set to 0. (Default = 3)	4E	1
Example 100 Print PEEK (&H2E) + 256*PEEK (&H2F) $\int$ 100 L Hex 64 H Hex 00		

BASIC Workspace Variables

2-8 ROM BIOS

Starting Address in Hex

00000	BIOS Interrupt Vectors
00080	Available Interrupt Vectors
00400	BIOS Data Area
00500	User Read/Write Memory
C8000	Disk Adapter
F0000	Read Only Memory
FE000	Bios Program Area

BIOS Memory Map**BIOS Programming Hints**

The BIOS code is invoked through software interrupts. The programmer should not "hard code" BIOS addresses into applications. The internal workings and absolute addresses within BIOS are subject to change without notice.

If an error is reported by the disk or diskette code, you should reset the drive adapter and retry the operation. A specified number of retries should be required on diskette reads to ensure the problem is not due to motor start-up.

When altering I/O port bit values, the programmer should change only those bits which are necessary to the current task. Upon completion, the programmer should restore the original environment. Failure to adhere to this practice may be incompatible with present and future applications.

Adapter Cards with System-Accessible ROM Modules

The ROM BIOS provides a facility to integrate adapter cards with on board ROM code into the system. During the POST, interrupt vectors are established for the BIOS calls. After the default vectors are in place, a scan for additional ROM modules takes place. At this point, a ROM routine on the adapter card may gain control. The routine may establish or intercept interrupt vectors to hook themselves into the system.

The absolute addresses hex C8000 through hex F4000 are scanned in 2K blocks in search of a valid adapter card ROM. A valid ROM is defined as follows:

- Byte 0: Hex 55
- Byte 1: Hex AA
- Byte 2: A length indicator representing the number of 512 byte blocks in the ROM ($\text{length}/512$).
A checksum is also done to test the integrity of the ROM module. Each byte in the defined ROM is summed modulo hex 100. This sum must be 0 for the module to be deemed valid.

When the POST identifies a valid ROM, it does a far call to byte 3 of the ROM (which should be executable code). The adapter card may now perform its power-on initialization tasks. The feature ROM should return control to the BIOS routines by executing a far return.

Keyboard Encoding and Usage

Encoding

The keyboard routine provided by IBM in the ROM BIOS is responsible for converting the keyboard scan codes into what will be termed "extended ASCII." The extended ASCII codes returned by the ROM routine are mapped to a United States keyboard layout. Some operating systems may make provision for alternate keyboard layouts by providing an "interrupt replacer" which resides in read/write memory. This section documents the ROM routine only.

Extended ASCII encompasses one-byte character codes with possible values of 0 to 255, an extended code for certain extended keyboard functions, and functions handled within the keyboard routine or through interrupts.

Character Codes

The following character codes are passed through the BIOS keyboard routine to the system or application program. A "-1" means the combination is suppressed in the keyboard routine. The codes are returned in AL. See Appendix C for the exact codes. Also, see "Keyboard Scan Code Diagram" in Section 1.

Key Number	Base Case	Upper Case	Ctrl	Alt
1	Esc	Esc	Esc	-1
2	1	½	-1	Note 1
3	2	@	Nul (000) Note 1	Note 1
4	3	#	-1	Note 1
5	4	\$	-1	Note 1
6	5	%	-1	Note 1
7	6	^	RS(030)	Note 1
8	7	&	-1	Note 1
9	8	*	-1	Note 1
10	9	(	-1	Note 1
11	0	)	-1	Note 1
12	-	—	US(031)	Note 1
13	=	+	-1	Note 1
14	Backspace (008)	Backspace (008)	Del (127)	-1
15	→ (009)	← (Note 1)	-1	-1
16	q	Q	DC1 (017)	Note 1
17	w	W	ETB (023)	Note 1

Key Number	Base Case	Upper Case	Ctrl	Alt
18	e	E	ENQ (005)	Note 1
19	r	R	DC2 (018)	Note 1
20	t	T	DC4 (020)	Note 1
21	y	Y	EM (025)	Note 1
22	u	U	NAK (021)	Note 1
23	i	I	HT (009)	Note 1
24	o	O	SI (015)	Note 1
25	p	P	DLE (016)	Note 1
26	[	{	Esc (027)	-1
27	]	}	GS (029)	-1
28	CR	CR	LF (010)	-1
29 Ctrl	-1	-1	-1	-1
30	a	A	SOH (001)	Note 1
31	s	S	DC3 (019)	Note 1
32	d	D	EOT (004)	Note 1
33	f	F	ACK (006)	Note 1
34	g	G	BEL (007)	Note 1
35	h	H	BS (008)	Note 1
36	j	J	LF (010)	Note 1
37	k	K	VT (011)	Note 1
38	l	L	FF (012)	Note 1
39	;	:	-1	-1
40	'	"	-1	-1
41	`	~	-1	-1
42 Shift	-1	-1	-1	-1
43	\		FS (028)	-1
44	z	Z	SUB (026)	Note 1
45	x	X	CAN (024)	Note 1
46	c	C	ETX (003)	Note 1
47	v	V	SYN (022)	Note 1
48	b	B	STX (002)	Note 1
49	n	N	SO (014)	Note 1
50	m	M	CR (013)	Note 1
51	,	<	-1	-1
52	.	>	-1	-1
53	/	?	-1	-1
54 Shift	-1	-1	-1	-1
55	*	(Note 2)	(Note 1)	-1
56 Alt	-1	-1	-1	-1
57	SP	SP	SP	SP
58 Caps Lock	-1	-1	-1	-1
59	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
60	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
61	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
62	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
63	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
64	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)

Character Codes (Part 2 of 3)

2-12 Keyboard Encoding

Key Number	Base Case	Upper Case	Ctrl	Alt
65	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
66	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
67	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
68	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
69 Num Lock	-1	-1	Pause (Note 2)	-1
70 Scroll Lock	-1	-1	Break (Note 2)	-1

Notes: 1. Refer to "Extended Codes" in this section.
2. Refer to "Special Handling" in this section.

Character Codes (Part 3 of 3)

Keys 71 to 83 have meaning only in base case, in Num Lock (or shifted) states, or in Ctrl state. It should be noted that the shift key temporarily reverses the current Num Lock state.

Key Number	Num Lock	Base Case	Alt	Ctrl
71	7	Home (Note 1)	-1	Clear Screen
72	8	↑ (Note 1)	-1	-1
73	9	Page Up (Note 1)	-1	Top of Text and Home
74	-	-----	-1	-1
75	4	← (Note 1)	-1	Reverse Word (Note 1)
76	5	-1	-1	-1
77	6	→ (Note 1)	-1	Advance Word (Note 1)
78	+	+	-1	-1
79	1	End (Note 1)	-1	Erase to EOL (Note 1)
80	2	↓ (Note 1)	-1	-1
81	3	Page Down (Note 1)	-1	Erase to EOS (Note 1)
82	0	Ins	-1	-1
83	.	Del (Notes 1,2)	Note 2	Note 2

Notes: 1. Refer to "Extended Codes" in this section.
2. Refer to "Special Handling" in this section.

Extended Codes

Extended Functions

For certain functions that cannot be represented in the standard ASCII code, an extended code is used. A character code of 000 (Nul) is returned in AL. This indicates that the system or application program should examine a second code that will indicate the actual function. Usually, but not always, this second code is the scan code of the primary key that was pressed. This code is returned in AH.

Second Code	Function
3	Nul Character
15	←
16-25	Alt Q, W, E, R, T, Y, U, I, O, P
30-38	Alt A, S, D, F, G, H, J, K, L
44-50	Alt Z, X, C, V, B, N, M
59-68	F1 to F10 Function Keys Base Case
71	Home
72	↑
73	Page Up and Home Cursor
75	←
77	→
79	End
80	↓
81	Page Down and Home Cursor
82	Ins (Insert)
83	Del (Delete)
84-93	F11 to F20 (Uppercase F1 to F10)
94-103	F21 to F30 (Ctrl F1 to F10)
104-113	F31 to F40 (Alt F1 to F10)
114	Ctrl PrtSc (Start/Stop Echo to Printer)
115	Ctrl ← (Reverse Word)
116	Ctrl → (Advance Word)
117	Ctrl End [Erase to End of Line (EOL)]
118	Ctrl PgDn [Erase to End of Screen (EOS)]
119	Ctrl Home (Clear Screen and Home)
120-131	Alt 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, -, = (Keys 2-13)
132	Ctrl PgUp (Top 25 Lines of Text and Home Cursor)

Keyboard Extended Functions

2-14 Keyboard Encoding

Shift States

Most shift states are handled within the keyboard routine, transparent to the system or application program. In any case, the current set of active shift states are available by calling an entry point in the ROM keyboard routine. The following keys result in altered shift states:

Shift

This key temporarily shifts keys 2-13, 15-27, 30-41, 43-53, 55, and 59-68 to upper case (base case if in Caps Lock state). Also, the Shift key temporarily reverses the Num Lock or non-Num-Lock state of keys 71-73, 75, 77, and 79-83.

Ctrl

This key temporarily shifts keys 3, 7, 12, 14, 16-28, 30-38, 43-50, 55, 59-71, 73, 75, 77, 79, and 81 to the Ctrl state. Also, the Ctrl key is used with the Alt and Del keys to cause the "system reset" function, with the Scroll Lock key to cause the "break" function, and with the Num Lock key to cause the "pause" function. The system reset, break, and pause functions are described in "Special Handling" on the following pages.

Alt

The key temporarily shifts keys 2-13, 16-25, 30-38, 44-50, and 59-68 to the Alt state. Also, the Alt key is used with the Ctrl and Del keys to cause the "system reset" function described in "Special Handling" on the following pages.

The Alt key has another use. This key allows the user to enter any character code from 0 to 255 into the system from the keyboard. The user holds down the Alt key and types the decimal value of the characters desired using the numeric keypad (keys 71-73, 75-77, and 79-82). The Alt key is then released. If more than three digits are typed, a modulo-256 result is created. These three digits are interpreted as a character code and are transmitted through the keyboard routine to the system or application program. Alt is handled internal to the keyboard routine.

Caps Lock

This key shifts keys 16-25, 30-38, and 44-50 to upper case. A second depression of the Caps Lock key reverses the action. Caps Lock is handled internal to the keyboard routine.

Scroll Lock

This key is interpreted by appropriate application programs as indicating use of the cursor-control keys should cause windowing over the text rather than cursor movement. A second depression of the Scroll Lock key reverses the action. The keyboard routine simply records the current shift state of the Scroll Lock key. It is the responsibility of the system or application program to perform the function.

Shift Key Priorities and Combinations

If combinations of the Alt, Ctrl, and Shift keys are pressed and only one is valid, the precedence is as follows: the Alt key is first, the Ctrl key is second, and the Shift key is third. The only valid combination is Alt and Ctrl, which is used in the "system reset" function.

Special Handling

System Reset

The combination of the Alt, Ctrl, and Del keys will result in the keyboard routine initiating the equivalent of a "system reset" or "reboot." System reset is handled internal to the keyboard.

Break

The combination of the Ctrl and Break keys will result in the keyboard routine signaling interrupt hex 1A. Also, the extended characters (AL = hex 00, AH = hex 00) will be returned.

Pause

The combination of the Ctrl and Num Lock keys will cause the keyboard interrupt routine to loop, waiting for any key except the Num Lock key to be pressed. This provides a system- or application-transparent method of temporarily suspending list, print, and so on, and then resuming the operation. The "unpause" key is thrown away. Pause is handled internal to the keyboard routine.

Print Screen

The combination of the Shift and PrtSc (key 55) keys will result in an interrupt invoking the print screen routine. This routine works in the alphanumeric or graphics mode, with unrecognizable characters printing as blanks.

Other Characteristics

The keyboard routine does its own buffering. The keyboard buffer is large enough to support a fast typist. However, if a key is entered when the buffer is full, the key will be ignored and the "bell" will be sounded.

Also, the keyboard routine suppresses the typematic action of the following keys: Ctrl, Shift, Alt, Num Lock, Scroll Lock, Caps Lock, and Ins.

Keyboard Usage

This section is intended to outline a set of guidelines of key usage when performing commonly used functions.

Function	Key(s)	Comment
Home Cursor	Home	Editors; word processors
Return to outermost menu	Home	Menu driven applications
Move cursor up	↑	Full screen editor, word processor
Page up, scroll backward 25 lines and home	PgUp	Editors; word processors
Move cursor left	← Key 75	Text, command entry
Move cursor right	→	Text, command entry
Scroll to end of text Place cursor at end of line	End	Editors; word processors
Move cursor down	↓	Full screen editor, word processor
Page down, scroll forward 25 lines and home	Pg Dn	Editors; word processors
Start/Stop insert text at cursor, shift text right in buffer	Ins	Text, command entry
Delete character at cursor	Del	Text, command entry
Destructive backspace	← Key 14	Text, command entry
Tab forward	→	Text entry
Tab reverse	←	Text entry
Clear screen and home	Ctrl Home	Command entry
Scroll up	↑	In scroll lock mode
Scroll down	↓	In scroll lock mode
Scroll left	←	In scroll lock mode
Scroll right	→	In scroll lock mode
Delete from cursor to EOL	Ctrl End	Text, command entry
Exit/Escape	Esc	Editor, 1 level of menu, and so on
Start/Stop Echo screen to printer	Ctrl Prt Sc (Key 55)	Any time
Delete from cursor to EOS	Ctrl PgDn	Text, command entry
Advance word	Ctrl →	Text entry
Reverse word	Ctrl ←	Text entry
Window Right	Ctrl →	When text is too wide to fit screen
Window Left	Ctrl ←	When text is too wide to fit screen
Enter insert mode	Ins	Line editor

Keyboard - Commonly Used Functions (Part 1 of 2)

2-18 Keyboard Encoding

Function	Key(s)	Comment
Exit insert mode	Ins	Line editor
Cancel current line	Esc	Command entry, text entry
Suspend system (pause)	Ctrl Num Lock	Stop list, stop program, and so on Resumes on any key
Break interrupt	Ctrl Break	Interrupt current process
System reset	Alt Ctrl Del	Reboot
Top of document and home cursor	Ctrl PgUp	Editors, word processors
Standard function keys	F1-F10	Primary function keys
Secondary function keys	Shift F1-F10 Ctrl F1-F10 Alt F1-F10	Extra function keys if 10 are not sufficient
Extra function keys	Alt Keys 2-13 (1-9,0,-,=)	Used when stickers are put along top of keyboard
Extra function keys	Alt A-Z	Used when function starts with same letter as one of the alpha keys

Keyboard - Commonly Used Functions (Part 2 of 2)

Function	Key
Carriage return	↵
Line feed	Ctrl ↵
Bell	Ctrl G
Home	Home
Cursor up	↑
Cursor down	↓
Cursor left	←
Cursor right	→
Advance one word	Ctrl →
Reverse one word	Ctrl ←
Insert	Ins
Delete	Del
Clear screen	Ctrl Home
Freeze output	Ctrl Num Lock
Tab advance	→
Stop execution (break)	Ctrl Break
Delete current line	Esc
Delete to end of line	Ctrl End
Position cursor to end of line	End

BASIC Screen Editor Special Functions

Function	Key
Suspend	Ctrl Num Lock
Echo to printer	Ctrl PrtSc
Stop echo to printer	(Key 55 any case) Ctrl PrtSc
Exit current function (break)	(Key 55 any case) Ctrl Break
Backspace	← Key 14
Line feed	Ctrl ↵
Cancel line	Esc
Copy character	F1 or →
Copy until match	F2
Copy remaining	F3
Skip character	Del
Skip until match	F4
Enter skip mode	Ins
Exit insert mode	Ins
Make new line the template	F5
String separator in REPLACE	F6
End of file in keyboard input	F6

DOS Special Functions

BIOS Cassette Logic

Software Algorithms - Interrupt Hex 15

The cassette routine will be called by the request type in AH. The address of the bytes to be read from or written to the tape will be specified by ES:BX and the number of bytes to be read or written will be specified by CX. The actual number of bytes read will be returned in DX. The read block and write block will automatically turn the cassette motor on at the start and off at the end. The request types in AH and the cassette status descriptions follow:

Request Type	Function
AH = 0	Turn Cassette Motor On
AH = 1	Turn Cassette Motor Off
AH = 2	Read Tape Block Read CX bytes into memory starting at Address ES:BX Return actual number of bytes read in DX Return Cassette Status in AH
AH = 3	Write Tape Block Write CX bytes onto cassette starting at Address DS:BX Return Cassette Status in AH

Cassette Status	Description
AH = 00	No Errors
AH = 01	Cyclic Redundancy Check (CRC) Error in Read Block
AH = 02	No Data Transitions
AH = 04	No Leader
AH = 80	Invalid Command

Note: The carry flag will be set on any error.

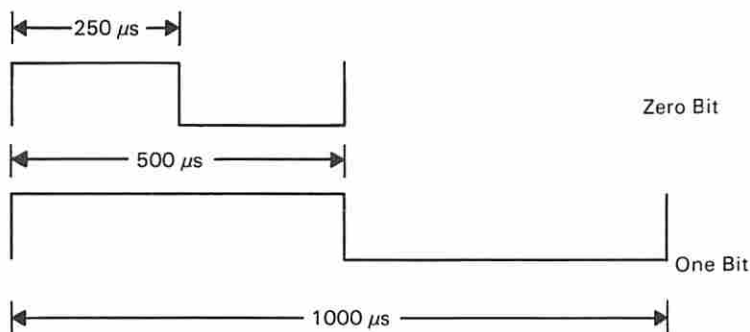
Cassette Write

The write-block routine writes a tape block onto the cassette tape. The tape block is described in "Data Record Architecture" later in this section.

The write-block routine turns on the cassette drive motor and a synchronization bit (0) and then writes the leader (256 bytes of all 1's) to the tape. Next, the routine writes the number of data blocks specified by CX. After each data block of 256 bytes, a 2-byte cyclic redundancy check (CRC) is written. The data bytes are taken from the memory location pointed at by ES.

The write-byte routine disassembles and writes the byte a bit at a time to the cassette. The method used is to set Timer 2 to the period of the desired data bit. The timer is set to a period of 1.0 millisecond for a 1 bit and 0.5 millisecond for a 0 bit.

The timer is set to mode 3, which means the timer outputs a square wave with a period given by its count register. The timer's period is changed on the fly for each data bit written to the cassette. If the number of data bytes to be written is not an integral multiple of 256, then, after the last desired data byte from memory has been written, the data block is extended to 256 bytes of writing multiples of the last data byte. The last block is closed with two CRC bytes as usual. After the last data block, a trailer consisting of four bytes of all 1 bits is written. Finally, the cassette motor is turned off, if there are no errors reported by the routine.



Cassette Read

The read-block routine turns on the cassette drive motor and then delays for approximately 0.5 second to allow the motor to come up to speed.

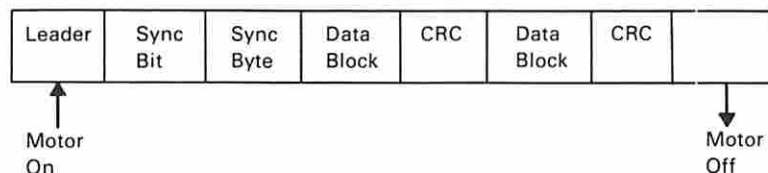
The read-block routine then searches for the leader and must detect all 1 bits for approximately 1/4 of the leader length before it can look for the sync (0) bit. After the sync bit is detected, the sync byte (ASCII character hex 16) is read. If the sync byte is read correctly, the data portion can be read. If a correct sync byte is not found, the routine goes back and searches for the leader again. The data is read a bit at a time and assembled into bytes. After each byte is assembled, it is written into memory at location ES:BX and BX is incremented by 1.

After each multiple of 256 data bytes is read, the CRC is read and compared to the CRC generated. If a CRC error is detected, the routine will exit with the carry flag set to indicate an error and the status of AH set to hex 01. DX will contain the number of bytes written memory.

The time of day interrupt (IRQ0) is disabled during the cassette-read operation.

Data Record Architecture

The write-block routine uses the following format to record a tape block onto a cassette tape:



Component	Description
Leader	256 Bytes (of All 1's)
Sync Bit	One 0 Bit
Sync Byte	ASCII Character Hex 16
Data Blocks	256 Bytes in Length
CRC	2 Bytes for each Data Block

Data Record Components

Error Recovery

Error recovery is handled through software. A CRC is used to detect errors. The polynomial used is $G(X) = X^{16} + X^{12} + X^5 + 1$, which is the polynomial used by the synchronous data link control interface. Essentially, as bits are written to or read from the cassette tape, they are passed through the CRC register in software. After a block of data is written, the complemented value of the calculated CRC register is written on the tape. Upon reading the cassette data, the CRC bytes are read and compared to the generated CRC value. If the read CRC does not equal the generated CRC, the processor's carry flag is set and the status of AH is set to hex 01, which indicates a CRC error has occurred. Also, the routine is exited on a CRC error.

APPENDIX A: ROM BIOS LISTINGS

	Page	Line Number
System ROM BIOS		
Equates	A-2	12
8088 Interrupt Locations	A-2	34
Stack	A-2	66
Data Areas	A-2	74
Power-On Self-Test	A-5	229
Boot Strap Loader	A-21	1493
I/O Support		
Asynchronous Communications		
(RS-232C)	A-22	1551
Keyboard	A-26	1818
Diskette	A-36	2426
Printer	A-46	3201
Display	A-47	3327
System Configuration Analysis		
Memory Size Determination	A-73	5177
Equipment Determination	A-73	5208
Cassette I/O Support	A-74	5253
Graphics Character Generator	A-80	5769
Time of Day	A-82	5903
Print Screen	A-84	6077
Fixed Disk ROM BIOS		
Fixed Disk I/O Interface	A-87	1
Boot Strap Loader	A-92	399


```

1      $TITLE(BIOS FOR IBM PERSONAL COMPUTER)
2
3      ;-----
4      ;   THE BIOS ROUTINES ARE MEANT TO BE ACCESSED THROUGH   :
5      ;   SOFTWARE INTERRUPTS ONLY.  ANY ADDRESSES PRESENT IN   :
6      ;   THE LISTINGS ARE INCLUDED ONLY FOR COMPLETENESS,     :
7      ;   NOT FOR REFERENCE.  APPLICATIONS WHICH REFERENCE     :
8      ;   ABSOLUTE ADDRESSES WITHIN THE CODE SEGMENT          :
9      ;   VIOLATE THE STRUCTURE AND DESIGN OF BIOS.             :
10     ;-----
11
12     ;-----
13     ;   EQUATES                                               :
14     ;-----
15     0060 PORT_A      EQU    60H          ; 8255 PORT A ADDR
16     0061 PORT_B      EQU    61H          ; 8255 PORT B ADDR
17     0062 PORT_C      EQU    62H          ; 8255 PORT C ADDR
18     0063 CMD_PORT    EQU    63H
19     0020 INTA00      EQU    20H          ; 8259 PORT
20     0021 INTA01      EQU    21H          ; 8259 PORT
21     0020 EOI         EQU    20H
22     0040 TIMER       EQU    40H
23     0063 TIM_CTL     EQU    43H          ; 8253 TIMER CONTROL PORT ADDR
24     0040 TIMERO      EQU    40H          ; 8253 TIMER/CNTR 0 PORT ADDR
25     0001 TINT        EQU    01          ; TIMER 0 INTR RECD MASK
26     0008 DMA00       EQU    08          ; DMA STATUS REG PORT ADDR
27     0000 DMA         EQU    00          ; DMA CHANNEL 0 ADDR REG PORT ADDR
28     0540 MAX_PERIOD  EQU    540H
29     0410 MIN_PERIOD  EQU    410H
30     0060 KBD_IN      EQU    60H          ; KEYBOARD DATA IN ADDR PORT
31     0002 KBDINT      EQU    02          ; KEYBOARD INTR MASK
32     0060 KB_DATA     EQU    60H          ; KEYBOARD SCAN CODE PORT
33     0061 KB_CTL      EQU    61H          ; CONTROL BITS FOR KB SENSE DATA
34     ;-----
35     ;   8088 INTERRUPT LOCATIONS                             :
36     ;-----
37     ABS0             SEGMENT AT 0
38     STG_LOCO        LABEL BYTE
39     0008             ORG    2*4
40     0008 NMI_PTR     LABEL WORD
41     0014             ORG    5*4
42     0014 INT5_PTR    LABEL WORD
43     0020             ORG    8*4
44     0020 INT_ADDR    LABEL WORD
45     0020 INT_PTR     LABEL DWORD
46     0040             ORG    10H*4
47     0040 VIDEO_INT   LABEL WORD
48     0074             ORG    1DH*4
49     0074 PARM_PTR    LABEL DWORD          ; POINTER TO VIDEO PARMS
50     0060             ORG    18H*4
51     0060 BASIC_PTR   LABEL WORD          ; ENTRY POINT FOR CASSETTE BASIC
52     0078             ORG    01EH*4       ; INTERRUPT 1EH
53     0078 DISK_POINTER LABEL DWORD
54     007C             ORG    01FH*4       ; LOCATION OF POINTER
55     007C EXT_PTR     LABEL DWORD          ; POINTER TO EXTENSION
56     0100             ORG    040H*4       ; ROUTINE
57     0100 ????        IO_ROM_INIT DW    ? ;
58     0102 ????        IO_ROM_SEG  DW    ? ; OPTIONAL ROM SEGMENT
59     0400             ORG    400H
60     0400 DATA_AREA  LABEL BYTE          ; ABSOLUTE LOCATION OF DATA SEGMENT
61     0400 DATA_WORD  LABEL WORD
62     7C00             ORG    7C00H
63     7C00 BOOT_LOCN   LABEL FAR
64     64             ABS0             ENDS
65
66     ;-----
67     ;   STACK -- USED DURING INITIALIZATION ONLY             :
68     ;-----
69     STACK            SEGMENT AT 30H
70     0000 (128        DW    128 DUP(?)
71     0100             )
72
73     0100 TOS          LABEL WORD
74     72             STACK            ENDS
75
76     ;-----
77     ;   ROM BIOS DATA AREAS                                 :
78     ;-----
79     DATA            SEGMENT AT 40H

```

LOC OBJ

LINE SOURCE

```

0000 (4      76 RS232_BASE    DW    4 DUP(?)      ; ADDRESSES OF RS232 ADAPTERS
      ????
    )
0008 (4      79 PRINTER_BASE DW    4 DUP(?)      ; ADDRESSES OF PRINTERS
      ????
    )
0010 ????    80 EQUIP_FLAG    DW    ?              ; INSTALLED HARDWARE
0012 ??      81 MFG_TST      DB    ?              ; INITIALIZATION FLAG
0013 ????    82 MEMORY_SIZE   DW    ?              ; MEMORY SIZE IN K BYTES
0015 ????    83 IO_RAM_SIZE    DW    ?              ; MEMORY IN I/O CHANNEL
      84
      ;-----
      ;      KEYBOARD DATA AREAS
      ;-----
0017 ??      87 KB_FLAG      DB    ?              ;
      88
      89 ;----- SHIFT FLAG EQUATES WITHIN KB_FLAG
      90
0080          91 INS_STATE    EQU    80H          ; INSERT STATE IS ACTIVE
0040          92 CAPS_STATE    EQU    40H          ; CAPS LOCK STATE HAS BEEN TOGGLED
0020          93 NUM_STATE     EQU    20H          ; NUM LOCK STATE HAS BEEN TOGGLED
0010          94 SCROLL_STATE  EQU    10H          ; SCROLL LOCK STATE HAS BEEN TOGGLED
0008          95 ALT_SHIFT     EQU    08H          ; ALTERNATE SHIFT KEY DEPRESSED
0004          96 CTL_SHIFT     EQU    04H          ; CONTROL SHIFT KEY DEPRESSED
0002          97 LEFT_SHIFT    EQU    02H          ; LEFT SHIFT KEY DEPRESSED
0001          98 RIGHT_SHIFT   EQU    01H          ; RIGHT SHIFT KEY DEPRESSED
      99
0018 ??      100 KB_FLAG_1    DB    ?              ; SECOND BYTE OF KEYBOARD STATUS
      101
0080          102 INS_SHIFT     EQU    80H          ; INSERT KEY IS DEPRESSED
0040          103 CAPS_SHIFT    EQU    40H          ; CAPS LOCK KEY IS DEPRESSED
0020          104 NUM_SHIFT     EQU    20H          ; NUM LOCK KEY IS DEPRESSED
0010          105 SCROLL_SHIFT  EQU    10H          ; SCROLL LOCK KEY IS DEPRESSED
0008          106 HOLD_STATE    EQU    08H          ; SUSPEND KEY HAS BEEN TOGGLED
      107
0019 ??      108 ALT_INPUT     DB    ?              ; STORAGE FOR ALTERNATE KEYPAD ENTRY
001A ????    109 BUFFER_HEAD   DW    ?              ; POINTER TO HEAD OF KEYBOARD BUFFER
001C ????    110 BUFFER_TAIL   DW    ?              ; POINTER TO TAIL OF KEYBOARD BUFFER
001E (16      111 KB_BUFFER     DW    16 DUP(?)      ; ROOM FOR 15 ENTRIES
      ????
    )
003E          112 KB_BUFFER_END LABEL WORD
      113
      114 ;----- HEAD = TAIL INDICATES THAT THE BUFFER IS EMPTY
      115
0045          116 NUM_KEY       EQU    69             ; SCAN CODE FOR NUMBER LOCK
0046          117 SCROLL_KEY    EQU    70             ; SCROLL LOCK KEY
0038          118 ALT_KEY       EQU    56             ; ALTERNATE SHIFT KEY SCAN CODE
0010          119 CTL_KEY       EQU    29             ; SCAN CODE FOR CONTROL KEY
003A          120 CAPS_KEY      EQU    58             ; SCAN CODE FOR SHIFT LOCK
002A          121 LEFT_KEY      EQU    42             ; SCAN CODE FOR LEFT SHIFT
0036          122 RIGHT_KEY     EQU    54             ; SCAN CODE FOR RIGHT SHIFT
0052          123 INS_KEY       EQU    82             ; SCAN CODE FOR INSERT KEY
0053          124 DEL_KEY       EQU    83             ; SCAN CODE FOR DELETE KEY
      125
      126 ;-----
      127 ;      DISKETTE DATA AREAS
      128 ;-----
003E ??      129 SEEK_STATUS  DB    ?              ; DRIVE RECALIBRATION STATUS
      130 ;
      131 ; BIT 3-0 = DRIVE 3-0 NEEDS RECAL BEFORE
      132 ; NEXT SEEK IF BIT IS = 0
0080          132 INT_FLAG      EQU    080H          ; INTERRUPT OCCURRENCE FLAG
003F ??      133 MOTOR_STATUS DB    ?              ; MOTOR STATUS
      134 ;
      135 ; BIT 3-0 = DRIVE 3-0 IS CURRENTLY RUNNING
      136 ; BIT 7 = CURRENT OP IS A WRITE, REQUIRES DELAY
0040 ??      136 MOTOR_COUNT  DB    ?              ; TIME OUT COUNTER FOR DRIVE TURN OFF
0025          137 MOTOR_WAIT   EQU    37             ; TWO SEC OF COUNT FOR MOTOR TURN OFF
      138
0041 ??      139 DISKETTE_STATUS DB    ?              ; BYTE OF RETURN CODE INFO FOR STATUS
0080          140 TIME_OUT      EQU    80H          ; ATTACHMENT FAILED TO RESPOND
0040          141 BAD_SEEK      EQU    40H          ; SEEK OPERATION FAILED
0020          142 BAD_NEC       EQU    20H          ; NEC CONTROLLER HAS FAILED
0010          143 BAD_CRC       EQU    10H          ; BAD CRC ON DISKETTE READ
0009          144 DMA_BOUNDARY  EQU    09H          ; ATTEMPT TO DMA ACROSS 64K BOUNDARY
0008          145 BAD_DMA       EQU    08H          ; DMA OVERRUN ON OPERATION
0004          146 RECORD_NOT_FND EQU    04H          ; REQUESTED SECTOR NOT FOUND
0003          147 WRITE_PROTECT EQU    03H          ; WRITE ATTEMPTED ON WRITE PROT DISK
0002          148 BAD_ADDR_MARK EQU    02H          ; ADDRESS MARK NOT FOUND

```

LOC OBJ	LINE	SOURCE
0001	149	BAD_CMD EQU 01H ; BAD COMMAND PASSED TO DISKETTE I/O
	150	
0042 (7	151	NEC_STATUS DB 7 DUP(?) ; STATUS BYTES FROM NEC
??		
)		
	152	
	153	-----
	154	; VIDEO DISPLAY DATA AREA ;
	155	-----
0049 ??	156	CRT_MODE DB ? ; CURRENT CRT MODE
004A ????	157	CRT_COLS DW ? ; NUMBER OF COLUMNS ON SCREEN
004C ????	158	CRT_LEN DW ? ; LENGTH OF REGEN IN BYTES
004E ????	159	CRT_START DW ? ; STARTING ADDRESS IN REGEN BUFFER
0050 (8	160	CURSOR_POSH DW 8 DUP(?) ; CURSOR FOR EACH OF UP TO 8 PAGES
????		
)		
0060 ????	161	CURSOR_MODE DW ? ; CURRENT CURSOR MODE SETTING
0062 ??	162	ACTIVE_PAGE DB ? ; CURRENT PAGE BEING DISPLAYED
0063 ????	163	ADDR_6845 DW ? ; BASE ADDRESS FOR ACTIVE DISPLAY CARD
0065 ??	164	CRT_MODE_SET DB ? ; CURRENT SETTING OF THE 3x8 REGISTER
0066 ??	165	CRT_PALETTE DB ? ; CURRENT PALETTE SETTING COLOR CARD
	166	
	167	-----
	168	; CASSETTE DATA AREA ;
	169	-----
0067 ????	170	EDGE_CNT DW ? ; TIME COUNT AT DATA EDGE
0069 ????	171	CRC_REG DW ? ; CRC REGISTER
006B ??	172	LAST_VAL DB ? ; LAST INPUT VALUE
	173	
	174	-----
	175	; TIMER DATA AREA ;
	176	-----
006C ????	177	TIMER_LOW DW ? ; LOW WORD OF TIMER COUNT
006E ????	178	TIMER_HIGH DW ? ; HIGH WORD OF TIMER COUNT
0070 ??	179	TIMER_OFL DB ? ; TIMER HAS ROLLED OVER SINCE LAST READ
	180	;COUNTS_SEC EQU 18
	181	;COUNTS_MIN EQU 1092
	182	;COUNTS_HOUR EQU 65543
	183	;COUNTS_DAY EQU 1573040 = 1800B0H
	184	
	185	-----
	186	; SYSTEM DATA AREA ;
	187	-----
0071 ??	188	BIDS_BREAK DB ? ; BIT 7 = 1 IF BREAK KEY WAS DEPRESSED
0072 ????	189	RESET_FLAG DW ? ; WORD = 1234H IF KB RESET UNDERWAY
	190	-----
	191	; FIXED DISK DATA AREA ;
	192	-----
0074 ????	193	DW ? ;
0076 ????	194	DW ? ;
	195	-----
	196	; PRINTER AND RS232 TIMEOUT CTRS ;
	197	-----
0078 (4	198	PRINT_TIM_OUT DB 4 DUP(?) ; PRINTER TIME OUT COUNTER
??		
)		
007C (4	199	RS232_TIM_OUT DB 4 DUP(?) ; RS232 TIME OUT COUNTER
??		
)		
	200	-----
	201	; EXTRA KEYBOARD DATA AREA ;
	202	-----
0080 ????	203	BUFFER_START DW ?
0082 ????	204	BUFFER_END DW ?
----	205	DATA ENDS
	206	-----
	207	; EXTRA DATA AREA ;
	208	-----
	209	xxDATA SEGMENT AT 50H
0000 ??	210	STATUS_BYTE DB ?
----	211	xxDATA ENDS
	212	
	213	-----
	214	; VIDEO DISPLAY BUFFER ;
	215	-----
----	216	VIDEO_RAM SEGMENT AT 0B800H

LOC OBJ

LINE SOURCE

```

0000          217  REGEN          LABEL  BYTE
0000          218  REGENM        LABEL  WORD
0000 16384     219  DB          16384 DUP(?)

    ??
    )
----
          220  VIDEO_RAM      ENDS
          221  ;-----
          222  ;          ROM RESIDENT CODE          :
          223  ;-----
----
          224  CODE          SEGMENT AT 0F000H
0000 57344     225  DB          57344 DUP(?)          ; FILL LOWEST 56K
    ??
    )

          226
E000 31353031343736     227  DB          '1501476 COPR. IBM 1951'          ; COPYRIGHT NOTICE
    20434F50522E20
    49424D20313938
    32

          228
          229  ;-----
          230  ;          INITIAL RELIABILITY TESTS -- PHASE 1          :
          231  ;-----
          232  ;          ASSUME          CS:CODE,SS:CODE,ES:ABS0,DS:DATA
          233  ;-----
          234  ;          DATA DEFINITIONS          :
          235  ;-----
E016 D1E0          236  C1          DW          C11          ; RETURN ADDRESS
          237
          238  ;-----
          239  ;          THIS SUBROUTINE PERFORMS A READ/WRITE STORAGE TEST ON          :
          240  ;          A 16K BLOCK OF STORAGE.          :
          241  ; ENTRY REQUIREMENTS:          :
          242  ;          ES = ADDRESS OF STORAGE SEGMENT BEING TESTED          :
          243  ;          DS = ADDRESS OF STORAGE SEGMENT BEING TESTED          :
          244  ;          WHEN ENTERING AT STGTST_CNT, CX MUST BE LOADED WITH          :
          245  ;          THE BYTE COUNT.          :
          246  ; EXIT PARAMETERS:          :
          247  ;          ZERO FLAG = 0 IF STORAGE ERROR (DATA COMPARE OR PARITY CHECK.          :
          248  ;          AL = 0 DENOTES A PARITY CHECK. ELSE AL=XOR'ED BIT          :
          249  ;          PATTERN OF THE EXPECTED DATA PATTERN VS THE          :
          250  ;          ACTUAL DATA READ.          :
          251  ;          AX,BX,CX,DX,DI, AND SI ARE ALL DESTROYED.          :
          252  ;-----
          253
          254  STGTST          PROC          NEAR
E018          255  MOV          CX,4000H          ; SETUP CNT TO TEST A 16K BLK
E018 B90040     256  STGTST_CNT:
          257  CLD
E01B FC          258  MOV          BX,CX          ; SET DIR FLAG TO INCREMENT
E01C 8BD9     259  MOV          AX,0AAAAH          ; SAVE BYTE CNT (4K FOR VIDEO OR 16K)
E01E B8AAAA     260  MOV          DX,0FF55H          ; GET DATA PATTERN TO WRITE
E021 BA55FF     261  SUB          DI,DI          ; SETUP OTHER DATA PATTERNS TO USE
E024 2BFF     262  REP          STOSB          ; DI = OFFSET 0 RELATIVE TO ES REG
E026 F3          ; WRITE STORAGE LOCATIONS
E027 AA
          263  C3:
E028          264  DEC          DI          ; STG01
E028 4F          265  STD          ; POINT TO LAST BYTE JUST WRITTEN
E029 FD          ; SET DIR FLAG TO GO BACKWARDS
E02A
          266  C4:
E02A 8BF7     267  MOV          SI,DI
E02C 8BCB     268  MOV          CX,BX          ; SETUP BYTE CNT
E02E          269  C5:          ; INNER TEST LOOP
          270  LOOVB          ; READ OLD TST BYTE FROM STORAGE [SI]+
          271  XOR          AL,AH          ; DATA READ AS EXPECTED ?
E02F 32C4     272  JNE          C7          ; NO - RETURN TO CALLER WITH AL=0
E031 7525     273  MOV          AL,DL          ; GET NEXT DATA PATTERN TO WRITE
E033 8AC2     274  STOSB          ; WRITE INTO LOCATION JUST READ [DI]+
E035 AA          275  LOOP          C5          ; DECREMENT BYTE COUNT AND LOOP CX
E036 E2F6
          276
          277  AND          AH,AH          ; ENDING ZERO PATTERN WRITTEN TO STG ?
E038 22E4     278  JZ          C6X          ; YES - RETURN TO CALLER WITH AL=0
E03A 7416     279  MOV          AH,AL          ; SETUP NEW VALUE FOR COMPARE
E03C 8AE2     280  XCHG          DH,DL          ; MOVE NEXT DATA PATTERN TO DL
E03E 86F2     281  AND          AH,AH          ; READING ZERO PATTERN THIS PASS ?
E040 22E4     282  JNZ          C6          ; CONTINUE TEST SEQUENCE TILL ZERO DATA
E042 7504     283  MOV          DL,AH          ; ELSE SET ZERO FOR END READ PATTERN
E044 8AD4     284  JMP          C3          ; AND MAKE FINAL BACKWARDS PASS
E046 EBE0
E048          285  C6:

```

LOC OBJ	LINE	SOURCE
E048 FC	286	CLD ; SET DIR FLAG TO GO FORWARD
E049 47	287	INC DI ; SET POINTER TO BEG LOCATION
E04A 7ADE	288	JZ C4 ; READ/WRITE FORWARD IN STG
E04C 4F	289	DEC DI ; ADJUST POINTER
E04D BA0100	290	MOV DX,00001H ; SETUP 01 FOR PARITY BIT
	291	; AND 00 FOR END
E050 EBD6	292	JMP C3 ; READ/WRITE BACKWARD IN STG
E052	293	C6X:
E052 E462	294	IN AL,PORT_C ; DID A PARITY ERROR OCCUR ?
E054 24C0	295	AND AL,0C0H ; ZERO FLAG WILL BE OFF PARITY ERROR
E056 B000	296	MOV AL,000H ; AL=0 DATA COMPARE OK
E058	297	C7:
E058 FC	298	CLD ; SET DEFAULT DIRCTN FLAG BACK TO INC
E059 C3	299	RET
	300	STGTST ENDP
	301	-----
	302	; 8088 PROCESSOR TEST ;
	303	; DESCRIPTION ;
	304	; VERIFY 8088 FLAGS, REGISTERS AND CONDITIONAL JUMPS ;
	305	-----
	306	ASSUME CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
E05B	307	ORG 0E05BH
E05B	308	RESET LABEL FAR
E05B	309	START:
E05B FA	310	CLI ; DISABLE INTERRUPTS
E05C B4D5	311	MOV AH,0D5H ; SET SF, CF, ZF, AND AF FLAGS ON
E05E 9E	312	SAHF
E05F 734C	313	JNC ERR01 ; GO TO ERR ROUTINE IF CF NOT SET
E061 754A	314	JNZ ERR01 ; GO TO ERR ROUTINE IF ZF NOT SET
E063 7B48	315	JNP ERR01 ; GO TO ERR ROUTINE IF PF NOT SET
E065 7946	316	JNS ERR01 ; GO TO ERR ROUTINE IF SF NOT SET
E067 9F	317	LAHF ; LOAD FLAG IMAGE TO AH
E068 B105	318	MOV CL,5 ; LOAD CNT REG WITH SHIFT CNT
E06A D2EC	319	SHR AH,CL ; SHIFT AF INTO CARRY BIT POS
E06C 733F	320	JNC ERR01 ; GO TO ERR ROUTINE IF AF NOT SET
E06E B040	321	MOV AL,40H ; SET THE OF FLAG ON
E070 D0E0	322	SHL AL,1 ; SETUP FOR TESTING
E072 7139	323	JNO ERR01 ; GO TO ERR ROUTINE IF OF NOT SET
E074 32E4	324	XOR AH,AH ; SET AH = 0
E076 9E	325	SAHF ; CLEAR SF, CF, ZF, AND PF
E077 7634	326	JBE ERR01 ; GO TO ERR ROUTINE IF CF ON
	327	; OR TO TO ERR ROUTINE IF ZF ON
E079 7832	328	JS ERR01 ; GO TO ERR ROUTINE IF SF ON
E07B 7A30	329	JP ERR01 ; GO TO ERR ROUTINE IF PF ON
E07D 9F	330	LAHF ; LOAD FLAG IMAGE TO AH
E07E B105	331	MOV CL,5 ; LOAD CNT REG WITH SHIFT CNT
E080 D2EC	332	SHR AH,CL ; SHIFT 'AF' INTO CARRY BIT POS
E082 7229	333	JC ERR01 ; GO TO ERR ROUTINE IF ON
E084 D0E4	334	SHL AH,1 ; CHECK THAT 'OF' IS CLEAR
E086 7025	335	JO ERR01 ; GO TO ERR ROUTINE IF ON
	336	
	337	;----- READ/WRITE THE 8088 GENERAL AND SEGMENTATION REGISTERS
	338	; WITH ALL ONE'S AND ZEROES'S.
	339	
E088 B0FFFF	340	MOV AX,0FFFFH ; SETUP ONE'S PATTERN IN AX
E08B F9	341	STC
E08C	342	C8:
E08C 8ED8	343	MOV DS,AX ; WRITE PATTERN TO ALL REGS
E08E 8CDB	344	MOV BX,DS
E090 8EC3	345	MOV ES,BX
E092 8CC1	346	MOV CX,ES
E094 8ED1	347	MOV SS,CX
E096 8CD2	348	MOV DX,SS
E098 8BE2	349	MOV SP,DX
E09A 8BEC	350	MOV BP,SP
E09C 8BF5	351	MOV SI,BP
E09E 8BF6	352	MOV DI,SI
E0A0 7307	353	JNC C9 ; TSTIA
E0A2 33C7	354	XOR AX,DI ; PATTERN MAKE IT THRU ALL REGS
E0A4 7507	355	JNZ ERR01 ; NO - GO TO ERR ROUTINE
E0A6 F8	356	CLC
E0A7 EBE3	357	JMP C8
E0A9	358	C9:
E0A9 0BC7	359	OR AX,DI ; TSTIA
E0AB 7401	360	JZ C10 ; ZERO PATTERN MAKE IT THRU?
E0AD F4	361	ERR01: HLT ; YES - GO TO NEXT TEST
	362	HALT SYSTEM

```

363      ;      ROS CHECKSUM TEST 1      :
364      ; DESCRIPTION      :
365      ;      A CHECKSUM IS DONE FOR THE 8K ROS MODULE      :
366      ;      CONTAINING POB AND BIOS.      :
367      ;-----:
E0AE      368      C10:
369
370      OUT      0A0H,AL      ; ZERO IN AL ALREADY
371      OUT      83H,AL      ; DISABLE NMI INTERRUPTS
372      MOV      DX,308H      ; INITIALIZE DMA PAGE REG
373      OUT      DX,AL      ;
374      INC      AL      ;
375      MOV      DL,0B8H      ; DISABLE COLOR VIDEO
376      OUT      DX,AL      ;
377      MOV      AL,99H      ; DISABLE B/W VIDEO,EN HIGH RES
378      OUT      CMD_PORT,AL      ; SET 8255 A,C-INPUT,B-OUTPUT
379      MOV      AL,0FCH      ; WRITE 8255 CMD/MODE REG
380      OUT      PORT_B,AL      ; DISABLE PARITY CHECKERS AND
381      MOV      AX,CS      ; GATE SMS SMS,CASS MOTOR OFF
382      MOV      SS,AX      ; SETUP SS SEG REG
383      MOV      DS,AX      ;
384      ; SET UP DATA SEG TO POINT TO
385      ; ROM ADDRESS
386      ASSUME    SS:CODE
387      MOV      BH,0E0H      ; SETUP STARTING ROS ADDR (E0000)
388      MOV      SP,OFFSET C1      ; SETUP RETURN ADDRESS
389      JMP      ROS_CHECKSUM
E0C9 B7E0      390      C11:
E0CB BC16E0      391      JNE      ERR01      ; HALT SYSTEM IF ERROR
E0CE E97B0B      392      ;-----:
E0D1      393      ;      8237 DMA INITIALIZATION CHANNEL REGISTER TEST      :
E0D1 750A      394      ; DESCRIPTION      :
395      ;      DISABLE THE 8237 DMA CONTROLLER. VERIFY THAT TIMER 1      :
396      ;      FUNCTIONS OK. WRITE/READ THE CURRENT ADDRESS AND WORD      :
397      ;      COUNT REGISTERS FOR ALL CHANNELS. INITIALIZE AND      :
398      ;      START DMA FOR MEMORY REFRESH.      :
399      ;-----:
E0D3 B004      399      MOV      AL,04      ; DISABLE DMA CONTROLLER
E0D5 E608      400      OUT      DMA08,AL
401
402      ;----- VERIFY THAT TIMER 1 FUNCTIONS OK
403
404      MOV      AL,54H      ; SET TIMER 1,LSB,MODE 2
405      OUT      TIMER+3,AL
406      MOV      AL,CL      ; SET INITIAL TIMER CNT TO 0
407      OUT      TIMER+1,AL
408      C12:      ; TIMER1_BITS_ON
409      MOV      AL,40H      ; LATCH TIMER 1 COUNT
410      OUT      TIMER+3,AL
411      CMP      BL,0FFH      ; YES - SEE IF ALL BITS GO OFF
412      JE      C13      ; TIMER1_BITS_OFF
413      IN      AL,TIMER+1      ; READ TIMER 1 COUNT
414      OR      BL,AL      ; ALL BITS ON IN TIMER
415      LOOP     C12      ; TIMER1_BITS_ON
416      HLT      ; TIMER 1 FAILURE, HALT SYS
417      C13:      ; TIMER1_BITS_OFF
418      MOV      AL,BL      ; SET TIMER 1 CNT
419      SUB      CX,CX
420      OUT      TIMER+1,AL
421      C14:      ; TIMER_LOOP
422      MOV      AL,40H      ; LATCH TIMER 1 COUNT
423      OUT      TIMER+3,AL
424      NOP      ; DELAY FOR TIMER
425      NOP
426      IN      AL,TIMER+1      ; READ TIMER 1 COUNT
427      AND      BL,AL
428      JZ      C15      ; GO TO WRAP_DMA_REG
429      LOOP     C14      ; TIMER_LOOP
430      HLT      ; TIMER ERROR - HALT SYSTEM
431
432      ;----- INITIALIZE TIMER 1 TO REFRESH MEMORY
433
434      C15:      ; WRAP_DMA_REG
435      MOV      AL,18      ; SETUP DIVISOR.FOR REFRESH
436      OUT      TIMER+1,AL      ; WRITE TIMER 1 CNT REG
437      OUT      DMA+0DH,AL      ; SEND MASTER CLEAR TO DMA
438

```

LOC	OBJ	LINE	SOURCE
		439	;----- WRAP DMA CHANNELS ADDRESS AND COUNT REGISTERS
		440	
E10A B0FF		441	MOV AL,0FFH ; WRITE PATTERN FF TO ALL REGS
E10C		442	C16:
E10C 8AD8		443	MOV BL,AL ; SAVE PATTERN FOR COMPARE
E10E 8AF8		444	MOV BH,AL
E110 B90800		445	MOV CX,8 ; SETUP LOOP CNT
E113 2B02		446	SUB DX,DX ; SETUP I/O PORT ADDR OF REG (0000)
E115		447	C17:
E115 EE		448	OUT DX,AL ; WRITE PATTERN TO REG, LSB
E116 50		449	PUSH AX
E117 EE		450	OUT DX,AL ; MSB OF 16 BIT REG
E118 B80101		451	MOV AX,0101H ; AX TO ANOTHER PAT BEFORE RD
E11B EC		452	IN AL,DX ; READ 16-BIT DMA CH REG, LSB
E11C 8AE0		453	MOV AH,AL ; SAVE LSB OF 16-BIT REG
E11E EC		454	IN AL,DX ; READ MSB OF DMA CH REG
E11F 3B08		455	CMP BX,AX ; PATTERN READ AS WRITTEN?
E121 7401		456	JE C18 ; YES - CHECK NEXT REG
E123 F4		457	HLT ; NO - HALT THE SYSTEM
E124		458	C18:
E124 42		459	INC DX ; NEXT_DMA_CH
E125 E2EE		460	LOOP C17 ; SET I/O PORT TO NEXT CH REG
E127 FEC0		461	INC AL ; WRITE PATTERN TO NEXT REG
E129 74E1		462	JZ C16 ; SET PATTERN TO 0
		463	WRITE TO CHANNEL REGS
		464	;----- INITIALIZE AND START DMA FOR MEMORY REFRESH.
		465	
E12B 0EDB		466	MOV DS,BX ; SET UP ABS0 INTO DS AND ES
E12D 0EC3		467	MOV ES,BX
		468	ASSUME DS:ABS0,ES:ABS0
		469	
E12F B0FF		470	MOV AL,0FFH ; SET CNT OF 64K FOR RAM REFRESH
E131 E601		471	OUT DMA+1,AL
E133 50		472	PUSH AX
E134 E601		473	OUT DMA+1,AL
E136 B20B		474	MOV DL,0BH ; DX=000B
E138 B058		475	MOV AL,058H ; SET DMA MODE,CH 0,READ,AUTOINT
E13A EE		476	OUT DX,AL ; WRITE DMA MODE REG
E13B B000		477	MOV AL,0 ; ENABLE DMA CONTROLLER
E13D E608		478	OUT DMA+8,AL ; SETUP DMA COMMAND REG
E13F 50		479	PUSH AX
E140 E60A		480	OUT DMA+10,AL ; ENABLE CHANNEL 0 FOR REFRESH
E142 B103		481	MOV CL,3
E144 B041		482	MOV AL,41H ; SET MODE FOR CHANNEL 1
E146		483	C18A:
E146 EE		484	OUT DX,AL
E147 FEC0		485	INC AL ; POINT TO NEXT CHANNEL
E149 E2FB		486	LOOP C18A
		487	;-----
		488	; BASE 16K READ/WRITE STORAGE TEST :
		489	; DESCRIPTION :
		490	; WRITE/READ/VERIFY DATA PATTERNS FF,55,AA,01, AND 00 :
		491	; TO 1ST 16K OF STORAGE. VERIFY STORAGE ADDRESSABILITY. :
		492	; INITIALIZE THE 8259 INTERRUPT CONTROLLER CHIP FOR :
		493	; CHECKING MANUFACTURING TEST 2 MODE. :
		494	;-----
		495	
		496	;----- DETERMINE MEMORY SIZE AND FILL MEMORY WITH DATA
		497	
E14B B1302		498	MOV DX,0213H ; ENABLE EXPANSION BOX
E14E B001		499	MOV AL,01H
E150 EE		500	OUT DX,AL
E151 8B2E7204		501	MOV BP,DATA_WORD[OFFSET RESET_FLAG] ; SAVE 'RESET_FLAG' IN BP
E155 81FD3412		502	CMP BP,1234H ; WARN START?
E159 740A		503	JE C18B ; BYPASS STG TST.
E15B BC41F090		504	MOV SP,OFFSET C2
E15F E9B6FE		505	JMP STGTST
E162		506	C24:
E162 7401		507	JE C18B ; PROCEED IF STGTST OK
E164 F4		508	HLT ; HALT IF NOT
E165		509	C18B:
E165 2BFF		510	SUB DI,DI
E167 E460		511	IN AL,PORT_A ; DETERMINE BASE RAM SIZE
E169 240C		512	AND AL,0CH ; ISOLATE RAM SIZE SMS
E16B 0404		513	ADD AL, 4 ; CALCULATE MEMORY SIZE
E16D B10C		514	MOV CL, 12

LOC OBJ	LINE	SOURCE
E16F D3E0	515	SHL AX, CL
E171 0BC8	516	MOV CX, AX
E173 FC	517	CLD ; SET DIR FLAG TO INCR
E174	518	
E174 AA	519	STOSB ; FILL BASE RAM WITH DATA
E175 E2FD	520	LOOP C19 ; LOOP TIL ALL ZERO
E177 892E7204	521	MOV DATA_WORD[OFFSET RESET_FLAG],BP
	522	
	523	----- DETERMINE IO CHANNEL RAM SIZE
	524	
E17B B0F8	525	MOV AL,0F8H ; ENABLE SWITCH 5
E17D E661	526	OUT PORT_B,AL
E17F E462	527	IN AL,PORT_C ; READ SWITCHES
E181 2401	528	AND AL,00000001B ; ISOLATE SWITCH 5
E183 B10C	529	MOV CL,12D
E185 D3C0	530	ROL AX,CL
E187 B0FC	531	MOV AL,0FCH ; DISABLE SW. 5
E189 E661	532	OUT PORT_B,AL
E18B E462	533	IN AL,PORT_C
E18D 240F	534	AND AL,0FH
E18F 0AC4	535	OR AL,AH ; COMBINE SWITCH VALUES
E191 8AD8	536	MOV BL,AL ; SAVE
E193 B420	537	MOV AH,32
E195 F6E4	538	MUL AH ; CALC. LENGTH
E197 A31504	539	MOV DATA_WORD[OFFSET IO_RAM_SIZE],AX ; SAVE IT
E19A 7418	540	JZ C21
E19C BA0010	541	MOV DX,1000H ; SEGMENT FOR I/O RAM
E19F 8AE0	542	MOV AH,AL
E1A1 B000	543	MOV AL,0
E1A3	544	C20: ; FILL_IO:
E1A3 8EC2	545	MOV ES,DX
E1A5 B90080	546	MOV CX,8000H ; FILL 32K BYTES
E1A8 2BFF	547	SUB DI,DI
E1AA F3	548	REP STOSB
E1AB AA		
E1AC 81C20008	549	ADD DX,800H ; NEXT SEGMENT VALUE
E1B0 FECB	550	DEC BL
E1B2 75EF	551	JNZ C20 ; FILL_IO
	552	-----
	553	INITIALIZE THE 8259 INTERRUPT CONTROLLER CHIP ;
	554	-----
E1B4	555	C21:
E1B4 B013	556	MOV AL,13H ; ICN1 - EDGE, SNGL, ICN4
E1B6 E620	557	OUT INTA00,AL
E1B8 B008	558	MOV AL,8 ; SETUP ICN2 - INT TYPE 8 (8-F)
E1BA E621	559	OUT INTA01,AL
E1BC B009	560	MOV AL,9 ; SETUP ICN4 - BUFFRD,8086 MODE
E1BE E621	561	OUT INTA01,AL
E1C0 2BC0	562	SUB AX,AX ; POINT ES TO BEGIN
E1C2 8EC0	563	MOV ES,AX ; OF R/W STORAGE
	564	
	565	-----
	566	CHECK FOR MANUFACTURING TEST 2 TO LOAD TEST PROGRAMS FROM KEYBOARD. ;
	567	-----
	568	----- SETUP STACK SEG AND SP
	569	
E1C4 B83000	570	MOV AX,STACK ; GET STACK VALUE
E1C7 8ED0	571	MOV SS,AX ; SET THE STACK UP
E1C9 BC0001	572	MOV SP,OFFSET TOS ; STACK IS READY TO GO
E1CC 81FD3412	573	CHP BP,1234H ; RESET_FLAG SET?
E1D0 7425	574	JE C25 ; YES - SKIP MFG TEST
E1D2 2BFF	575	SUB DI,DI
E1D4 8EDF	576	MOV DS, DI
E1D6 B82400	577	MOV BX, 24H
E1D9 C70747FF	578	MOV WORD PTR [BX],OFFSET D11 ; SET UP KB INTERRUPT
E1DD 43	579	INC BX
E1DE 43	580	INC BX
E1DF 8C0F	581	MOV [BX],CS
E1E1 E85F04	582	CALL KBD_RESET ; READ IN KB RESET CODE TO BL
E1E4 80FB65	583	CHP BL,065H ; IS THIS MANUFACTURING TEST 2?
E1E7 750E	584	JNZ C25 ; JUMP IF NOT MAN. TEST
E1E9 B2FF	585	MOV DL,255 ; READ IN TEST PROGRAM
E1EB	586	
E1EB E86204	587	C22: CALL SP_TEST
E1EE 8AC3	588	MOV AL,BL
E1F0 AA	589	STOSB

LOC	OBJ	LINE	SOURCE
E1F1	FECA	590	DEC DL
E1F3	75F6	591	JNZ C22 ; JUMP IF NOT DONE YET
E1F5	CD3E	592	INT 3EH ; SET INTERRUPT TYPE 62 ADDRESS F0H
E1F7		593	C25:
		594	
		595	;----- SET UP THE BIOS INTERRUPT VECTORS TO TEMP INTERRUPT
		596	
E1F7	B92000	597	MOV CX,32 ; FILL ALL 32 INTERRUPTS
E1FA	2BFF	598	SUB DI,DI ; FIRST INTERRUPT LOCATOIN
E1FC		599	D3:
E1FC	B047FF	600	MOV AX,OFFSET D11 ; MOVE ADDR OF INTR PROC TO TBL
E1FF	AB	601	STOSW
E200	8CC8	602	MOV AX,CS ; GET ADDR OF INTR PROC SEG
E202	AB	603	STOSW
E203	E2F7	604	LOOP D3 ; VECTBLO
		605	
		606	;----- SET UP OTHER INTERRUPTS AS NECESSARY
		607	
E205	C7060800C3E2	608	MOV NMI_PTR,OFFSET NMI_INT ; NMI INTERRUPT
E20B	C706140054FF	609	MOV INT5_PTR,OFFSET PRINT_SCREEN ; PRINT SCREEN
E211	C706620000F6	610	MOV BASIC_PTR+2,0F600H ; SEGMENT FOR CASSETTE BASIC
		611	
		612	;-----
		613	; 8259 INTERRUPT CONTROLLER TEST ;
		614	; DESCRIPTION ;
		615	; READ/WRITE THE INTERRUPT MASK REGISTER (IMR) WITH ALL ;
		616	; ONES AND ZEROES. ENABLE SYSTEM INTERRUPTS. MASK DEVICE ;
		617	; INTERRUPTS OFF. CHECK FOR HOT INTERRUPTS (UNEXPECTED). ;
		618	;-----
		619	
		620	;----- TEST THE IMR REGISTER
		621	
E217	BA2100	622	MOV DX,0021H ; POINT INTR. CHIP ADDR 21
E21A	B000	623	MOV AL,0 ; SET IMR TO ZERO
E21C	EE	624	OUT DX,AL
E21D	EC	625	IN AL,DX ; READ IMR
E21E	0AC0	626	OR AL,AL ; IMR = 0?
E220	7515	627	JNZ D6 ; GO TO ERR ROUTINE IF NOT 0
E222	B0FF	628	MOV AL,OFFFH ; DISABLE DEVICE INTERRUPTS
E224	EE	629	OUT DX,AL ; WRITE TO IMR
E225	EC	630	IN AL,DX ; READ IMR
E226	0401	631	ADD AL,1 ; ALL IMR BIT ON?
E228	750D	632	JNZ D6 ; NO - GO TO ERR ROUTINE
		633	
		634	;----- CHECK FOR HOT INTERRUPTS
		635	
		636	;----- INTERRUPTS ARE MASKED OFF. CHECK THAT NO INTERRUPTS OCCUR.
		637	
E22A	32E4	638	XOR AH,AH ; CLEAR AH REG
E22C	FB	639	STI ; ENABLE EXTERNAL INTERRUPTS
E22D	2BC9	640	SUB CX,CX ; WAIT 1 SEC FOR ANY INTRs THAT
E22F		641	D4:
E22F	E2FE	642	LOOP D4 ; MIGHT OCCUR
E231		643	D5:
E231	E2FE	644	LOOP D5
E233	0AE4	645	OR AH,AH ; DID ANY INTERRUPTS OCCUR?
E235	7408	646	JZ D7 ; NO - GO TO NEXT TEST
E237		647	D6:
E237	BA0101	648	MOV DX,101H ; BEEP SPEAKER IF ERROR
E23A	E09203	649	CALL ERR_BEEP ; GO TO BEEP SUBROUTINE
E23D	FA	650	CLI
E23E	F4	651	HLT ; HALT THE SYSTEM
		652	;-----
		653	; 8253 TIMER CHECKOUT ;
		654	; DESCRIPTION ;
		655	; VERIFY THAT THE SYSTEM TIMER (0) ;
		656	; DOESN'T COUNT TOO FAST OR TOO SLOW. ;
		657	;-----
E23F		658	D7:
E23F	B0FE	659	MOV AL,0FEH ; MASK ALL INTRs EXCEPT LVL 0
E241	EE	660	OUT DX,AL ; WRITE THE 8259 IMR
E242	B010	661	MOV AL,00010000B ; SET TIM 0, LSB, MODE 0, BINARY
E244	E643	662	OUT TIM_CTL,AL ; WRITE TIMER CONTROL MODE REG
E246	B91600	663	MOV CX,16H ; SET PGM LOOP CNT
E249	BAC1	664	MOV AL,CL ; SET TIMER 0 CNT REG
E24B	E640	665	OUT TIMER0,AL ; WRITE TIMER 0 CNT REG

LOC OBJ	LINE	SOURCE
E24D	666	D0:
E24D F6C4FF	667	TEST AH,OFFH ; DID TIMER 0 INTERRUPT OCCUR?
E250 7504	668	JNZ D9 ; YES - CHECK TIMER 0P FOR SLOW TIME
E252 E2F9	669	LOOP D8 ; WAIT FOR INTR FOR SPECIFIED TIME
E254 EBE1	670	JMP D6 ; TIMER 0 INTR DIDN'T OCCUR - ERR
E256	671	D9:
E256 B112	672	MOV CL,18 ; SET PGH LOOP CNT
E258 B0FF	673	MOV AL,OFFH ; WRITE TIMER 0 CNT REG
E25A E640	674	OUT TIMER0,AL
E25C B8FE00	675	MOV AX,0FEH
E25F EE	676	OUT DX,AL
E260	677	D10:
E260 F6C4FF	678	TEST AH,OFFH ; DID TIMER 0 INTERRUPT OCCUR?
E263 75D2	679	JNZ D6 ; YES - TIMER CNTING TOO FAST, ERR
E265 E2F9	680	LOOP D10 ; WAIT FOR INTR FOR SPECIFIED TIME
	681	
	682	;----- ESTABLISH BIOS SUBROUTINE CALL INTERRUPT VECTORS
	683	
E267 1E	684	PUSH DS ; SAVE POINTER TO DATA AREA
E268 BF4000	685	MOV DI,OFFSET VIDEO_INT ; SETUP ADDR TO INTR AREA
E26B 0E	686	PUSH CS
E26C 1F	687	POP DS ; SETUP ADDR OF VECTOR TABLE
E26D BE03FF90	688	MOV SI,OFFSET VECTOR_TABLE*16 ; START WITH VIDEO ENTRY
E271 B91000	689	MOV CX,16
	690	
	691	;----- SETUP TIMER 0 TO MODE 3
	692	
E274 B0FF	693	MOV AL,OFFH ; DISABLE ALL DEVICE INTERRUPTS
E276 EE	694	OUT DX,AL
E277 B036	695	MOV AL,36H ; SEL TIM 0,LSB,MSB,MODE 3
E279 E643	696	OUT TIMER+3,AL ; WRITE TIMER MODE REG
E27B B000	697	MOV AL,0
E27D E640	698	OUT TIMER,AL ; WRITE LSB TO TIMER 0 REG
E27F	699	E1A:
E27F A5	700	MOVSW ; MOVE VECTOR TABLE TO RAM
E280 47	701	INC DI ; MOVE PAST SEGMENT POINTER
E281 47	702	INC DI
E282 E2FB	703	LOOP E1A
E284 E640	704	OUT TIMER,AL ; WRITE MSB TO TIMER 0 REG
E286 1F	705	POP DS ; RECOVER DATA SEG POINTER
	706	
	707	;----- SETUP TIMER 0 TO BLINK LED IF MANUFACTURING TEST MODE
	708	
E287 E8B903	709	CALL KBD_RESET ; SEND SOFTWARE RESET TO KEYBRD
E28A 80FBAA	710	CHP BL,0AAH ; SCAN CODE 'AA' RETURNED?
E28D 741E	711	JE E6 ; YES - CONTINUE (NON MFG MODE)
E28F B03C	712	MOV AL,3CH ; EN KBD, SET KBD CLK LINE LOW
E291 E661	713	OUT PORT_B,AL ; WRITE 8255 PORT B
E293 90	714	NOP
E294 90	715	NOP
E295 E460	716	IN AL,PORT_A ; WAS A BIT CLOCKED IN?
E297 24FF	717	AND AL,OFFH
E299 750E	718	JNZ E2 ; YES - CONTINUE (NON MFG MODE)
E29B FE061204	719	INC DATA_AREA[OFFSET MFG_TST] ; ELSE SET SW FOR MFG TEST MODE
E29F C70620006DE6	720	MOV INT_ADDR,OFFSET BLINK_INT ; SETUP TIMER INTR TO BLINK LED
E2A5 B0FE	721	MOV AL,0FEH ; ENABLE TIMER INTERRUPT
E2A7 E621	722	OUT INTA01,AL
E2A9	723	E2: ; JUMPER_NOT_IN:
E2A9 B0CC	724	MOV AL,0CCH ; RESET THE KEYBOARD
E2AB E661	725	OUT PORT_B,AL
	726	
	727	;-----
	728	; INITIALIZE AND START CRT CONTROLLER (6845) ;
	729	; TEST VIDEO READ/WRITE STORAGE. ;
	730	; DESCRIPTION ;
	731	; RESET THE VIDEO ENABLE SIGNAL. ;
	732	; SELECT ALPHANUMERIC MODE, 40 = 25, B & M. ;
	733	; READ/WRITE DATA PATTERNS TO STG. CHECK STG. ;
	734	; ADDRESSABILITY. ;
	735	;-----
E2AD	736	E6:
E2AD E640	737	IN AL,PORT_A ; READ SENSE SWITCHES
E2AF B400	738	MOV AH,0
E2B1 A31004	739	MOV DATA_WORD[OFFSET EQUIP_FLAG],AX ; STORE SENSE SW INFO
E2B4	740	E6A:
E2B4 2430	741	AND AL,30H ; ISOLATE VIDEO SMS
E2B6 7529	742	JNZ E7 ; VIDEO SMS SET TO 0?

LOC OBJ	LINE	SOURCE
E2B8 C706400053FF	743	MOV VIDEO_INT,OFFSET DUMMY_RETURN
E2BE E9A200	744	JMP E18_1 ; SKIP VIDEO TESTS FOR BURN-IN
	745	
E2C3	746	ORG 0E2C3H
E2C3	747	NMI_INT PROC NEAR
E2C3 50	748	PUSH AX ; SAVE ORIG CONTENTS OF AX
E2C4 E462	749	IN AL,PORT_C
E2C6 A8C0	750	TEST AL,0C0H ; PARITY CHECK?
E2C8 7415	751	JZ D14 ; NO, EXIT FROM ROUTINE
E2CA BEDAFF90	752	MOV SI,OFFSET D1 ; ADDR OF ERROR MSG
E2CE AB40	753	TEST AL,40H ; I/O PARITY CHECK
E2D0 7504	754	JNZ D13 ; DISPLAY ERROR MSG
E2D2 BE23FF90	755	MOV SI,OFFSET D2 ; MUST BE PLANAR
E2D6	756	D13: ;
E2D6 2BC0	757	SUB AX,AX ; INIT AND SET MODE FOR VIDEO
E2D8 CD10	758	INT 10H ; CALL VIDEO_IO PROCEDURE
E2DA E8DD03	759	CALL P_MSG ; PRINT ERROR MSG
E2DD FA	760	CLI
E2DE F4	761	HLT ; HALT SYSTEM
E2DF	762	D14: ;
E2DF 58	763	POP AX ; RESTORE ORIG CONTENTS OF AX
E2E0 CF	764	IRET
	765	NMI_INT ENDP
E2E1	766	E7: ; TEST_VIDEO:
E2E1 3C30	767	CMP AL,30H ; B/W CARD ATTACHED?
E2E3 7408	768	JE E8 ; YES - SET MODE FOR B/W CARD
E2E5 FEC4	769	INC AH ; SET COLOR MODE FOR COLOR CD
E2E7 3C20	770	CMP AL,20H ; 80X25 MODE SELECTED?
E2E9 7502	771	JNE E8 ; NO - SET MODE FOR 40X25
E2EB B403	772	MOV AH,3 ; SET MODE FOR 80X25
E2ED	773	E8: ;
E2ED 86E0	774	XCHG AH,AL ; SET_MODE
E2EF 50	775	PUSH AX ; SAVE VIDEO MODE ON STACK
E2F0 2AE4	776	SUB AH,AH ; INITIALIZE TO ALPHANUMERIC MD
E2F2 CD10	777	INT 10H ; CALL VIDEO_IO
E2F4 58	778	POP AX ; RESTORE VIDEO SENSE SMS IN AH
E2F5 50	779	PUSH AX ; RESAVE VALUE
E2F6 BB00B0	780	MOV BX,0B000H ; BEG VIDEO RAM ADDR B/W CD
E2F9 BAB003	781	MOV DX,3B8H ; MODE REG FOR B/W
E2FC B90010	782	MOV CX,4096 ; RAM BYTE CNT FOR B/W CD
E2FF B001	783	MOV AL,1 ; SET MODE FOR BW CARD
E301 80FC30	784	CMPL AH,30H ; B/W VIDEO CARD ATTACHED?
E304 7408	785	JE E9 ; YES - GO TEST VIDEO STG
E306 B7B8	786	MOV BH,0B8H ; BEG VIDEO RAM ADDR COLOR CD
E308 B2D8	787	MOV DL,0D8H ; MODE REG FOR COLOR CD
E30A B540	788	MOV CH,40H ; RAM BYTE CNT FOR COLOR CD
E30C FEC8	789	DEC AL ; SET MODE TO 0 FOR COLOR CD
E30E	790	E9: ; TEST_VIDEO_STG:
E30E EE	791	OUT DX,AL ; DISABLE VIDEO FOR COLOR CD
E30F 81FD3412	792	CMPL BP,1234H ; P00 INITIATED BY KBD RESET?
E313 8EC3	793	MOV ES,BX ; POINT ES TO VIDEO RAM STG
E315 7407	794	JE E10 ; YES - SKIP VIDEO RAM TEST
E317 8EDB	795	MOV DS,BX ; POINT DS TO VIDEO RAM STG
	796	ASSUME DS:NOTHING,ES:NOTHING
E319 E8FFFC	797	CALL STGTST_CNT ; GO TEST VIDEO R/W STG
E31C 7532	798	JNE E17 ; R/W STG FAILURE - BEEP SPK
	799	;
	800	;-----
	801	; SETUP VIDEO DATA ON SCREEN FOR VIDEO LINE TEST. ;
	802	; DESCRIPTION ;
	803	; ENABLE VIDEO SIGNAL AND SET MODE. ;
	804	; DISPLAY A HORIZONTAL BAR ON SCREEN. ;
	805	;-----
E31E	806	E10: ;
E31E 58	807	POP AX ; GET VIDEO SENSE SMS (AH)
E31F 50	808	PUSH AX ; SAVE IT
E320 B400	809	MOV AH,0 ; ENABLE VIDEO AND SET MODE
E322 CD10	810	INT 10H ; VIDEO
E324 B82070	811	MOV AX,7020H ; WRT BLANKS IN REVERSE VIDEO
E327 2BFF	812	SUB DI,DI ; SETUP STARTING LOC
E329 B92800	813	MOV CX,40 ; NO. OF BLANKS TO DISPLAY
E32C F3		REP STOSW ; WRITE VIDEO STORAGE
E32D AB		
	814	;-----
	815	; CRT INTERFACE LINES TEST ;
	816	; DESCRIPTION ;
	817	; SENSE ON/OFF TRANSITION OF THE VIDEO ENABLE ;

```

818      AND HORIZONTAL SYNC LINES.
819      ;-----
820      POP      AX          ; GET VIDEO SENSE SW INFO
821      PUSH     AX          ; SAVE IT
822      CMP      AH,30H      ; B/M CARD ATTACHED?
823      MOV      DX,03BAH    ; SETUP ADDR OF BM STATUS PORT
824      JE       E11         ; YES - GO TEST LINES
825      MOV      DL,0DAH     ; COLOR CARD IS ATTACHED
826      E11:          ; LINE_TST:
827      MOV      AH,8
828      E12:          ; OFLOOP_CNT:
829      SUB      CX,CX
830      E13:
831      IN       AL,DX       ; READ CRT STATUS PORT
832      AND      AL,AH       ; CHECK VIDEO/HORZ LINE
833      JNZ      E14         ; ITS ON - CHECK IF IT GOES OFF
834      LOOP     E13         ; LOOP TILL ON OR TIMEOUT
835      JMP      SHORT E17   ; GO PRINT ERROR MSG
836
837      SUB      CX,CX
838      E15:
839      IN       AL,DX       ; READ CRT STATUS PORT
840      AND      AL,AH       ; CHECK VIDEO/HORZ LINE
841      JZ       E16         ; ITS ON - CHECK NEXT LINE
842      LOOP     E15         ; LOOP IF OFF TILL IT GOES ON
843      E17:          ; CRT_ERR
844      MOV      DX,102H     ; GO BEEP SPEAKER
845      CALL     ERR_BEEP
846      JMP      SHORT E18
847
848      MOV      CL,3
849      SHR      AH,CL
850      JNZ      E12         ; GO CHECK HORIZONTAL LINE
851      E18:          ; DISPLAY_CURSOR:
852      POP      AX          ; GET VIDEO SENSE SMS (AH)
853      MOV      AH,0
854      INT      10H        ; SET MODE AND DISPLAY CURSOR
855                          ; CALL VIDEO I/O PROCEDURE
856
857      E18_1:
858      MOV      DX,0C000H
859
860      E18A:
861      MOV      DS,DX
862      SUB      BX,BX
863      MOV      AX,[BX]
864      PUSH     BX          ; GET FIRST 2 LOCATIONS
865      POP      BX
866      POP      BX          ; LET BUS SETTLE
867      CMP      AX,0AA55H   ; PRESENT?
868      JNZ      E18B       ; NO? GO LOOK FOR OTHER MODULES
869      CALL     ROM_CHECK   ; GO SCAN MODULE
870      JMP      SHORT E18C
871
872      E18B:
873      ADD      DX,0080H    ; POINT TO NEXT 2K BLOCK
874
875      E18C:
876      CMP      DX,0C800H   ; TOP OF VIDEO ROM AREA YET?
877      JL       E18A       ; GO SCAN FOR ANOTHER MODULE
878
879      ;-----
880      ; EXPANSION I/O BOX TEST
881      ;
882      ; CHECK TO SEE IF EXPANSION BOX PRESENT - IF INSTALLED,
883      ; TEST DATA AND ADDRESS BUSES TO I/O BOX.
884      ;
885      ; ERROR='1801'
886      ;-----
887
888      ;----- DETERMINE IF BOX IS PRESENT
889
890      EXP_IO:
891      MOV      DX,0210H    ; (CARD HAS ENABLED EARLIER)
892      MOV      AX,5555H    ; CONTROL PORT ADDRESS
893      OUT      DX,AL       ; SET DATA PATTERN
894      MOV      AL,01H
895      IN       AL,DX       ; RECOVER DATA
896      CMP      AL,AH       ; REPLY?
897      JNE      E19         ; NO RESPONSE, GO TO NEXT TEST
898      NOT      AX          ; MAKE DATA=AAAA
899      OUT      DX,AL
900      MOV      AL,01H
901      IN       AL,DX
902      IN       AL,DX       ; RECOVER DATA
903      CMP      AL,AH

```

LOC OBJ	LINE	SOURCE
E398 752A	895	JNE E19 ; NO ANSWER=NEXT TEST
	896	
	897	;----- CHECK ADDRESS AND DATA BUS
	898	
E39A	899	EXP1:
E39A 8BD8	900	MOV BX,AX
E39C BA1402	901	MOV DX,0214H ; LOAD DATA REG ADDRESS
E39F 2E8807	902	MOV CS:[BX],AL ; WRITE ADDRESS F0000+BX
E3A2 EE	903	OUT DX,AL ; WRITE DATA
E3A3 90	904	NOP
E3A4 EC	905	IN AL,DX ; READ DATA
E3A5 3AC7	906	CMP AL,BH
E3A7 7514	907	JNE EXP_ERR
E3A9 42	908	INC DX ; DX=215H (ADDR. HI REG)
E3AA EC	909	IN AL,DX
E3AB 3AC4	910	CMP AL,AH ; COMPARE TO HI ADDRESS
E3AD 750E	911	JNE EXP_ERR
E3AF 42	912	INC DX ; DX=216H (ADDR. LOW REG)
E3B0 EC	913	IN AL,DX
E3B1 3AC4	914	CMP AL,AH ; ADDR. LOW OK?
E3B3 7508	915	JNE EXP_ERR
E3B5 F7D0	916	NOT AX ; INVERT AX
E3B7 3CAA	917	CMP AL,0AAH ; BACK TO STARTING VALUE (AAAA) YET
E3B9 7409	918	JE E19 ; GO ON TO NEXT TEST IF SO
E3BB E8DD	919	JMP EXP1 ; LOOP BACK THROUGH WITH DATA OF 5555
E3BD	920	EXP_ERR:
E3BD BEEDFE90	921	MOV SI,OFFSET F3B
E3C1 E0F602	922	CALL P_MSG
	923	;-----
	924	; ADDITIONAL READ/WRITE STORAGE TEST ;
	925	; DESCRIPTION ;
	926	; WRITE/READ DATA PATTERNS TO ANY READ/WRITE STORAGE ;
	927	; AFTER THE BASIC 16K. STORAGE ADDRESSABILITY IS CHECKED. ;
	928	;-----
	929	ASSUME DS:DATA
E3C4	930	E19:
	931	
	932	;----- DETERMINE RAM SIZE ON PLANAR BOARD
	933	
E3C4 E8771B	934	CALL DDS
E3C7 A01000	935	MOV AL,BYTE PTR EQUIP_FLAG ; GET SENSE SMS INFO
E3CA 240C	936	AND AL,0CH ; ISOLATE RAM SIZE SMS
E3CC B404	937	MOV AH,4
E3CE F6E4	938	MUL AH
E3D0 0410	939	ADD AL,16 ; ADD BASIC 16K
E3D2 88D0	940	MOV DX,AX ; SAVE PLANAR RAM SIZE IN DX
E3D4 88D8	941	MOV BX,AX ; AND IN BX
	942	
	943	;----- DETERMINE IO CHANNEL RAM SIZE
	944	
E3D6 A11500	945	MOV AX,IO_RAM_SIZE ; GET IO CHANNEL RAM SIZE
E3D9 83FB40	946	CMP BX,40H ; PLANAR RAM SIZE = 64K?
E3DC 7402	947	JE E20 ; YES - ADD IO CHN RAM SIZE
E3DE 2BC0	948	SUB AX,AX ; NO - DON'T ADD ANY IO RAM
E3E0	949	E20: ; ADD_IO_SIZE:
E3E0 03C3	950	ADD AX,BX ; SUM TOTAL RAM SIZE
E3E2 A31300	951	MOV MEMORY_SIZE,AX ; SETUP MEMORY SIZE PARAM
E3E5 81FD3412	952	CMP BP,1234H ; POD INITIATED BY KBD RESET?
E3E9 1E	953	PUSH DS ; SAVE DATA SEGMENT
E3EA 744F	954	JE TST12 ; YES - SKIP MEMORY TEST
	955	
	956	;----- TEST ANY OTHER READ/WRITE STORAGE AVAILABLE
	957	
E3EC BB0004	958	MOV BX,400H
E3EF B91000	959	MOV CX,16
E3F2	960	E21:
E3F2 38D1	961	CMP DX,CX ; ANY MORE STG TO BE TESTED?
E3F4 76D0	962	JBE E23 ; NO - GO TO NEXT TEST
E3F6 8EDB	963	MOV DS,BX ; SETUP STG ADDR IN DS AND ES
E3F8 8EC3	964	MOV ES,BX
E3FA 83C110	965	ADD CX,16 ; INCREMENT STG BYTE COUNTER
E3FD 81C30004	966	ADD BX,400H ; SET POINTER TO NEXT 16K BLK
E401 51	967	PUSH CX ; SAVE REGS
E402 53	968	PUSH BX
E403 52	969	PUSH DX
E404 E811FC	970	CALL STGTST ; GO TEST A 16K BLK OF STG
E407 5A	971	POP DX

```

E408 5B          972      POP     BX                ; RESTORE REGS
E409 59          973      POP     CX
E40A 74E6        974      JE      E21                ; CHECK IF MORE STG TO TEST
975
976      ;----- PRINT FAILING ADDRESS AND XOR'ED PATTERN IF DATA COMPARE ERROR
977
E40C 8CDA        978      MOV     DX,DS                ; CONVERT FAILING HIGH-ORDER
E40E 8AE8        979      MOV     CH,AL                ; SAVE FAILING BIT PATTERN
E410 8AC6        980      MOV     AL,0H                ; GET FAILING ADDR
E412 E81002      981      CALL    XPC_BYTE                ; CONVERT AND PRINT CODE
E415 8AC5        982      MOV     AL,CH                ; GET FAILING BIT PATTERN
E417 E80D02      983      CALL    XPC_BYTE                ; CONVERT AND PRINT CODE
E41A BE67FA90    984      MOV     SI,OFFSET E1                ; SETUP ADDRESS OF ERROR MSG
E41E E89902      985      CALL    P_MSG                 ; PRINT ERROR MSG
E421
E421 EB18        986      E22:    JMP     SHORT TST12                ; GO TO NEXT TEST
E423
E423 1F          989      E23:    POP     DS                ; STG_TEST_DONE
E424 1E          990      PUSH    DS                ; POINT DS TO DATA SEGMENT
E425 8B161500    991      MOV     DX,IO_RAM_SIZE                ; GET IO CHANNEL RAM SIZE
E429 0B02        992      OR      DX,DX                ; SET FLAG RESULT
E42B 740E        993      JZ      TST12                ; NO IO RAM, GO TO NEXT TEST
E42D B90000      994      MOV     CX,0
E430 81FB0010    995      CMP     BX,1000H                ; HAS IO RAM BEEN TESTED
E434 7705        996      JA      TST12                ; YES - GO TO NEXT TEST
E436 BB0010      997      MOV     BX,1000H                ; SETUP BEG LOC FOR IO RAM
E439 EBB7        998      JMP     E21                ; GO TEST IO CHANNEL RAM
999
1000      ;-----
1000      ;      KEYBOARD TEST
1001      ; DESCRIPTION
1002      ;      RESET THE KEYBOARD AND CHECK THAT SCAN CODE
1003      ;      'AA' IS RETURNED TO THE CPU. CHECK FOR STUCK
1004      ;      KEYS.
1005      ;-----
1006      ASSUME DS:DATA
1007      TST12:
E43B
E43B 1F          1008      POP     DS
E43C 803E120001  1009      CMP     MF6_TST,1                ; MANUFACTURING TEST MODE?
E441 742A        1010      JE      F7                ; YES - SKIP KEYBOARD TEST
E443 E8FD01      1011      CALL    KBD_RESET                ; ISSUE SOFTWARE RESET TO KEYBOARD
E446 E31E        1012      JCXZ    F6                ; PRINT ERR MSG IF NO INTERRUPT
E448 B04D        1013      MOV     AL,40H                ; ENABLE KEYBOARD
E44A E661        1014      OUT     PORT_B,AL
E44C 80FBAA      1015      CMP     BL,0AAH                ; SCAN CODE AS EXPECTED?
E44F 7515        1016      JNE     F6                ; NO - DISPLAY ERROR MSG
1017
1018      ;----- CHECK FOR STUCK KEYS
1019
E451 B0CC        1020      MOV     AL,0CCH                ; CLR KBD, SET CLK LINE HIGH
E453 E661        1021      OUT     PORT_B,AL
E455 B04C        1022      MOV     AL,4CH                ; ENABLE KBD,CLK IN NEXT BYTE
E457 E661        1023      OUT     PORT_B,AL
E459 2BC9        1024      SUB     CX,CX
E45B
E45B E2FE        1025      F5:    ; KBD_WAIT
E45D E460        1026      LOOP    F5                ; DELAY FOR A WHILE
E45F 3C00        1027      IN      AL,KBD_IN                ; CHECK FOR STUCK KEYS
E461 740A        1028      CMP     AL,0                ; SCAN CODE = 0?
E463 E0BF01      1029      JE      F7                ; YES - CONTINUE TESTING
E466 BE33FF90    1030      CALL    XPC_BYTE                ; CONVERT AND PRINT
E46A E84D02      1031      F6:    MOV     SI,OFFSET F1                ; GET MSG ADDR
1032      CALL    P_MSG                 ; PRINT MSG ON SCREEN
1033
1034      ;----- SETUP INTERRUPT VECTOR TABLE
1035
E46D
E46D 2BC0        1036      F7:    ; SETUP_INT_TABLE:
E46F 0EC0        1037      SUB     AX,AX
E471 B90800      1038      MOV     ES,AX
E474 1E          1039      MOV     CX,8                ; GET VECTOR CNT
E475 0E          1040      PUSH    DS                ; SAVE DATA SEGMENT
E476 1F          1041      PUSH    CS                ; SETUP DS SEG REG
E477 BEF3FE90    1042      POP     DS
E47B BF2000      1043      MOV     SI,OFFSET VECTOR_TABLE
E47E
E47E AS          1044      MOV     DI,OFFSET INT_PTR
E47F 47          1045      F7A:
E480 47          1046      MOVSM
E47F 47          1047      INC     DI                ; SKIP OVER SEGMENT
E480 47          1048      INC     DI

```

LOC OBJ	LINE	SOURCE
E481 E2FB	1049	LOOP F7A
	1050	-----
	1051	; CASSETTE DATA WRAP TEST :
	1052	; DESCRIPTION :
	1053	; TURN CASSETTE MOTOR OFF. WRITE A BIT OUT TO THE :
	1054	; CASSETTE DATA BUS. VERIFY THAT CASSETTE DATA :
	1055	; READ IS WITHIN A VALID RANGE. :
	1056	-----
	1057	
	1058	;----- TURN THE CASSETTE MOTOR OFF
	1059	
E483	1060	TST13:
E483 1F	1061	POP DS
E484 1E	1062	PUSH DS
E485 B04D	1063	MOV AL,040H ; SET TIMER 2 SPK OUT, AND CASST
E487 E661	1064	OUT PORT_B,AL ; OUT BITS ON, CASSETTE MOT OFF
	1065	
	1066	;----- WRITE A BIT
	1067	
E489 B0FF	1068	MOV AL,0FFH ; DISABLE TIMER INTERRUPTS
E48B E621	1069	OUT INTA01,AL
E48D B0B6	1070	MOV AL,0B6H ; SEL TIM 2, LSB, MSB, MD 3
E48F E643	1071	OUT TIMER+3,AL ; WRITE 8253 CMD/MODE REG
E491 B8D304	1072	MOV AX,1235 ; SET TIMER 2 CNT FOR 1000 USEC
E494 E642	1073	OUT TIMER+2,AL ; WRITE TIMER 2 COUNTER REG
E496 8AC4	1074	MOV AL,AH ; WRITE MSB
E498 E642	1075	OUT TIMER+2,AL
	1076	
	1077	;----- READ CASSETTE INPUT
	1078	
E49A E462	1079	IN AL,PORT_C ; READ VALUE OF CASS IN BIT
E49C 2410	1080	AND AL,10H ; ISOLATE FROM OTHER BITS
E49E A26B00	1081	MOV LAST_VAL,AL
E4A1 E8D514	1082	CALL READ_HALF_BIT
E4A4 E8D214	1083	CALL READ_HALF_BIT
E4A7 E30C	1084	JCXZ F8 ; CAS_ERR
E4A9 81FB4005	1085	CMP BX,MAX_PERIOD
E4AD 7306	1086	JNC F8 ; CAS_ERR
E4AF 81FB1004	1087	CMP BX,MIN_PERIOD
E4B3 7307	1088	JNC ROM_SCAN ; GO TO NEXT TEST IF OK
E4B5	1089	F8: ; CAS_ERR
E4B5 BE39FF90	1090	MOV SI,OFFSET F2 ; CASSETTE WRAP FAILED
E4B9 E8FE01	1091	CALL P_MSG ; GO PRINT ERROR MSG
	1092	-----
	1093	; CHECK FOR OPTIONAL ROM FROM C8000->F4000 IN 2K INCREMENTS :
	1094	; (A VALID MODULE HAS '55AA' IN THE FIRST 2 LOCATIONS, LENGTH :
	1095	; INDICATOR (LENGTH/512) IN THE 3RD LOCATION AND TEST/INIT. :
	1096	; CODE STARTING IN THE 4TH LOCATION.) :
	1097	-----
E4BC	1098	ROM_SCAN:
E4BC BA00C8	1099	MOV DX,0C800H ; SET BEGINNING ADDRESS
E4BF	1100	ROM_SCAN_1:
E4BF 8EDA	1101	MOV DS,DX
E4C1 2B0B	1102	SUB BX,BX ; SET BX=0000
E4C3 8B07	1103	MOV AX,[BX] ; GET 1ST WORD FROM MODULE
E4C5 3D55AA	1104	CMPL AX,0AA55H ; = TO ID WORD?
E4C8 7505	1105	JNZ NEXT_ROM ; PROCEED TO NEXT ROM IF NOT
E4CA E8B701	1106	CALL ROM_CHECK ; GO DO CHECKSUM AND CALL
E4CD EB04	1107	JMP SHORT ARE_WE_DONE ; CHECK FOR END OF ROM SPACE
E4CF	1108	NEXT_ROM:
E4CF 81C28000	1109	ADD DX,0080H ; POINT TO NEXT 2K ADDRESS
E4D3	1110	ARE_WE_DONE:
E4D3 81FA00F6	1111	CMPL DX,0F600H ; AT F6000 YET?
E4D7 7CE6	1112	JL ROM_SCAN_1 ; GO CHECK ANOTHER ADD. IF NOT
E4D9 EB0190	1113	JMP BASE_ROM_CHK ; GO CHECK BASIC ROM
	1114	-----
	1115	; ROS CHECKSUM II :
	1116	; DESCRIPTION :
	1117	; A CHECKSUM IS DONE FOR THE 4 ROS :
	1118	; MODULES CONTAINING BASIC CODE :
	1119	-----
E4DC	1120	BASE_ROM_CHK:
E4DC	1121	E4:
E4DC 2B0B	1122	SUB BX,BX ; SETUP STARTING ROS ADDR
E4DE 8EDA	1123	MOV DS,DX
E4E0 E86907	1124	CALL ROS_CHECKSUM ; CHECK ROS

LOC OBJ	LINE	SOURCE	
E4E3 7403	1125	JE E5	: CONTINUE IF OK
E4E5 E82103	1126	CALL ROM_ERR	: POST ERROR
E4E8	1127	E5:	
E4E8 80C602	1128	ADD DH,02H	: POINT TO NEXT 8K MODDULE
E4EB 80FEFE	1129	CMF DH,0FEH	
E4EE 75EC	1130	JNZ E4	: YES - CONTINUE
E4F0 1F	1131	POP DS	: RECOVER DATA SEG PTR
	1132		
	1133	: DISKETTE ATTACHMENT TEST	:
	1134	: DESCRIPTION	:
	1135	: CHECK IF IPL DISKETTE DRIVE IS ATTACHED TO SYSTEM. IF ATTACHED,	:
	1136	: VERIFY STATUS OF NEC FDC AFTER A RESET. ISSUE A RECAL AND SEEK :	:
	1137	: CMD TO FDC AND CHECK STATUS. COMPLETE SYSTEM INITIALIZATION :	:
	1138	: THEN PASS CONTROL TO THE BOOT LOADER PROGRAM. :	:
	1139		
E4F1	1140	F9:	
E4F1 A01000	1141	MOV AL,BYTE PTR EQUIP_FLAG	: GET SENSE SMS INFO
E4F4 A801	1142	TEST AL,01H	: IPL DISKETTE DRIVE ATTCH?
E4F6 750A	1143	JNZ F10	: NO -SKIP THIS TEST
E4F8 803E120001	1144	CMF MFG_TST,1	: MANUFACTURING TEST MODE?
E4FD 7530	1145	JNE F15A	: NO - GO TO BOOT LOADER
E4FF E959FB	1146	JMP START	: YES - LOOP POWER-ON-DIAGS
E502	1147	F10:	
E502 E421	1148	IN AL,INTA01	: DISK_TEST
E504 24BF	1149	AND AL,0BFH	: ENABLE DISKETTE INTERRUPTS
E506 E621	1150	OUT INTA01,AL	
E508 B400	1151	MOV AH,0	: RESET NEC FDC
E50A 8AD4	1152	MOV DL,AH	: (POINT TO DISKETTE)
E50C CD13	1153	INT 13H	: VERIFY STATUS AFTER RESET
E50E 7221	1154	JC F13	
	1155		
	1156	:----- TURN DRIVE 0 MOTOR ON	
	1157		
E510 BAF203	1158	MOV DX,03F2H	: GET ADDR OF FDC CARD
E513 52	1159	PUSH DX	: SAVE IT
E514 B01C	1160	MOV AL,1CH	: TURN MOTOR ON, EN DMA/INT
E516 EE	1161	OUT DX,AL	: WRITE FDC CONTROL REG
E517 2BC9	1162	SUB CX,CX	
E519	1163	F11:	
E519 E2FE	1164	LOOP F11	: MOTOR_WAIT:
E51B	1165	F12:	: WAIT FOR 1 SECOND
E51B E2FE	1166	LOOP F12	
E51D 3302	1167	XOR DX,DX	: SELECT DRIVE 0
E51F B501	1168	MOV CH,1	: SELECT TRACK 1
E521 88163E00	1169	MOV SEEK_STATUS,DL	
E525 E85509	1170	CALL SEEK	: RECALIBRATE DISKETTE
E528 7207	1171	JC F13	: GO TO ERR SUBROUTINE IF ERR
E52A B522	1172	MOV CH,34	: SELECT TRACK 34
E52C E84E09	1173	CALL SEEK	: SEEK TO TRACK 34
E52F 7307	1174	JNC F14	: OK, TURN MOTOR OFF
E531	1175	F13:	: DSK_ERR:
E531 BEEAFF90	1176	MOV SI,OFFSET F3	: GET ADDR OF MSG
E535 E88201	1177	CALL P_MSG	: GO PRINT ERROR MSG
	1178		
	1179	:----- TURN DRIVE 0 MOTOR OFF	
	1180		
E538	1181	F14:	
E538 B90C	1182	MOV AL,0CH	: DRQ_OFF:
E53A 5A	1183	POP DX	: TURN DRIVE 0 MOTOR OFF
E53B EE	1184	OUT DX,AL	: RECOVER FDC CTL ADDRESS
	1185		
	1186	:----- SETUP PRINTER AND RS232 BASE ADDRESSES IF DEVICE ATTACHED	
	1187		
E53C	1188	F15A:	
E53C BE1E00	1189	MOV SI,OFFSET KB_BUFFER	
E53F 89361A00	1190	MOV BUFFER_HEAD,SI	: SETUP KEYBOARD PARAMETERS
E543 89361C00	1191	MOV BUFFER_TAIL,SI	
E547 89368000	1192	MOV BUFFER_START,SI	: DEFAULT TO STANDARD BUFFER
E54B 83C620	1193	ADD SI,32	: (32 BYTES LONG)
E54E 89368200	1194	MOV BUFFER_END,SI	
E552 E421	1195	IN AL,INTA01	
E554 24FC	1196	AND AL,0FCH	: ENABLE TIMER AND KBD INTS
E556 E621	1197	OUT INTA01,AL	
E558 BD3DE690	1198	MOV BP,OFFSET F4	: PRT_SRC_TBL
E55C 2BF6	1199	SUB SI,SI	
E55E	1200	F16:	
E55E 2E885600	1201	MOV DX,CS:[BP]	: PRT_BASE:
			: GET PRINTER BASE ADDR

LOC	OBJ	LINE	SOURCE
E562	B0AA	1202	MOV AL,0AAH ; WRITE DATA TO PORT A
E564	EE	1203	OUT DX,AL
E565	52	1204	PUSH DX
E566	EC	1205	IN AL,DX ; READ PORT A
E567	5A	1206	POP DX
E568	3CAA	1207	CHP AL,0AAH ; DATA PATTERN SAME
E56A	7505	1208	JNE F17 ; NO - CHECK NEXT PRT CD
E56C	895408	1209	MOV PRINTER_BASE[SI],DX ; YES - STORE PRT BASE ADDR
E56F	46	1210	INC SI ; INCREMENT TO NEXT WORD
E570	46	1211	INC SI
E571		1212	F17: ; NO_STORE:
E571	45	1213	INC BP ; POINT TO NEXT BASE ADDR
E572	45	1214	INC BP
E573	81FD43E6	1215	CHP BP,OFFSET FAE ; ALL POSSIBLE ADDRS CHECKED?
E577	75E5	1216	JNE F16 ; PRT_BASE
E579	2B0B	1217	SUB BX,BX ; POINTER TO RS232 TABLE
E57B	BAFA03	1218	MOV DX,3FAH ; CHECK IF RS232 CD 1 ATTCH?
E57E	EC	1219	IN AL,DX ; READ INTR ID REG
E57F	A0F8	1220	TEST AL,0F8H
E581	7506	1221	JNZ F18
E583	C707F803	1222	MOV RS232_BASE[BX],3F8H ; SETUP RS232 CD #1 ADDR
E587	43	1223	INC BX
E588	43	1224	INC BX
E589		1225	F18:
E589	B602	1226	MOV DH,02H ; CHECK IF RS232 CD 2 ATTCH (AT 2FA)
E58B	EC	1227	IN AL,DX ; READ INTERRUPT ID REG
E58C	A8F8	1228	TEST AL,0F8H
E58E	7506	1229	JNZ F19 ; BASE_END
E590	C707F802	1230	MOV RS232_BASE[BX],2F8H ; SETUP RS232 CD #2
E594	43	1231	INC BX
E595	43	1232	INC BX
		1233	
		1234	;----- SET UP EQUIP FLAG TO INDICATE NUMBER OF PRINTERS AND RS232 CARDS
		1235	
E596		1236	F19: ; BASE_END:
E596	8BC6	1237	MOV AX,SI ; SI HAS 2* NUMBER OF RS232
E598	B103	1238	MOV CL,3 ; SHIFT COUNT
E59A	D2C8	1239	ROR AL,CL ; ROTATE RIGHT 3 POSITIONS
E59C	0AC3	1240	OR AL,BL ; OR IN THE PRINTER COUNT
E59E	A21100	1241	MOV BYTE PTR EQUIP_FLAG+1,AL ; STORE AS SECOND BYTE
E5A1	B201	1242	MOV DL,01H ; DX=201
E5A3	EC	1243	IN AL,DX
E5A4	A80F	1244	TEST AL,0FH
E5A6	7505	1245	JNZ F20 ; NO_GAME_CARD
E5A8	800E110010	1246	OR BYTE PTR EQUIP_FLAG+1,16
E5AD		1247	F20:
		1248	
		1249	;----- SET DEFAULT TIMEOUT VALUES FOR PRINTER AND RS232
		1250	
E5AD	1E	1251	PUSH DS
E5AE	07	1252	POP ES
E5AF	BF7800	1253	MOV DI,OFFSET PRINT_TIM_OUT
E5B2	B81414	1254	MOV AX,1414H ; PRINTER DEFAULTS (COUNT=20)
E5B5	AB	1255	STOSW
E5B6	AB	1256	STOSW
E5B7	B80101	1257	MOV AX,0101H ; RS232 DEFAULTS=01
E5BA	AB	1258	STOSW
E5BB	AB	1259	STOSW
		1260	
		1261	;----- ENABLE NMI INTERRUPTS
		1262	
E5BC	B080	1263	MOV AL,80H ; ENABLE NMI INTERRUPTS
E5BE	E6A0	1264	OUT 0A0H,AL
E5C0	803E120001	1265	CHP MFG_TST,1 ; MFG MODE?
E5C5	7406	1266	JE F21 ; LOAD_BOOT_STRAP
E5C7	BA0100	1267	MOV DX,1
E5CA	E80200	1268	CALL ERR_BEEP ; BEEP 1 SHORT TONE
		1269	
E5CD		1270	F21: ; LOAD_BOOT_STRAP:
E5CD	C019	1271	INT 19H ; BOOTSTRAP
		1272	
		1273	;-----
		1274	; INITIAL RELIABILITY TEST -- SUBROUTINES ;
		1275	;-----
		1276	ASSUME CS:CODE,DS:DATA
		1277	;-----
		1278	; SUBROUTINES FOR POWER ON DIAGNOSTICS ;

```

1279      ; THIS PROCEDURE WILL ISSUE ONE LONG TONE (3 SECS) AND ONE OR :
1280      ; MORE SHORT TONES (1 SEC) TO INDICATE A FAILURE ON THE PLANAR :
1281      ; BOARD, A BAD RAM MODULE, OR A PROBLEM WITH THE CRT. :
1282      ; ENTRY PARAMETERS: :
1283      ; DH = NUMBER OF LONG TONES TO BEEP :
1284      ; DL = NUMBER OF SHORT TONES TO BEEP :
1285      ;-----
1286      ERR_BEEP PROC NEAR
1287          PUSHF                                ; SAVE FLAGS
1288          CLI                                  ; DISABLE SYSTEM INTERRUPTS
1289          PUSH DS                               ; SAVE DS REG CONTENTS
1290          CALL DDS
1291          OR DH,DH                             ; ANY LONG ONES TO BEEP
1292          JZ G3                                 ; NO, DO THE SHORT ONES
1293          G1:                                  ; LONG_BEEP:
1294          MOV BL,6                             ; COUNTER FOR BEEPS
1295          CALL BEEP                           ; DO THE BEEP
1296          G2: LOOP G2                         ; DELAY BETWEEN BEEPS
1297          DEC DH                               ; ANY MORE TO DO
1298          JNZ G1                              ; DO IT
1299          CMP MFG_TST,1                       ; MFG TEST MODE?
1300          JNE G3                              ; YES - CONTINUE BEEPING SPEAKER
1301          MOV AL,0CDH                         ; STOP BLINKING LED
1302          OUT PORT_B,AL
1303          JMP SHORT G1
1304          G3:
1305          MOV BL,1                             ; SHORT_BEEP:
1306          CALL BEEP                           ; COUNTER FOR A SHORT BEEP
1307          G4:                                  ; DO THE SOUND
1308          LOOP G4                             ; DELAY BETWEEN BEEPS
1309          DEC DL                               ; DONE WITH SHORTS
1310          JNZ G3                              ; DO SOME MORE
1311          G5:                                  ; LONG DELAY BEFORE RETURN
1312          LOOP G5
1313          G6:                                  ; LONG DELAY BEFORE RETURN
1314          LOOP G6
1315          POP DS                               ; RESTORE ORIG CONTENTS OF DS
1316          POPF                                ; RESTORE FLAGS TO ORIG SETTINGS
1317          RET                                  ; RETURN TO CALLER
1318      ERR_BEEP ENDP
1319
1320      ;----- ROUTINE TO SOUND BEEPER
1321
1322      BEEP PROC NEAR
1323          MOV AL,10110110B                    ; SEL TIM 2,LSB,MSB,BINARY
1324          OUT TIMER+3,AL                      ; WRITE THE TIMER MODE REG
1325          MOV AX,533H                         ; DIVISOR FOR 1000 HZ
1326          OUT TIMER+2,AL                      ; WRITE TIMER 2 CNT - LSB
1327          MOV AL,AH
1328          OUT TIMER+2,AL                      ; WRITE TIMER 2 CNT - MSB
1329          IN AL,PORT_B                        ; GET CURRENT SETTING OF PORT
1330          MOV AH,AL                           ; SAVE THAT SETTING
1331          OR AL,03                            ; TURN SPEAKER ON
1332          OUT PORT_B,AL
1333          SUB CX,CX                           ; SET CNT TO WAIT 500 MS
1334          G7:
1335          LOOP G7                             ; DELAY BEFORE TURNING OFF
1336          DEC BL                               ; DELAY CNT EXPIRED?
1337          JNZ G7                              ; NO - CONTINUE BEEPING SPK
1338          MOV AL,AH                           ; RECOVER VALUE OF PORT
1339          OUT PORT_B,AL
1340          RET                                  ; RETURN TO CALLER
1341      BEEP ENDP
1342
1343      ;-----
1344      ; CONVERT AND PRINT ASCII CODE :
1345      ; AL MUST CONTAIN NUMBER TO BE CONVERTED. :
1346      ; AX AND BX DESTROYED. :
1347      ;-----
1348      XPC_BYTE PROC NEAR
1349          PUSH AX                               ; RESAVE FOR LOW NIBBLE DISPLAY
1350          MOV CL,4                             ; SHIFT COUNT
1351          SHR AL,CL                            ; NIBBLE SHAP
1352          CALL XLAT_PR                         ; DO THE HIGH NIBBLE DISPLAY
1353          POP AX                               ; RECOVER THE NIBBLE
1354          AND AL,0FH                           ; ISOLATE TO LOW NIBBLE
1355          ; FALL INTO LOW NIBBLE CONVERSION

```

LOC OBJ	LINE	SOURCE	
E630	1356	XLAT_PR PROC NEAR	; CONVERT 00-0F TO ASCII CHARACTER
E630 0490	1357	ADD AL,090H	; ADD FIRST CONVERSION FACTOR
E632 27	1358	DAA	; ADJUST FOR NUMERIC AND ALPHA RANGE
E633 1440	1359	ADC AL,040H	; ADD CONVERSION AND ADJUST LOW NIBBLE
E635 27	1360	DAA	; ADJUST HI NIBBLE TO ASCII RANGE
E636	1361	PRT_HEX PROC NEAR	
E636 B40E	1362	MOV AH,14	; DISPLAY CHAR. IN AL
E638 B700	1363	MOV BH,0	
E63A CD10	1364	INT 10H	; CALL VIDEO_IO
E63C C3	1365	RET	
	1366	PRT_HEX ENDP	
	1367	XLAT_PR ENDP	
	1368	XPC_BYTE ENDP	
	1369		
E63D	1370	F4 LABEL WORD	; PRINTER SOURCE TABLE
:63D BC03	1371	DW 3BCH	
E63F 7803	1372	DW 378H	
E641 7802	1373	DW 278H	
E643	1374	F4E LABEL WORD	
	1375		
	1376		
	1377		
	1378		
	1379		
E643	1380	KBD_RESET PROC NEAR	
E643 B00C	1381	MOV AL,0CH	; SET KBD CLK LINE LOW
E645 E661	1382	OUT PORT_B,AL	; WRITE 8255 PORT B
E647 B95629	1383	MOV CX,10582	; HOLD KBD CLK LOW FOR 20 MS
E64A	1384	G6:	
E64A E2FE	1385	LOOP G6	; LOOP FOR 20 MS
E64C B0CC	1386	MOV AL,0CCH	; SET CLK, ENABLE LINES HIGH
E64E E661	1387	OUT PORT_B,AL	
E650	1388	SP_TEST:	
E650 B04C	1389	MOV AL,4CH	; ENTRY FOR MANUFACTURING TEST 2
E652 E661	1390	OUT PORT_B,AL	; SET KBD CLK HIGH, ENABLE LOW
E654 B0FD	1391	MOV AL,0FDH	
E656 E621	1392	OUT INTA01,AL	; ENABLE KEYBOARD INTERRUPTS
E658 FB	1393	STI	; WRITE 8259 IMR
E659 B400	1394	MOV AH,0	; ENABLE SYSTEM INTERRUPTS
E65B 2BC9	1395	SUB CX,CX	; RESET INTERRUPT INDICATOR
E65D	1396	G9:	; SETUP INTERRUPT TIMEOUT CNT
E65D F6C4FF	1397	TEST AH,0FFH	
E660 7502	1398	JNZ G10	; DID A KEYBOARD INTR OCCUR?
E662 E2F9	1399	LOOP G9	; YES - READ SCAN CODE RETURNED
E664	1400	G10:	; NO - LOOP TILL TIMED OUT
E664 E460	1401	IN AL,PORT_A	; READ KEYBOARD SCAN CODE
E666 8AD8	1402	MOV BL,AL	; SAVE SCAN CODE JUST READ
E668 B0CC	1403	MOV AL,0CCH	; CLEAR KEYBOARD
E66A E661	1404	OUT PORT_B,AL	
E66C C3	1405	RET	; RETURN TO CALLER
	1406	KBD_RESET ENDP	
	1407		
	1408		
	1409		
	1410		
	1411		
E66D	1412	BLINK_INT PROC NEAR	
E66D FB	1413	STI	
E66E 50	1414	PUSH AX	; SAVE AX REG CONTENTS
E66F E461	1415	IN AL,PORT_B	; READ CURRENT VAL OF PORT B
E671 8AE0	1416	MOV AH,AL	
E673 F6D0	1417	NOT AL	; FLIP ALL BITS
E675 2440	1418	AND AL,01000000B	; ISOLATE CONTROL BIT
E677 80E4BF	1419	AND AH,10111111B	; MASK OUT OF ORIGINAL VAL
E67A 0AC4	1420	OR AL,AH	; OR NEW CONTROL BIT IN
E67C E661	1421	OUT PORT_B,AL	
E67E B020	1422	MOV AL,E0I	
E680 E620	1423	OUT INTA00,AL	
E682 58	1424	POP AX	; RESTORE AX REG
E683 CF	1425	IRET	
	1426	BLINK_INT ENDP	
	1427		
	1428		
	1429		
E684	1430	ROM_CHECK PROC NEAR	
E684 B84000	1431	MOV AX,DATA	; SET ES=DATA
E687 8EC0	1432	MOV ES,AX	

```

E689 2AE4      1433      SUB    AH,AH          ; ZERO OUT AH
E68B 8A4702    1434      MOV    AL,[BX+2]      ; GET LENGTH INDICATOR
E68E B109      1435      MOV    CL,09H        ; MULTIPLY BY 512
E690 D3E0      1436      SHL    AX,CL
E692 8BC8      1437      MOV    CX,AX          ; SET COUNT
E694 51        1438      PUSH   CX
E695 B104      1439      MOV    CL,4
E697 D3E8      1440      SHR    AX,CL
E699 03D0      1441      ADD    DX,AX          ; SET POINTER TO NEXT MODULE
E69B 59        1442      POP     CX
                        1443
E69C E8B005    1444      CALL   ROM_CHECKSUM_CNT    ; DO CHECKSUM
E69F 7405      1445      JZ     ROM_CHECK_1
E6A1 E86501    1446      CALL   ROM_ERR          ; PRINT ERROR INFO
E6A4 EB13      1447      JMP     SHORT ROM_CHECK_END
E6A6           1448      ROM_CHECK_1:
E6A6 52        1449      PUSH   DX          ; SAVE POINTER
E6A7 26C7060010300 1450      MOV    ES:IO_ROM_INIT,0003H    ; LOAD OFFSET
E6AE 268C1E0201 1451      MOV    ES:IO_ROM_SEG,DS        ; LOAD SEGMENT
E6B3 26FF1E0001 1452      CALL   DWORD PTR ES:IO_ROM_INIT ; CALL INIT RTN.
E6B8 5A        1453      POP     DX
E6B9           1454      ROM_CHECK_END:
E6B9 C3        1455      RET
                        1456      ROM_CHECK      ENDP
                        1457
                        1458      ;-----
                        1459      ; THIS SUBROUTINE WILL PRINT A MESSAGE ON THE DISPLAY :
                        1460      ;
                        1461      ; ENTRY REQUIREMENTS:
                        1462      ; SI = OFFSET(ADDRESS) OF MESSAGE BUFFER
                        1463      ; CX = MESSAGE BYTE COUNT
                        1464      ; MAXIMUM MESSAGE LENGTH IS 36 CHARACTERS
                        1465      ;-----
E6BA           1466      P_MSG  PROC    NEAR
E6BA E8B118    1467      CALL   DDS
E6BD 803E120001 1468      CHP    HF6_TST,1          ; MFG TEST MODE?
E6C2 7505      1469      JNE     G12              ; NO - DISPLAY ERROR MSG
E6C4 B601      1470      MOV    DH,1          ; YES - SETUP TO BEEP SPEAKER
E6C6 E906FF    1471      JMP     ERR_BEEP        ; YES - BEEP SPEAKER
E6C9           1472      G12:
E6C9 2E8A04    1473      MOV    AL,CS:[SI]      ; WRITE MSG:
E6CC 46        1474      INC     SI              ; PUT CHAR IN AL
E6CD 50        1475      PUSH   AX              ; POINT TO NEXT CHAR
E6CE E865FF    1476      CALL   PRT_HEX          ; SAVE PRINT CHAR
E6D1 58        1477      POP     AX              ; CALL VIDEO_IO
E6D2 3C0A      1478      CHP    AL,10          ; RECOVER PRINT CHAR
E6D4 75F3      1479      JNE     G12              ; WAS IT LINE FEED
E6D6 C3        1480      RET
                        1481      P_MSG  ENDP
                        1482
E6D7 20524F40 1483      F3A   DB      ' ROM',13,10
E6DB 0D
E6DC 0A
                        1484
E6DD           1485      D_EOI  PROC    NEAR
E6DD 50        1486      PUSH   AX
E6DE B020      1487      MOV    AL,20H
E6E0 E62C      1488      OUT    20H,AL
E6E2 58        1489      POP     AX
E6E3 CF        1490      IRET
                        1491      D_EOI  ENDP
                        1492
                        1493      ;--- INT 19 ---
                        1494      ; BOOT STRAP LOADER
                        1495      ; IF A 5 1/4" DISKETTE DRIVE IS AVAILABLE ON THE SYSTEM,
                        1496      ; TRACK 0, SECTOR 1 IS READ INTO THE BOOT LOCATION
                        1497      ; (SEGMENT 0, OFFSET 7C00) AND CONTROL IS TRANSFERRED
                        1498      ; THERE.
                        1499      ;
                        1500      ; IF THERE IS NO DISKETTE DRIVE, OR IF THERE IS A
                        1501      ; HARDWARE ERROR CONTROL IS TRANSFERRED TO THE RESIDENT
                        1502      ; BASIC ENTRY POINT.
                        1503      ;
                        1504      ; IPL ASSUMPTIONS:
                        1505      ; 8255 PORT 60H BIT 0 = 1 IF IPL FROM DISKETTE
                        1506      ;-----
                        1507      ASSUME CS:CODE,DS:ABS0

```

```

1508
1509 ;----- IPL WAS SUCCESSFUL
1510
E6E4      1511      H4:
E6E4 EA007C0000 1512      JMP     BOOT_LOCN
E6F2      1513      ORG     0E6F2H
E6F2      1514      BOOT_STRAP  PROC    NEAR
E6F2 FB      1515      STI             ; ENABLE INTERRUPTS
E6F3 2BC0    1516      SUB     AX,AX
E6F5 8ED8    1517      MOV     DS,AX
1518
1519 ;----- RESET DISKETTE PARAMETER TABLE VECTOR
1520
E6F7 C7067800C7EF 1521      MOV     WORD PTR DISK_POINTER,OFFSET DISK_BASE
E6FD 8C0E7A00    1522      MOV     WORD PTR DISK_POINTER+2,CS
E701 A11004    1523      MOV     AX,DATA_WORD[OFFSET EQUIP_FLAG] ; GET THE EQUIPMENT SWITCHES
E704 A801      1524      TEST    AL,1             ; ISOLATE IPL SENSE SWITCH
E706 741E      1525      JZ      H3             ; GO TO CASSETTE BASIC ENTRY POINT
1526
1527 ;----- MUST LOAD SYSTEM FROM DISKETTE -- CX HAS RETRY COUNT
1528
E708 B90400    1529      MOV     CX,4             ; SET RETRY COUNT
E708      1530      H1:             ; IPL_SYSTEM
E708 51      1531      PUSH    CX             ; SAVE RETRY COUNT
E70C B400      1532      MOV     AH,0             ; RESET THE DISKETTE SYSTEM
E70E CD13      1533      INT     13H             ; DISKETTE_IO
E710 720F      1534      JC      H2             ; IF ERROR, TRY AGAIN
E712 B80102    1535      MOV     AX,201H          ; READ IN THE SINGLE SECTOR
E715 2B02      1536      SUB     DX,DX
E717 8EC2      1537      MOV     ES,DX
E719 BB007C    1538      MOV     BX,OFFSET BOOT_LOCN
E71C B90100    1539      MOV     CX,1             ; SECTOR 1, TRACK 0
E71F CD13      1540      INT     13H             ; DISKETTE_IO
E721 59      1541      H2:             ; RECOVER RETRY COUNT
E722 73C0      1542      JNC     H4             ; CF SET BY UNSUCCESSFUL READ
E724 E2E5      1543      LOOP    H1             ; DO IT FOR RETRY TIMES
1544
1545 ;----- UNABLE TO IPL FROM THE DISKETTE
1546
E726      1547      H3:             ; CASSETTE_JUMP:
E726 CD18      1548      INT     18H             ; USE INTERRUPT VECTOR TO GET TO BASIC
1549      BOOT_STRAP  ENDP
1550
1551 ;-----INT 14-----
1552 ; RS232_IO
1553 ; THIS ROUTINE PROVIDES BYTE STREAM I/O TO THE COMMUNICATIONS
1554 ; PORT ACCORDING TO THE PARAMETERS:
1555 ; (AH)=0 INITIALIZE THE COMMUNICATIONS PORT
1556 ; (AL) HAS PARAMETERS FOR INITIALIZATION
1557 ;
1558 ; 7 6 5 4 3 2 1 0
1559 ; ----- BAUD RATE -- -PARITY-- STOPBIT --WORD LENGTH--
1560 ; 000 - 110 X0 - NONE 0 - 1 10 - 7 BITS
1561 ; 001 - 150 01 - ODD 1 - 2 11 - 8 BITS
1562 ; 010 - 300 11 - EVEN
1563 ; 011 - 600
1564 ; 100 - 1200
1565 ; 101 - 2400
1566 ; 110 - 4800
1567 ; 111 - 9600
1568 ;
1569 ; ON RETURN, CONDITIONS SET AS IN CALL TO COMMO STATUS (AH=3)
1570 ; (AH)=1 SEND THE CHARACTER IN (AL) OVER THE COMMO LINE
1571 ; (AL) REGISTER IS PRESERVED
1572 ; ON EXIT, BIT 7 OF AH IS SET IF THE ROUTINE WAS UNABLE
1573 ; TO TRANSMIT THE BYTE OF DATA OVER THE LINE.
1574 ; IF BIT 7 OF AH IS NOT SET, THE REMAINDER OF AH
1575 ; IS SET AS IN A STATUS REQUEST, REFLECTING THE
1576 ; CURRENT STATUS OF THE LINE.
1577 ; (AH)=2 RECEIVE A CHARACTER IN (AL) FROM COMMO LINE BEFORE
1578 ; RETURNING TO CALLER
1579 ; ON EXIT, AH HAS THE CURRENT LINE STATUS, AS SET BY THE
1580 ; THE STATUS ROUTINE, EXCEPT THAT THE ONLY BITS
1581 ; LEFT ON ARE THE ERROR BITS (7,4,3,2,1)
1582 ; IF AH HAS BIT 7 ON (TIME OUT) THE REMAINING
1583 ; BITS ARE NOT PREDICTABLE.
1584 ;
1585 ;
1586 ;
1587 ;
1588 ;
1589 ;
1590 ;
1591 ;
1592 ;
1593 ;
1594 ;
1595 ;
1596 ;
1597 ;
1598 ;
1599 ;
1600 ;
1601 ;
1602 ;
1603 ;
1604 ;
1605 ;
1606 ;
1607 ;
1608 ;
1609 ;
1610 ;
1611 ;
1612 ;
1613 ;
1614 ;
1615 ;
1616 ;
1617 ;
1618 ;
1619 ;
1620 ;
1621 ;
1622 ;
1623 ;
1624 ;
1625 ;
1626 ;
1627 ;
1628 ;
1629 ;
1630 ;
1631 ;
1632 ;
1633 ;
1634 ;
1635 ;
1636 ;
1637 ;
1638 ;
1639 ;
1640 ;
1641 ;
1642 ;
1643 ;
1644 ;
1645 ;
1646 ;
1647 ;
1648 ;
1649 ;
1650 ;
1651 ;
1652 ;
1653 ;
1654 ;
1655 ;
1656 ;
1657 ;
1658 ;
1659 ;
1660 ;
1661 ;
1662 ;
1663 ;
1664 ;
1665 ;
1666 ;
1667 ;
1668 ;
1669 ;
1670 ;
1671 ;
1672 ;
1673 ;
1674 ;
1675 ;
1676 ;
1677 ;
1678 ;
1679 ;
1680 ;
1681 ;
1682 ;
1683 ;
1684 ;
1685 ;
1686 ;
1687 ;
1688 ;
1689 ;
1690 ;
1691 ;
1692 ;
1693 ;
1694 ;
1695 ;
1696 ;
1697 ;
1698 ;
1699 ;
1700 ;
1701 ;
1702 ;
1703 ;
1704 ;
1705 ;
1706 ;
1707 ;
1708 ;
1709 ;
1710 ;
1711 ;
1712 ;
1713 ;
1714 ;
1715 ;
1716 ;
1717 ;
1718 ;
1719 ;
1720 ;
1721 ;
1722 ;
1723 ;
1724 ;
1725 ;
1726 ;
1727 ;
1728 ;
1729 ;
1730 ;
1731 ;
1732 ;
1733 ;
1734 ;
1735 ;
1736 ;
1737 ;
1738 ;
1739 ;
1740 ;
1741 ;
1742 ;
1743 ;
1744 ;
1745 ;
1746 ;
1747 ;
1748 ;
1749 ;
1750 ;
1751 ;
1752 ;
1753 ;
1754 ;
1755 ;
1756 ;
1757 ;
1758 ;
1759 ;
1760 ;
1761 ;
1762 ;
1763 ;
1764 ;
1765 ;
1766 ;
1767 ;
1768 ;
1769 ;
1770 ;
1771 ;
1772 ;
1773 ;
1774 ;
1775 ;
1776 ;
1777 ;
1778 ;
1779 ;
1780 ;
1781 ;
1782 ;
1783 ;
1784 ;
1785 ;
1786 ;
1787 ;
1788 ;
1789 ;
1790 ;
1791 ;
1792 ;
1793 ;
1794 ;
1795 ;
1796 ;
1797 ;
1798 ;
1799 ;
1800 ;
1801 ;
1802 ;
1803 ;
1804 ;
1805 ;
1806 ;
1807 ;
1808 ;
1809 ;
1810 ;
1811 ;
1812 ;
1813 ;
1814 ;
1815 ;
1816 ;
1817 ;
1818 ;
1819 ;
1820 ;
1821 ;
1822 ;
1823 ;
1824 ;
1825 ;
1826 ;
1827 ;
1828 ;
1829 ;
1830 ;
1831 ;
1832 ;
1833 ;
1834 ;
1835 ;
1836 ;
1837 ;
1838 ;
1839 ;
1840 ;
1841 ;
1842 ;
1843 ;
1844 ;
1845 ;
1846 ;
1847 ;
1848 ;
1849 ;
1850 ;
1851 ;
1852 ;
1853 ;
1854 ;
1855 ;
1856 ;
1857 ;
1858 ;
1859 ;
1860 ;
1861 ;
1862 ;
1863 ;
1864 ;
1865 ;
1866 ;
1867 ;
1868 ;
1869 ;
1870 ;
1871 ;
1872 ;
1873 ;
1874 ;
1875 ;
1876 ;
1877 ;
1878 ;
1879 ;
1880 ;
1881 ;
1882 ;
1883 ;
1884 ;
1885 ;
1886 ;
1887 ;
1888 ;
1889 ;
1890 ;
1891 ;
1892 ;
1893 ;
1894 ;
1895 ;
1896 ;
1897 ;
1898 ;
1899 ;
1900 ;
1901 ;
1902 ;
1903 ;
1904 ;
1905 ;
1906 ;
1907 ;
1908 ;
1909 ;
1910 ;
1911 ;
1912 ;
1913 ;
1914 ;
1915 ;
1916 ;
1917 ;
1918 ;
1919 ;
1920 ;
1921 ;
1922 ;
1923 ;
1924 ;
1925 ;
1926 ;
1927 ;
1928 ;
1929 ;
1930 ;
1931 ;
1932 ;
1933 ;
1934 ;
1935 ;
1936 ;
1937 ;
1938 ;
1939 ;
1940 ;
1941 ;
1942 ;
1943 ;
1944 ;
1945 ;
1946 ;
1947 ;
1948 ;
1949 ;
1950 ;
1951 ;
1952 ;
1953 ;
1954 ;
1955 ;
1956 ;
1957 ;
1958 ;
1959 ;
1960 ;
1961 ;
1962 ;
1963 ;
1964 ;
1965 ;
1966 ;
1967 ;
1968 ;
1969 ;
1970 ;
1971 ;
1972 ;
1973 ;
1974 ;
1975 ;
1976 ;
1977 ;
1978 ;
1979 ;
1980 ;
1981 ;
1982 ;
1983 ;
1984 ;
1985 ;
1986 ;
1987 ;
1988 ;
1989 ;
1990 ;
1991 ;
1992 ;
1993 ;
1994 ;
1995 ;
1996 ;
1997 ;
1998 ;
1999 ;
2000 ;

```

```

1585      : OCCURRED.
1586      : (AH)=3 RETURN THE COMMO PORT STATUS IN (AX)
1587      : AH CONTAINS THE LINE STATUS
1588      : BIT 7 = TIME OUT
1589      : BIT 6 = TRANS SHIFT REGISTER EMPTY
1590      : BIT 5 = TRAN HOLDING REGISTER EMPTY
1591      : BIT 4 = BREAK DETECT
1592      : BIT 3 = FRAMING ERROR
1593      : BIT 2 = PARITY ERROR
1594      : BIT 1 = OVERRUN ERROR
1595      : BIT 0 = DATA READY
1596      : AL CONTAINS THE MODEN STATUS
1597      : BIT 7 = RECEIVED LINE SIGNAL DETECT
1598      : BIT 6 = RING INDICATOR
1599      : BIT 5 = DATA SET READY
1600      : BIT 4 = CLEAR TO SEND
1601      : BIT 3 = DELTA RECEIVE LINE SIGNAL DETECT
1602      : BIT 2 = TRAILING EDGE RING DETECTOR
1603      : BIT 1 = DELTA DATA SET READY
1604      : BIT 0 = DELTA CLEAR TO SEND
1605      :
1606      : (DX) = PARAMETER INDICATING WHICH RS232 CARD (0,1 ALLOWED)
1607      :
1608      : DATA AREA RS232_BASE CONTAINS THE BASE ADDRESS OF THE 8250 ON THE
1609      : CARD LOCATION 400H CONTAINS UP TO 4 RS232 ADDRESSES POSSIBLE
1610      : DATA AREA LABEL RS232_TIM_OUT (BYTE) CONTAINS OUTER LOOP COUNT
1611      : VALUE FOR TIMEOUT (DEFAULT=1)
1612      : OUTPUT
1613      : AX MODIFIED ACCORDING TO PARMS OF CALL
1614      : ALL OTHERS UNCHANGED
1615      : -----
1616      : ASSUME CS:CODE,DS:DATA
1617      : ORG 0E729H
1618      : A1 LABEL WORD ; TABLE OF INIT VALUE
1619      : DW 1047 ; 110 BAUD
1620      : DW 768 ; 150
1621      : DW 384 ; 300
1622      : DW 192 ; 600
1623      : DW 96 ; 1200
1624      : DW 48 ; 2400
1625      : DW 24 ; 4800
1626      : DW 12 ; 9600
1627      :
1628      : RS232_I0 PROC FAR
1629      :
1630      : ;----- VECTOR TO APPROPRIATE ROUTINE
1631      :
1632      : STI ; INTERRUPTS BACK ON
1633      : PUSH DS ; SAVE SEGMENT
1634      : PUSH DX
1635      : PUSH SI
1636      : PUSH DI
1637      : PUSH CX
1638      : PUSH BX
1639      : MOV SI,DX ; RS232 VALUE TO SI
1640      : MOV DI,DX
1641      : SHL SI,1 ; WORD OFFSET
1642      : CALL DDS
1643      : MOV DX,RS232_BASE[SI] ; GET BASE ADDRESS
1644      : OR DX,DX ; TEST FOR 0 BASE ADDRESS
1645      : JZ A3 ; RETURN
1646      : OR AH,AH ; TEST FOR (AH)=0
1647      : JZ A4 ; COMMUN INIT
1648      : DEC AH ; TEST FOR (AH)=1
1649      : JZ A5 ; SEND AL
1650      : DEC AH ; TEST FOR (AH)=2
1651      : JZ A12 ; RECEIVE INTO AL
1652      : A2:
1653      : DEC AH ; TEST FOR (AH)=3
1654      : JNZ A3
1655      : JMP A18 ; COMMUNICATION STATUS
1656      : A3: ; RETURN FROM RS232
1657      : POP BX
1658      : POP CX
1659      : POP DI
1660      : POP SI
1661      : POP DX

```

LOC OBJ	LINE	SOURCE
E767 1F	1662	POP DS
E768 CF	1663	IRET
	1664	
	1665	----- INITIALIZE THE COMMUNICATIONS PORT
	1666	
E769	1667	A4:
E769 8AE0	1668	MOV AH,AL
E76B 83C203	1669	ADD DX,3
E76E B0B0	1670	MOV AL,B0H
E770 EE	1671	OUT DX,AL
	1672	
	1673	----- DETERMINE BAUD RATE DIVISOR
	1674	
E771 8AD4	1675	MOV DL,AH
E773 B104	1676	MOV CL,4
E775 D2C2	1677	ROL DL,CL
E777 81E20E00	1678	AND DX,0EH
E77B BF29E7	1679	MOV DI,OFFSET A1
E77E 03FA	1680	ADD DI,DX
E780 0B14	1681	MOV DX,RS232_BASE[SI]
E782 42	1682	INC DX
E783 2E8A4501	1683	MOV AL,CS:[DI]+1
E787 EE	1684	OUT DX,AL
E788 4A	1685	DEC DX
E789 2E8A05	1686	MOV AL,CS:[DI]
E78C EE	1687	OUT DX,AL
E78D 83C203	1688	ADD DX,3
E790 8AC4	1689	MOV AL,AH
E792 241F	1690	AND AL,01FH
E794 EE	1691	OUT DX,AL
E795 4A	1692	DEC DX
E796 4A	1693	DEC DX
E797 B000	1694	MOV AL,0
E799 EE	1695	OUT DX,AL
E79A EB49	1696	JMP SHORT A10
	1697	
	1698	----- SEND CHARACTER IN (AL) OVER COMMO LINE
	1699	
E79C	1700	A5:
E79C 50	1701	PUSH AX
E79D 83C204	1702	ADD DX,4
E7A0 B003	1703	MOV AL,3
E7A2 EE	1704	OUT DX,AL
E7A3 42	1705	INC DX
E7A4 42	1706	INC DX
E7A5 B730	1707	MOV BH,30H
E7A7 E84800	1708	CALL WAIT_FOR_STATUS
E7AA 7408	1709	JE A9
E7AC	1710	A7:
E7AC 59	1711	POP CX
E7AD 8AC1	1712	MOV AL,CL
E7AF	1713	A8:
E7AF 80CC80	1714	OR AH,B0H
E7B2 EBAE	1715	JMP A3
E7B4	1716	A9:
E7B4 4A	1717	DEC DX
E7B5	1718	A10:
E7B5 B720	1719	MOV BH,20H
E7B7 E83800	1720	CALL WAIT_FOR_STATUS
E7BA 75F0	1721	JNZ A7
E7BC	1722	A11:
E7BC 83EA05	1723	SUB DX,5
E7BF 59	1724	POP CX
E7C0 8AC1	1725	MOV AL,CL
E7C2 EE	1726	OUT DX,AL
E7C3 EB90	1727	JMP A3
	1728	
	1729	----- RECEIVE CHARACTER FROM COMMO LINE
	1730	
E7C5	1731	A12:
E7C5 83C204	1732	ADD DX,4
E7C8 B001	1733	MOV AL,1
E7CA EE	1734	OUT DX,AL
E7CB 42	1735	INC DX
E7CC 42	1736	INC DX
E7CD	1737	A13:
E7CD B720	1738	MOV BH,20H

LOC OBJ	LINE	SOURCE	
E7CF E82000	1739	CALL	WAIT_FOR_STATUS ; TEST FOR DSR
E7D2 750B	1740	JNZ	A8 ; RETURN WITH ERROR
E7D4	1741	A15:	WAIT_DSR_END ; WAIT_DSR_END
E7D4 4A	1742	DEC	DX ; LINE STATUS REGISTER
E7D5	1743	A16:	WAIT_RECV ; WAIT_RECV
E7D5 8701	1744	MOV	BH,1 ; RECEIVE BUFFER FULL
E7D7 E81800	1745	CALL	WAIT_FOR_STATUS ; TEST FOR REC. BUFF. FULL
E7DA 75D3	1746	JNZ	A8 ; SET TIME OUT ERROR
E7DC	1747	A17:	GET_CHAR ; GET_CHAR
E7DC 80E41E	1748	AND	AH,00011110B ; TEST FOR ERR CONDITIONS ON REC CHAR
E7DF 8B14	1749	MOV	DX,RS232_BASE[SI] ; DATA PORT
E7E1 EC	1750	IN	AL,DX ; GET CHARACTER FROM LINE
E7E2 E97DFF	1751	JMP	A3 ; RETURN
	1752		
	1753	;----- COMMO PORT STATUS ROUTINE	
	1754		
E7E5	1755	A18:	
E7E5 8B14	1756	MOV	DX,RS232_BASE[SI]
E7E7 83C205	1757	ADD	DX,5 ; CONTROL PORT
E7EA EC	1758	IN	AL,DX ; GET LINE CONTROL STATUS
E7EB 8AE0	1759	MOV	AH,AL ; PUT IN AH FOR RETURN
E7ED 42	1760	INC	DX ; POINT TO MODEM STATUS REGISTER
E7EE EC	1761	IN	AL,DX ; GET MODEM CONTROL STATUS
E7EF E97DFF	1762	JMP	A3 ; RETURN
	1763	;-----	
	1764	; WAIT FOR STATUS ROUTINE	
	1765	;	
	1766	; ENTRY:	
	1767	; BH=STATUS BIT(S) TO LOOK FOR,	
	1768	; DX=ADDR. OF STATUS REG	
	1769	; EXIT:	
	1770	; ZERO FLAG ON = STATUS FOUND	
	1771	; ZERO FLAG OFF = TIMEOUT.	
	1772	; AH=LAST STATUS READ	
	1773	;-----	
E7F2	1774	WAIT_FOR_STATUS PROC	NEAR
E7F2 8A5D7C	1775	MOV	BL,RS232_TIM_OUT[DI] ; LOAD OUTER LOOP COUNT
E7F5	1776	WFS0:	
E7F5 2BC9	1777	SUB	CX,CX
E7F7	1778	WFS1:	
E7F7 EC	1779	IN	AL,DX ; GET STATUS
E7F8 8AE0	1780	MOV	AH,AL ; MOVE TO AH
E7FA 22C7	1781	AND	AL,BH ; ISOLATE BITS TO TEST
E7FC 3AC7	1782	CHP	AL,BH ; EXACTLY = TO MASK
E7FE 7408	1783	JE	WFS_END ; RETURN WITH ZERO FLAG ON
E800 E2F5	1784	LOOP	WFS1 ; TRY AGAIN
E802 FECB	1785	DEC	BL
E804 75EF	1786	JNZ	WFS0
E806 0AFF	1787	OR	BH,BH ; SET ZERO FLAG OFF
E808	1788	WFS_END:	
E808 C3	1789	RET	
	1790	WAIT_FOR_STATUS ENDP	
	1791	RS232_IO	ENDP
	1792		
	1793	;-----	
	1794	; PRINT ADDRESS AND ERROR MESSAGE FOR ROM CHECKSUM ERRORS	
	1795	;-----	
E809	1796	ROM_ERR PROC	NEAR
E809 52	1797	PUSH	DX ; SAVE POINTER
E80A 50	1798	PUSH	AX
E80B 8CDA	1799	MOV	DX,DS ; GET ADDRESS POINTER
E80D 81FA00C8	1800	CHP	DX,0C800H
E811 7E13	1801	JLE	ROM_ERR_BEEP ; SPECIAL ERROR INDICATION
E813 8AC6	1802	MOV	AL,DH
E815 E80DFF	1803	CALL	XPC_BYTE ; DISPLAY ADDRESS
E818 8AC2	1804	MOV	AL,DL
E81A E808FE	1805	CALL	XPC_BYTE
E81D BED7E6	1806	MOV	SI,OFFSET F3A ; DISPLAY ERROR MSG
E820 E897FE	1807	CALL	P_MSG
E823	1808	ROM_ERR_END:	
E823 58	1809	POP	AX
E824 5A	1810	POP	DX
E825 C3	1811	RET	
E826	1812	ROM_ERR_BEEP:	
E826 BA0201	1813	MOV	DX,0102H ; BEEP 1 LONG, 2 SHORT
E829 E8A3FD	1814	CALL	ERR_BEEP
E82C EBF5	1815	JMP	SHORT ROM_ERR_END


```

1016 ROM_ERR ENDP
1017
1018 ;----- INT 16 -----
1019 ; KEYBOARD I/O
1020 ; THESE ROUTINES PROVIDE KEYBOARD SUPPORT
1021 ; INPUT
1022 ; (AH)=0 READ THE NEXT ASCII CHARACTER STRUCK FROM THE KEYBOARD
1023 ; RETURN THE RESULT IN (AL), SCAN CODE IN (AH)
1024 ; (AH)=1 SET THE Z FLAG TO INDICATE IF AN ASCII CHARACTER IS
1025 ; AVAILABLE TO BE READ.
1026 ; (ZF)=1 -- NO CODE AVAILABLE
1027 ; (ZF)=0 -- CODE IS AVAILABLE
1028 ; IF ZF = 0, THE NEXT CHARACTER IN THE BUFFER TO BE READ
1029 ; IS IN AX, AND THE ENTRY REMAINS IN THE BUFFER
1030 ; (AH)=2 RETURN THE CURRENT SHIFT STATUS IN AL REGISTER
1031 ; THE BIT SETTINGS FOR THIS CODE ARE INDICATED IN THE
1032 ; THE EQUATES FOR KB_FLAG
1033 ; OUTPUT
1034 ; AS NOTED ABOVE, ONLY AX AND FLAGS CHANGED
1035 ; ALL REGISTERS PRESERVED
1036 ;-----
1037 ASSUME CS:CODE,DS:DATA
1038 ORG 0E82EH
1039 KEYBOARD_IO PROC FAR
1040 STI ; INTERRUPTS BACK ON
1041 PUSH DS ; SAVE CURRENT DS
1042 PUSH BX ; SAVE BX TEMPORARILY
1043 CALL DDS
1044 OR AH,AH ; AH=0
1045 JZ K1 ; ASCII_READ
1046 DEC AH ; AH=1
1047 JZ K2 ; ASCII_STATUS
1048 DEC AH ; AH=2
1049 JZ K3 ; SHIFT_STATUS
1050 JMP SHORT INT10_END ; EXIT
1051
1052 ;----- READ THE KEY TO FIGURE OUT WHAT TO DO
1053
1054 K1: ; ASCII READ
1055 STI ; INTERRUPTS BACK ON DURING LOOP
1056 NOP ; ALLOW AN INTERRUPT TO OCCUR
1057 CLI ; INTERRUPTS BACK OFF
1058 MOV BX,BUFFER_HEAD ; GET POINTER TO HEAD OF BUFFER
1059 CMP BX,BUFFER_TAIL ; TEST END OF BUFFER
1060 JZ K1 ; LOOP UNTIL SOMETHING IN BUFFER
1061 MOV AX,[BX] ; GET SCAN CODE AND ASCII CODE
1062 CALL K4 ; MOVE POINTER TO NEXT POSITION
1063 MOV BUFFER_HEAD,BX ; STORE VALUE IN VARIABLE
1064 JMP SHORT INT10_END ; RETURN
1065
1066 ;----- ASCII STATUS
1067
1068 K2:
1069 CLI ; INTERRUPTS OFF
1070 MOV BX,BUFFER_HEAD ; GET HEAD POINTER
1071 CMP BX,BUFFER_TAIL ; IF EQUAL (Z=1) THEN NOTHING THERE
1072 MOV AX,[BX]
1073 STI ; INTERRUPTS BACK ON
1074 POP BX ; RECOVER REGISTER
1075 POP DS ; RECOVER SEGMENT
1076 RET 2 ; THROW AWAY FLAGS
1077
1078 ;----- SHIFT STATUS
1079
1080 K3:
1081 MOV AL,KB_FLAG ; GET THE SHIFT STATUS FLAGS
1082 INT10_END:
1083 POP BX ; RECOVER REGISTER
1084 POP DS ; RECOVER REGISTERS
1085 IRET ; RETURN TO CALLER
1086 KEYBOARD_IO ENDP
1087
1088 ;----- INCREMENT A BUFFER POINTER
1089
1090 K4 PROC NEAR
1091 INC BX ; MOVE TO NEXT WORD IN LIST
1092 INC BX

```

LOC OBJ	LINE	SOURCE
E873 3B1E8200	1893	CMP BX,BUFFER_END ; AT END OF BUFFER?
E877 7504	1894	JNE K5 ; NO, CONTINUE
E879 8B1E8000	1895	MOV BX,BUFFER_START ; YES, RESET TO BUFFER BEGINNING
E87D	1896	K5:
E87D C3	1897	RET
	1898	K4 ENDP
	1899	
	1900	!----- TABLE OF SHIFT KEYS AND MASK VALUES
	1901	
E87E	1902	K6 LABEL BYTE
E87E 52	1903	DB INS_KEY ; INSERT KEY
E87F 3A	1904	DB CAPS_KEY,NUM_KEY,SCROLL_KEY,ALT_KEY,CTL_KEY
E880 45		
E881 46		
E882 38		
E883 1D		
E884 2A	1905	DB LEFT_KEY,RIGHT_KEY
E885 36		
0008	1906	K6L EQU \$-K6
	1907	
	1908	!----- SHIFT MASK TABLE
	1909	
E886	1910	K7 LABEL BYTE
E886 80	1911	DB INS_SHIFT ; INSERT MODE SHIFT
E887 40	1912	DB CAPS_SHIFT,NUM_SHIFT,SCROLL_SHIFT,ALT_SHIFT,CTL_SHIFT
E888 20		
E889 10		
E88A 08		
E88B 04		
E88C 02	1913	DB LEFT_SHIFT,RIGHT_SHIFT
E88D 01		
	1914	
	1915	!----- SCAN CODE TABLES
	1916	
E88E 1B	1917	K8 DB 27,-1,0,-1,-1,-1,30,-1
E88F FF		
E890 00		
E891 FF		
E892 FF		
E893 FF		
E894 1E		
E895 FF		
E896 FF	1918	DB -1,-1,-1,31,-1,127,-1,17
E897 FF		
E898 FF		
E899 1F		
E89A FF		
E89B 7F		
E89C FF		
E89D 11		
E89E 17	1919	DB 23,5,18,20,25,21,9,15
E89F 05		
E8A0 12		
E8A1 14		
E8A2 19		
E8A3 15		
E8A4 09		
E8A5 0F		
E8A6 10	1920	DB 16,27,29,10,-1,1,19
E8A7 1B		
E8A8 1D		
E8A9 0A		
E8AA FF		
E8AB 01		
E8AC 13		
E8AD 04	1921	DB 4,6,7,8,10,11,12,-1,-1
E8AE 06		
E8AF 07		
E8B0 08		
E8B1 0A		
E8B2 0B		
E8B3 0C		
E8B4 FF		
E8B5 FF		
E8B6 FF	1922	DB -1,-1,28,26,24,3,22,2
E8B7 FF		
E8B8 1C		

LOC OBJ

LINE SOURCE

```

E0B9 1A
E0BA 18
E0BB 03
E0BC 16
E0BD 02
E0BE 0E      1923      DB      14,13,-1,-1,-1,-1,-1
E0BF 0D
E0C0 FF
E0C1 FF
E0C2 FF
E0C3 FF
E0C4 FF
E0C5 FF
E0C6 20      1924      DB      ' ', -1
E0C7 FF

E0C8      1925      ;----- CTL TABLE SCAN
E0C8 5E      1926      K9      LABEL BYTE
E0C9 5F      1927      DB      94,95,96,97,98,99,100,101
E0CA 60
E0CB 61
E0CC 62
E0CD 63
E0CE 64
E0CF 65
E0D0 66      1928      DB      102,103,-1,-1,119,-1,132,-1
E0D1 67
E0D2 FF
E0D3 FF
E0D4 77
E0D5 FF
E0D6 84
E0D7 FF
E0D8 73      1929      DB      115,-1,116,-1,117,-1,118,-1
E0D9 FF
E0DA 74
E0DB FF
E0DC 75
E0DD FF
E0DE 76
E0DF FF
E0E0 FF      1930      DB      -1
E0E1      1931      ;----- LC TABLE
E0E1 1B      1932      K10     LABEL BYTE
E0E2 31323334353637      1933      DB      01BH, '1234567890-=' , 00H, 09H
      3839302D3D
E0EE 08
E0EF 09
E0F0 71776572747975      1934      DB      'qwertyuiop! ', 00H, -1, 'asdfghjkl;', 027H
      696F705B5D
E0FC 0D
E0FD FF
E0FE 6173646667686A      686C3B
E908 27
E909 60      1935      DB      60H, -1, 5CH, 'zxcvbnm,./', -1, ' ', -1, ' '
E90A FF
E90B 5C
E90C 7A706376626E6D      2C2E2F
E916 FF
E917 2A
E918 FF
E919 20
E91A FF      1936      DB      -1
E91B      1937      ;----- UC TABLE
E91B 1B      1938      K11     LABEL BYTE
E91C 21402324      1939      DB      27, '1234', 37, 05EH, '&[ ]_*', 08H, 0
E920 25
E921 5E
E922 262A28295F2B
E928 08
E929 0D
E92A 51574552545955      1940      DB      'QWERTYUIOP{ }', 00H, -1, 'ASDFGHJKL:''''
      494F507B7D

```

LOC OBJ	LINE	SOURCE
E936 0D		
E937 FF		
E938 4153444647484A		
4B4C3A22		
E943 7E	1941	DB 07EH,-1,' ZXCVBNM<?>?','-1,0,-1,' ','-1
E944 FF		
E945 7C5A584356424E		
4D3C3E3F		
E950 FF		
E951 00		
E952 FF		
E953 20		
E954 FF		
	1942	;----- UC TABLE SCAN
E955	1943	K12 LABEL BYTE
E955 54	1944	DB 84,85,86,87,88,89,90
E956 55		
E957 56		
E958 57		
E959 58		
E95A 59		
E95B 5A		
E95C 5B	1945	DB 91,92,93
E95D 5C		
E95E 5D		
	1946	;----- ALT TABLE SCAN
E95F	1947	K13 LABEL BYTE
E95F 68	1948	DB 104,105,106,107,108
E960 69		
E961 6A		
E962 6B		
E963 6C		
E964 6D	1949	DB 109,110,111,112,113
E965 6E		
E966 6F		
E967 70		
E968 71		
	1950	;----- NUM STATE TABLE
E969	1951	K14 LABEL BYTE
E969 3738392D343536	1952	DB '789-456+1230.'
2B313233302E		
	1953	;----- BASE CASE TABLE
E976	1954	K15 LABEL BYTE
E976 47	1955	DB 71,72,73,-1,75,-1,77
E977 48		
E978 49		
E979 FF		
E97A 4B		
E97B FF		
E97C 4D		
E97D FF	1956	DB -1,79,80,81,82,83
E97E 4F		
E97F 50		
E980 51		
E981 52		
E982 53		
	1957	
	1958	;----- KEYBOARD INTERRUPT ROUTINE
	1959	
E987	1960	ORG 0E987H
E987	1961	KB_INT PROC FAR
E987 FB	1962	STI ; ALLOW FURTHER INTERRUPTS
E988 50	1963	PUSH AX
E989 53	1964	PUSH BX
E98A 51	1965	PUSH CX
E98B 52	1966	PUSH DX
E98C 56	1967	PUSH SI
E98D 57	1968	PUSH DI
E98E 1E	1969	PUSH DS
E98F 06	1970	PUSH ES
E990 FC	1971	CLD ; FORWARD DIRECTION
E991 E8AA15	1972	CALL DDS
E994 E460	1973	IN AL,KB_DATA ; READ IN THE CHARACTER
E996 50	1974	PUSH AX ; SAVE IT
E997 E461	1975	IN AL,KB_CTL ; GET THE CONTROL PORT
E999 8AE0	1976	MOV AH,AL ; SAVE VALUE
E99B 0C80	1977	OR AL,80H ; RESET BIT FOR KEYBOARD

LOC	OBJ	LINE	SOURCE
E99D	E661	1978	OUT KB_CTL,AL
E99F	06E0	1979	XCHG AH,AL ; GET BACK ORIGINAL CONTROL
E9A1	E661	1980	OUT KB_CTL,AL ; KB HAS BEEN RESET
E9A3	58	1981	POP AX ; RECOVER SCAN CODE
E9A4	0AE0	1982	MOV AH,AL ; SAVE SCAN CODE IN AH ALSO
		1983	
		1984	1----- TEST FOR OVERRUN SCAN CODE FROM KEYBOARD
		1985	
E9A6	3CFF	1986	CMP AL,0FFH ; IS THIS AN OVERRUN CHAR
E9A8	7503	1987	JNZ K16 ; NO, TEST FOR SHIFT KEY
E9AA	E97A02	1988	JMP K62 ; BUFFER_FULL_BEEP
		1989	
		1990	1----- TEST FOR SHIFT KEYS
		1991	
E9AD		1992	K16: ; TEST_SHIFT
E9AD	247F	1993	AND AL,07FH ; TURN OFF THE BREAK BIT
E9AF	0E	1994	PUSH CS
E9B0	07	1995	POP ES ; ESTABLISH ADDRESS OF SHIFT TABLE
E9B1	BF7EE8	1996	MOV DI,OFFSET K6 ; SHIFT KEY TABLE
E9B4	B90800	1997	MOV CX,K6L ; LENGTH
E9B7	F2	1998	REPNE SCASB ; LOOK THROUGH THE TABLE FOR A MATCH
E9B8	AE		
E9B9	0AC4	1999	MOV AL,AH ; RECOVER SCAN CODE
E9BB	7403	2000	JE K17 ; JUMP IF MATCH FOUND
E9BD	E98500	2001	JMP K25 ; IF NO MATCH, THEN SHIFT NOT FOUND
		2002	
		2003	1----- SHIFT KEY FOUND
		2004	
E9C0	01EF7FE8	2005	K17: SUB DI,OFFSET K6+1 ; ADJUST PTR TO SCAN CODE MTCH
E9C4	2EBA586E8	2006	MOV AH,CS:K7[DI] ; GET MASK INTO AH
E9C9	A880	2007	TEST AL,80H ; TEST FOR BREAK KEY
E9CB	7551	2008	JNZ K23 ; BREAK_SHIFT_FOUND
		2009	
		2010	1----- SHIFT MAKE FOUND, DETERMINE SET OR TOGGLE
		2011	
E9CD	80FC10	2012	CMP AH,SCROLL_SHIFT
E9D0	7307	2013	JAE K18 ; IF SCROLL SHIFT OR ABOVE, TOGGLE KEY
		2014	
		2015	1----- PLAIN SHIFT KEY, SET SHIFT ON
		2016	
E9D2	08261700	2017	OR KB_FLAG,AH ; TURN ON SHIFT BIT
E9D6	E98000	2018	JMP K26 ; INTERRUPT_RETURN
		2019	
		2020	1----- TOGGLED SHIFT KEY, TEST FOR 1ST MAKE OR NOT
		2021	
E9D9		2022	K18: ; SHIFT-TOGGLE
E9D9	F606170004	2023	TEST KB_FLAG, CTL_SHIFT ; CHECK CTL SHIFT STATE
E9DE	7565	2024	JNZ K25 ; JUMP IF CTL STATE
E9E0	3C52	2025	CMP AL, INS_KEY ; CHECK FOR INSERT KEY
E9E2	7522	2026	JNZ K22 ; JUMP IF NOT INSERT KEY
E9E4	F606170008	2027	TEST KB_FLAG, ALT_SHIFT ; CHECK FOR ALTERNATE SHIFT
E9E9	755A	2028	JNZ K25 ; JUMP IF ALTERNATE SHIFT
E9EB	F606170020	2029	K19: TEST KB_FLAG, NUM_STATE ; CHECK FOR BASE STATE
E9F0	750D	2030	JNZ K21 ; CHECK IF NUM LOCK IS ON
E9F2	F606170003	2031	TEST KB_FLAG, LEFT_SHIFT+ RIGHT_SHIFT
E9F7	740D	2032	JZ K22 ; JUMP IF BASE STATE
		2033	
E9F9		2034	K20: ; NUMERIC ZERO, NOT INSERT KEY
E9F9	B83052	2035	MOV AX, 5230H ; PUT OUT AN ASCII ZERO
E9FC	E9D601	2036	JMP K57 ; BUFFER_FILL
E9FF		2037	K21: ; MIGHT BE NUMERIC
E9FF	F606170003	2038	TEST KB_FLAG, LEFT_SHIFT+ RIGHT_SHIFT
EA04	74F3	2039	JZ K20 ; JUMP NUMERIC, NOT INSERT
		2040	
EA06		2041	K22: ; SHIFT TOGGLE KEY HIT; PROCESS IT
EA06	04261800	2042	TEST AH,KB_FLAG_1 ; IS KEY ALREADY DEPRESSED
EA0A	7540	2043	JNZ K26 ; JUMP IF KEY ALREADY DEPRESSED
EA0C	08261800	2044	OR KB_FLAG_1,AH ; INDICATE THAT THE KEY IS DEPRESSED
EA10	30261700	2045	XOR KB_FLAG,AH ; TOGGLE THE SHIFT STATE
EA14	3C52	2046	CMP AL,INS_KEY ; TEST FOR 1ST MAKE OF INSERT KEY
EA16	7541	2047	JNE K26 ; JUMP IF NOT INSERT KEY
EA18	B80052	2048	MOV AX,INS_KEY*256 ; SET SCAN CODE INTO AH, 0 INTO AL
EA1B	E9B701	2049	JMP K57 ; PUT INTO OUTPUT BUFFER
		2050	
		2051	1----- BREAK SHIFT FOUND
		2052	
EA1E		2053	K23: ; BREAK-SHIFT-FOUND

LOC OBJ	LINE	SOURCE	
EA1E 80FC10	2054	CHP	AH,SCROLL_SHIFT ; IS THIS A TOGGLE KEY
EA21 731A	2055	JAE	K24 ; YES, HANDLE BREAK TOGGLE
EA23 F604	2056	NOT	AH ; INVERT MASK
EA25 20261700	2057	AND	KB_FLAG,AH ; TURN OFF SHIFT BIT
EA29 3CB8	2058	CHP	AL,ALT_KEY+80H ; IS THIS ALTERNATE SHIFT RELEASE
EA2B 752C	2059	JNE	K26 ; INTERRUPT_RETURN
	2060		
	2061		
	2062		
			;----- ALTERNATE SHIFT KEY RELEASED, GET THE VALUE INTO BUFFER
EA2D A01900	2063	MOV	AL,ALT_INPUT
EA30 B400	2064	MOV	AH,0 ; SCAN CODE OF 0
EA32 68261900	2065	MOV	ALT_INPUT,AH ; ZERO OUT THE FIELD
EA36 3C00	2066	CHP	AL,0 ; WAS THE INPUT=0
EA38 741F	2067	JE	K26 ; INTERRUPT_RETURN
EA3A E9A101	2068	JMP	K58 ; IT WASN'T, SO PUT IN BUFFER
EA3D	2069		BREAK-TOGGLE
EA3D F604	2070	NOT	AH ; INVERT MASK
EA3F 20261800	2071	AND	KB_FLAG_1,AH ; INDICATE NO LONGER DEPRESSED
EA43 EB14	2072	JMP	SHORT K26 ; INTERRUPT_RETURN
	2073		
	2074		;----- TEST FOR HOLD STATE
	2075		
EA45	2076	K25:	; NO-SHIFT-FOUND
EA45 3C80	2077	CHP	AL,80H ; TEST FOR BREAK KEY
EA47 7310	2078	JAE	K26 ; NOTHING FOR BREAK CHARS FROM HERE ON
EA49 F606180008	2079	TEST	KB_FLAG_1,HOLD_STATE ; ARE WE IN HOLD STATE
EA4E 7417	2080	JZ	K28 ; BRANCH AROUND TEST IF NOT
EA50 3C45	2081	CHP	AL,NUM_KEY
EA52 7405	2082	JE	K26 ; CAN'T END HOLD ON NUM_LOCK
EA54 80261800F7	2083	AND	KB_FLAG_1,NOT HOLD_STATE ; TURN OFF THE HOLD STATE BIT
EA59	2084	K26:	INTERRUPT_RETURN
EA59 FA	2085	CLI	; TURN OFF INTERRUPTS
EA5A B020	2086	MOV	AL,EOI ; END OF INTERRUPT COMMAND
EA5C E620	2087	OUT	020H,AL ; SEND COMMAND TO INT CONTROL PORT
EA5E	2088	K27:	INTERRUPT_RETURN-NO-EOI
EA5E 07	2089	POP	ES
EA5F 1F	2090	POP	DS
EA60 5F	2091	POP	DI
EA61 5E	2092	POP	SI
EA62 5A	2093	POP	DX
EA63 59	2094	POP	CX
EA64 5B	2095	POP	BX
EA65 58	2096	POP	AX ; RESTORE STATE
EA66 CF	2097	IRET	; RETURN, INTERRUPTS BACK ON
	2098		; WITH FLAG CHANGE
	2099		
	2100		;----- NOT IN HOLD STATE, TEST FOR SPECIAL CHARS
	2101		
EA67	2102	K28:	; NO-HOLD-STATE
EA67 F606170008	2103	TEST	KB_FLAG,ALT_SHIFT ; ARE WE IN ALTERNATE SHIFT
EA6C 7503	2104	JNZ	K29 ; JUMP IF ALTERNATE SHIFT
EA6E E99100	2105	JMP	K38 ; JUMP IF NOT ALTERNATE
	2106		
	2107		;----- TEST FOR RESET KEY SEQUENCE (CTL ALT DEL)
	2108		
EA71	2109	K29:	; TEST-RESET
EA71 F606170004	2110	TEST	KB_FLAG,CTL_SHIFT ; ARE WE IN CONTROL SHIFT ALSO
EA76 7433	2111	JZ	K31 ; NO_RESET
EA78 3C53	2112	CHP	AL,DEL_KEY ; SHIFT STATE IS THERE, TEST KEY
EA7A 752F	2113	JNE	K31 ; NO_RESET
	2114		
	2115		;----- CTL-ALT-DEL HAS BEEN FOUND, DO I/O CLEANUP
	2116		
EA7C C70672003412	2117	MOV	RESET_FLAG, 1234H ; SET FLAG FOR RESET FUNCTION
EA82 EA5BE000F0	2118	JMP	RESET ; JUMP TO POWER ON DIAGNOSTICS
	2119		
	2120		;----- ALT-INPUT-TABLE
EA87	2121	K30	LABEL BYTE
EA87 52	2122	DB	82,79,80,81,75,76,77
EA88 4F			
EA89 50			
EA8A 51			
EA8B 4B			
EA8C 4C			
EA8D 4D			
EA8E 47	2123	DB	71,72,73 ; 10 NUMBERS ON KEYPAD
EA8F 48			

LOC OBJ	LINE	SOURCE
EA90 49		
EA91 10	2124	;------ SUPER-SHIFT-TABLE
EA92 11	2125	DB 16,17,18,19,20,21,22,23 ; A-Z TYPEWRITER CHARS
EA93 12		
EA94 13		
EA95 14		
EA96 15		
EA97 16		
EA98 17		
EA99 18	2126	DB 24,25,30,31,32,33,34,35
EA9A 19		
EA9B 1E		
EA9C 1F		
EA9D 20		
EA9E 21		
EA9F 22		
EAA0 23		
EAA1 24	2127	DB 36,37,38,44,45,46,47,48
EAA2 25		
EAA3 26		
EAA4 2C		
EAA5 2D		
EAA6 2E		
EAA7 2F		
EAA8 30		
EAA9 31	2128	DB 49,50
EAAA 32		
	2129	
	2130	;------ IN ALTERNATE SHIFT, RESET NOT FOUND
	2131	
EAAB	2132	K31: ; NO-RESET
EAAB 3C39	2133	CMP AL,57 ; TEST FOR SPACE KEY
EAAD 7505	2134	JNE K32 ; NOT THERE
EAAF B020	2135	MOV AL, ; SET SPACE CHAR
EAB1 E92101	2136	JMP K57 ; BUFFER_FILL
	2137	
	2138	;------ LOOK FOR KEY PAD ENTRY
	2139	
EAB4	2140	K32: ; ALT-KEY-PAD
EAB4 BF87EA	2141	MOV DI,OFFSET K30 ; ALT-INPUT-TABLE
EAB7 B90A00	2142	MOV CX,10 ; LOOK FOR ENTRY USING KEYPAD
EABA F2	2143	REPNE SCASB ; LOOK FOR MATCH
EABB AE		
EABC 7512	2144	JNE K33 ; NO-ALT-KEYPAD
EABE 81EF88EA	2145	SUB DI,OFFSET K30+1 ; DI,NOW HAS ENTRY VALUE
EAC2 A01900	2146	MOV AL,ALT_INPUT ; GET THE CURRENT BYTE
EAC5 B40A	2147	MOV AH,10 ; MULTIPLY BY 10
EAC7 F6E4	2148	MUL AH
EAC9 03C7	2149	ADD AX,DI ; ADD IN THE LATEST ENTRY
EACB A21900	2150	MOV ALT_INPUT,AL ; STORE IT AWAY
EACE EB89	2151	JMP K26 ; THROW AWAY THAT KEYSTROKE
	2152	
	2153	;------ LOOK FOR SUPERSHIFT ENTRY
	2154	
EAD0	2155	K33: ; NO-ALT-KEYPAD
EAD0 C606190000	2156	MOV ALT_INPUT,0 ; ZERO ANY PREVIOUS ENTRY INTO INPUT
EAD5 B91A00	2157	MOV CX,26 ; DI,ES ALREADY POINTING
EAD8 F2	2158	REPNE SCASB ; LOOK FOR MATCH IN ALPHABET
EAD9 AE		
EADA 7505	2159	JNE K34 ; NOT FOUND, FUNCTION KEY OR OTHER
EADC B000	2160	MOV AL,0 ; ASCII CODE OF ZERO
EADE E9F400	2161	JMP K57 ; PUT IT IN THE BUFFER
	2162	
	2163	;------ LOOK FOR TOP ROW OF ALTERNATE SHIFT
	2164	
EAE1	2165	K34: ; ALT-TOP-ROW
EAE1 3C02	2166	CMP AL,2 ; KEY WITH '1' ON IT
EAE3 720C	2167	JB K35 ; NOT ONE OF INTERESTING KEYS
EAE5 3C0E	2168	CMP AL,14 ; IS IT IN THE REGION
EAE7 7308	2169	JAE K35 ; ALT-FUNCTION
EAE9 80C476	2170	ADD AH,118 ; CONVERT PSUEDO SCAN CODE TO RANGE
EAE C B000	2171	MOV AL,0 ; INDICATE AS SUCH
EAAE E9E400	2172	JMP K57 ; BUFFER_FILL
	2173	
	2174	;------ TRANSLATE ALTERNATE SHIFT PSEUDO SCAN CODES
	2175	

LOC OBJ	LINE	SOURCE	
EAF1	2176	K35:	; ALT-FUNCTION
EAF1 3C3B	2177	CMP AL,59	; TEST FOR IN TABLE
EAF3 7303	2178	JAE K37	; ALT-CONTINUE
EAF5	2179	K36:	; CLOSE-RETURN
EAF5 E961FF	2180	JMP K26	; IGNORE THE KEY
EAF8	2181	K37:	; ALT-CONTINUE
EAF8 3C47	2182	CMP AL,71	; IN KEYPAD REGION
EAF8 73F9	2183	JAE K36	; IF SO, IGNORE
EAF8 B85FE9	2184	MOV BX,OFFSET K13	; ALT SHIFT PSEUDO SCAN TABLE
EAF8 E91B01	2185	JMP K63	; TRANSLATE THAT
	2186		
	2187	1-----	NOT IN ALTERNATE SHIFT
	2188		
EB02	2189	K38:	; NOT-ALT-SHIFT
EB02 F606170004	2190	TEST KB_FLAG,CTL_SHIFT	; ARE WE IN CONTROL SHIFT
EB07 7458	2191	JZ K44	; NOT-CTL-SHIFT
	2192		
	2193	1-----	CONTROL SHIFT, TEST SPECIAL CHARACTERS
	2194	1-----	TEST FOR BREAK AND PAUSE KEYS
	2195		
EB09 3C46	2196	CMP AL,SCROLL_KEY	; TEST FOR BREAK
EB0B 7518	2197	JNE K39	; NO-BREAK
EB0D 8B1E8000	2198	MOV BX,BUFFER_START	; RESET BUFFER TO EMPTY
EB11 891E1A00	2199	MOV BUFFER_HEAD,BX	
EB15 891E1C00	2200	MOV BUFFER_TAIL,BX	
EB19 C606710080	2201	MOV BIOS_BREAK,80H	; TURN ON BIOS_BREAK BIT
EB1E CD1B	2202	INT 1BH	; BREAK INTERRUPT VECTOR
EB20 28C0	2203	SUB AX,AX	; PUT OUT DUMMY CHARACTER
EB22 E9B000	2204	JMP K57	; BUFFER_FILL
EB25	2205	K39:	; NO-BREAK
EB25 3C45	2206	CMP AL,MUM_KEY	; LOOK FOR PAUSE KEY
EB27 7521	2207	JNE K41	; NO-PAUSE
EB29 800E180008	2208	OR KB_FLAG_1,HOLD_STATE	; TURN ON THE HOLD FLAG
EB2E B020	2209	MOV AL,EOI	; END OF INTERRUPT TO CONTROL PORT
EB30 E620	2210	OUT 020H,AL	; ALLOW FURTHER KEYSTROKE INTS
	2211		
	2212	1-----	DURING PAUSE INTERVAL, TURN CRT BACK ON
	2213		
EB32 803E490007	2214	CMP CRT_MODE,7	; IS THIS BLACK AND WHITE CARD
EB37 7407	2215	JE K40	; YES, NOTHING TO DO
EB39 B4D803	2216	MOV DX,0308H	; PORT FOR COLOR CARD
EB3C A06500	2217	MOV AL,CRT_MODE_SET	; GET THE VALUE OF THE CURRENT MODE
EB3F EE	2218	OUT DX,AL	; SET THE CRT MODE, SO THAT CRT IS ON
EB40	2219	K40:	; PAUSE-LOOP
EB40 F606180008	2220	TEST KB_FLAG_1,HOLD_STATE	
EB45 75F9	2221	JNZ K40	; LOOP UNTIL FLAG TURNED OFF
EB47 E914FF	2222	JMP K27	; INTERRUPT_RETURN_NO_EOI
EB4A	2223	K41:	; NO-PAUSE
	2224		
	2225	1-----	TEST SPECIAL CASE KEY 55
	2226		
EB4A 3C37	2227	CMP AL,55	
EB4C 7506	2228	JNE K42	; NOT-KEY-55
EB4E B80072	2229	MOV AX,114*256	; START/STOP PRINTING SWITCH
EB51 E98100	2230	JMP K57	; BUFFER_FILL
	2231		
	2232	1-----	SET UP TO TRANSLATE CONTROL SHIFT
	2233		
EB54	2234	K42:	; NOT-KEY-55
EB54 B8BEE8	2235	MOV BX,OFFSET K8	; SET UP TO TRANSLATE CTL
EB57 3C3B	2236	CMP AL,59	; IS IT IN TABLE
	2237		; CTL-TABLE-TRANSLATE
EB59 7276	2238	JB K56	; YES, GO TRANSLATE CHAR
EB5B	2239	K43:	; CTL-TABLE-TRANSLATE
EB5B B8C8E8	2240	MOV BX,OFFSET K9	; CTL TABLE SCAN
EB5E E9BC00	2241	JMP K63	; TRANSLATE_SCAN
	2242		
	2243	1-----	NOT IN CONTROL SHIFT
	2244		
EB61	2245	K44:	; NOT-CTL-SHIFT
EB61 3C47	2246	CMP AL,71	; TEST FOR KEYPAD REGION
EB63 732C	2247	JAE K48	; HANDLE KEYPAD REGION
EB65 F606170003	2248	TEST KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT	
EB6A 745A	2249	JZ K54	; TEST FOR SHIFT STATE
	2250		
	2251	1-----	UPPER CASE, HANDLE SPECIAL CASES
	2252		

LOC	OBJ	LINE	SOURCE
EB6C	3C0F	2253	CHP AL,15 ; BACK TAB KEY
EB6E	7505	2254	JNE K45 ; NOT-BACK-TAB
EB70	B800F	2255	MOV AX,15*256 ; SET PSEUDO SCAN CODE
EB73	EB60	2256	JMP SHORT K57 ; BUFFER_FILL
EB75		2257	K45: ; NOT-BACK-TAB
EB75	3C37	2258	CHP AL,55 ; PRINT SCREEN KEY
EB77	7509	2259	JNE K46 ; NOT-PRINT-SCREEN
		2260	
		2261	;----- ISSUE INTERRUPT TO INDICATE PRINT SCREEN FUNCTION
		2262	
EB79	B020	2263	MOV AL,EOI ; END OF CURRENT INTERRUPT
EB7B	E620	2264	OUT 020H,AL ; SO FURTHER THINGS CAN HAPPEN
EB7D	CD05	2265	INT 5H ; ISSUE PRINT SCREEN INTERRUPT
EB7F	E90CFE	2266	JMP K27 ; GO BACK WITHOUT EOI OCCURRING
EB82		2267	K46: ; NOT-PRINT-SCREEN
EB82	3C38	2268	CHP AL,59 ; FUNCTION KEYS
EB84	7206	2269	JB K47 ; NOT-UPPER-FUNCTION
EB86	BB55E9	2270	MOV BX,OFFSET K12 ; UPPER CASE PSEUDO SCAN CODES
EB89	E99100	2271	JMP K63 ; TRANSLATE_SCAN
EB8C		2272	K47: ; NOT-UPPER-FUNCTION
EB8C	BB1BE9	2273	MOV BX,OFFSET K11 ; POINT TO UPPER CASE TABLE
EB8F	EB40	2274	JMP SHORT K56 ; OK, TRANSLATE THE CHAR
		2275	
		2276	;----- KEYPAD KEYS, MUST TEST NUM LOCK FOR DETERMINATION
		2277	
EB91		2278	K48: ; KEYPAD-REGION
EB91	F606170020	2279	TEST KB_FLAG,NUM_STATE ; ARE WE IN NUM_LOCK
EB96	7520	2280	JNZ K52 ; TEST FOR SURE
EB98	F606170003	2281	TEST KB_FLAG,LEFT_SHIFT*RIGHT_SHIFT ; ARE WE IN SHIFT STATE
EB9D	7520	2282	JNZ K53 ; IF SHIFTED, REALLY NUM STATE
		2283	
		2284	;----- BASE CASE FOR KEYPAD
		2285	
EB9F		2286	K49: ; BASE-CASE
EB9F	3C4A	2287	CHP AL,74 ; SPECIAL CASE FOR A COUPLE OF KEYS
EBA1	740B	2288	JE K50 ; MINUS
EBA3	3C4E	2289	CHP AL,78
EBA5	740C	2290	JE K51
EBA7	2C47	2291	SUB AL,71 ; CONVERT ORIGIN
EBA9	BB76E9	2292	MOV BX,OFFSET K15 ; BASE CASE TABLE
EBAC	EB71	2293	JMP SHORT K64 ; CONVERT TO PSEUDO SCAN
EBAE		2294	K50: ;
EBAE	B82D4A	2295	MOV AX,74*256+'-' ; MINUS
EBB1	EB22	2296	JMP SHORT K57 ; BUFFER_FILL
EBB3		2297	K51: ;
EBB3	B82B4E	2298	MOV AX,78*256+'+' ; PLUS
EBB6	EB10	2299	JMP SHORT K57 ; BUFFER_FILL
		2300	
		2301	;----- MIGHT BE NUM LOCK, TEST SHIFT STATUS
		2302	
EBB8		2303	K52: ; ALMOST-NUM-STATE
EBB8	F606170003	2304	TEST KB_FLAG,LEFT_SHIFT*RIGHT_SHIFT ; TEST FOR FUNCTION KEYS
EBB0	75E0	2305	JNZ K49 ; SHIFTED TEMP OUT OF NUM STATE
EBBF		2306	K53: ; REALLY_NUM_STATE
EBBF	2C46	2307	SUB AL,70 ; CONVERT ORIGIN
EBC1	BB69E9	2308	MOV BX,OFFSET K14 ; NUM STATE TABLE
EBC4	EB0B	2309	JMP SHORT K56 ; TRANSLATE_CHAR
		2310	
		2311	;----- PLAIN OLD LOWER CASE
		2312	
EBC6		2313	K54: ; NOT-SHIFT
EBC6	3C38	2314	CHP AL,59 ; TEST FOR FUNCTION KEYS
EBC8	7204	2315	JB K55 ; NOT-LOWER-FUNCTION
EBCA	B000	2316	MOV AL,0 ; SCAN CODE IN AH ALREADY
EBCC	EB07	2317	JMP SHORT K57 ; BUFFER_FILL
EBCE		2318	K55: ; NOT-LOWER-FUNCTION
EBCE	BBE1E8	2319	MOV BX,OFFSET K10 ; LC TABLE
		2320	
		2321	;----- TRANSLATE THE CHARACTER
		2322	
EBD1		2323	K56: ; TRANSLATE-CHAR
EBD1	FEC8	2324	DEC AL ; CONVERT ORIGIN
EBD3	2ED7	2325	XLAT CS:K11 ; CONVERT THE SCAN CODE TO ASCII
		2326	
		2327	;----- PUT CHARACTER INTO BUFFER
		2328	
EBD5		2329	K57: ; BUFFER-FILL

LOC OBJ	LINE	SOURCE	
EB05 3CFF	2330	CMP AL,-1	; IS THIS AN IGNORE CHAR
EB07 741F	2331	JE K59	; YES, DO NOTHING WITH IT
EB09 80FCFF	2332	CMP AH,-1	; LOOK FOR -1 PSEUDO SCAN
EB0C 741A	2333	JE K59	; NEAR_INTERRUPT_RETURN
	2334		
	2335	;----- HANDLE THE CAPS LOCK PROBLEM	
	2336		
EB0E	2337	K58:	; BUFFER-FILL-NOTEST
EB0E F606170040	2338	TEST KB_FLAG,CAPS_STATE	; ARE WE IN CAPS LOCK STATE
EBE3 7420	2339	JZ K61	; SKIP IF NOT
	2340		
	2341	;----- IN CAPS LOCK STATE	
	2342		
EBE5 F60617003	2343	TEST KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT	; TEST FOR SHIFT STATE
EBEA 740F	2344	JZ K60	; IF NOT SHIFT, CONVERT LOWER TO UPPER
	2345		
	2346	;----- CONVERT ANY UPPER CASE TO LOWER CASE	
	2347		
EBEC 3C41	2348	CMP AL,'A'	; FIND OUT IF ALPHABETIC
EBEE 7215	2349	JB K61	; NOT_CAPS_STATE
EBF0 3C5A	2350	CMP AL,'Z'	
EBF2 7711	2351	JA K61	; NOT_CAPS_STATE
EBF4 0420	2352	ADD AL,'a'-'A'	; CONVERT TO LOWER CASE
EBF6 EB0D	2353	JMP SHORT K61	; NOT_CAPS_STATE
EBF8	2354	K59:	; NEAR_INTERRUPT_RETURN
EBF8 E95EFE	2355	JMP K26	; INTERRUPT_RETURN
	2356		
	2357	;----- CONVERT ANY LOWER CASE TO UPPER CASE	
	2358		
EBFB	2359	K60:	; LOWER-TO-UPPER
EBFB 3C61	2360	CMP AL,'a'	; FIND OUT IF ALPHABETIC
EBFD 7206	2361	JB K61	; NOT_CAPS_STATE
EBFF 3C7A	2362	CMP AL,'z'	
EC01 7702	2363	JA K61	; NOT_CAPS_STATE
EC03 2C20	2364	SUB AL,'a'-'A'	; CONVERT TO UPPER CASE
EC05	2365	K61:	; NOT_CAPS_STATE
EC05 8B1E1C00	2366	MOV BX,BUFFER_TAIL	; GET THE END POINTER TO THE BUFFER
EC09 8BF3	2367	MOV SI,BX	; SAVE THE VALUE
EC0B E863FC	2368	CALL K4	; ADVANCE THE TAIL
EC0E 3B1E1A00	2369	CMP BX,BUFFER_HEAD	; HAS THE BUFFER WRAPPED AROUND
EC12 7413	2370	JE K62	; BUFFER_FULL_BEEP
EC14 8904	2371	MOV [SI],AX	; STORE THE VALUE
EC16 891E1C00	2372	MOV BUFFER_TAIL,BX	; MOVE THE POINTER UP
EC1A E93CFE	2373	JMP K26	; INTERRUPT_RETURN
	2374		
	2375	;----- TRANSLATE SCAN FOR PSEUDO SCAN CODES	
	2376		
EC1D	2377	K63:	; TRANSLATE-SCAN
EC1D 2C3B	2378	SUB AL,59	; CONVERT ORIGIN TO FUNCTION KEYS
EC1F	2379	K64:	; TRANSLATE-SCAN-ORGD
EC1F 2ED7	2380	XLAT CS:K9	; CTL TABLE SCAN
EC21 0AE0	2381	MOV AH,AL	; PUT VALUE INTO AH
EC23 B000	2382	MOV AL,0	; ZERO ASCII CODE
EC25 EBAE	2383	JMP K57	; PUT IT INTO THE BUFFER
	2384		
	2385	KB_INT	ENDP
	2386		
	2387	;----- BUFFER IS FULL, SOUND THE BEEPER	
	2388		
EC27	2389	K62:	; BUFFER-FULL-BEEP
EC27 B020	2390	MOV AL,EOI	; END OF INTERRUPT COMMAND
EC29 E620	2391	OUT 20H,AL	; SEND COMMAND TO INT CONTROL PORT
EC2B BB8000	2392	MOV BX,080H	; NUMBER OF CYCLES FOR 1/12 SECOND TONE
EC2E E461	2393	IN AH,KB_CTL	; GET CONTROL INFORMATION
EC30 50	2394	PUSH AX	; SAVE
EC31	2395	K65:	; BEEP-CYCLE
EC31 24FC	2396	AND AL,0FCH	; TURN OFF TIMER GATE AND SPEAKER DATA
EC33 E661	2397	OUT KB_CTL,AL	; OUTPUT TO CONTROL
EC35 B94800	2398	MOV CX,48H	; HALF CYCLE TIME FOR TONE
EC38	2399	K66:	
EC38 E2FE	2400	LOOP K66	; SPEAKER OFF
EC3A 0C02	2401	OR AL,2	; TURN ON SPEAKER BIT
EC3C E661	2402	OUT KB_CTL,AL	; OUTPUT TO CONTROL
EC3E B94800	2403	MOV CX,48H	; SET UP COUNT
EC41	2404	K67:	
EC41 E2FE	2405	LOOP K67	; ANOTHER HALF CYCLE
EC43 4B	2406	DEC BX	; TOTAL TIME COUNT

LOC OBJ	LINE	SOURCE
EC44 75EB	2407	JNZ K65 ; DO ANOTHER CYCLE
EC46 58	2408	POP AX ; RECOVER CONTROL
EC47 E661	2409	OUT KB_CTL,AL ; OUTPUT THE CONTROL
EC49 E912FE	2410	JMP K27
	2411	;-----
	2412	; ROS CHECKSUM SUBROUTINE ;
	2413	;-----
EC4C	2414	ROS_CHECKSUM PROC NEAR ; NEXT_ROS_MODULE
EC4C B90020	2415	MOV CX,0192 ; NUMBER OF BYTES TO ADD
EC4F	2416	ROS_CHECKSUM_CNT: ; ENTRY FOR OPTIONAL ROS TEST
EC4F 32C0	2417	XOR AL,AL
EC51	2418	C26:
EC51 0207	2419	ADD AL,DS:[BX]
EC53 43	2420	INC BX ; POINT TO NEXT BYTE
EC54 E2FB	2421	LOOP C26 ; ADD ALL BYTES IN ROS MODULE
EC56 0AC0	2422	OR AL,AL ; SUM = 0?
EC58 C3	2423	RET
	2424	ROS_CHECKSUM ENDP
	2425	
	2426	;-- INT 13 --
	2427	; DISKETTE I/O ;
	2428	; THIS INTERFACE PROVIDES ACCESS TO THE 5 1/4" DISKETTE DRIVES ;
	2429	; INPUT ;
	2430	; (AH)=0 RESET DISKETTE SYSTEM ;
	2431	; HARD RESET TO NEC, PREPARE COMMAND, RECAL REQUIRED ;
	2432	; ON ALL DRIVES ;
	2433	; (AH)=1 READ THE STATUS OF THE SYSTEM INTO (AL) ;
	2434	; DISKETTE_STATUS FROM LAST OPERATION IS USED ;
	2435	; ;
	2436	; REGISTERS FOR READ/WRITE/VERIFY/FORMAT ;
	2437	; (DL) - DRIVE NUMBER (0-3 ALLOWED, VALUE CHECKED) ;
	2438	; (DH) - HEAD NUMBER (0-1 ALLOWED, NOT VALUE CHECKED) ;
	2439	; (CH) - TRACK NUMBER (0-39, NOT VALUE CHECKED) ;
	2440	; (CL) - SECTOR NUMBER (1-8, NOT VALUE CHECKED, ;
	2441	; NOT USED FOR FORMAT) ;
	2442	; (AL) - NUMBER OF SECTORS (MAX = 8, NOT VALUE CHECKED, NOT USED ;
	2443	; FOR FORMAT) ;
	2444	; (ES:BX) - ADDRESS OF BUFFER (NOT REQUIRED FOR VERIFY) ;
	2445	; ;
	2446	; (AH)=2 READ THE DESIRED SECTORS INTO MEMORY ;
	2447	; (AH)=3 WRITE THE DESIRED SECTORS FROM MEMORY ;
	2448	; (AH)=4 VERIFY THE DESIRED SECTORS ;
	2449	; (AH)=5 FORMAT THE DESIRED TRACK ;
	2450	; FOR THE FORMAT OPERATION, THE BUFFER POINTER (ES,BX) ;
	2451	; MUST POINT TO THE COLLECTION OF DESIRED ADDRESS FIELDS ;
	2452	; FOR THE TRACK. EACH FIELD IS COMPOSED OF 4 BYTES, ;
	2453	; (C,H,R,N), WHERE C = TRACK NUMBER, H=HEAD NUMBER, ;
	2454	; R = SECTOR NUMBER, N= NUMBER OF BYTES PER SECTOR ;
	2455	; (00=128, 01=256, 02=512, 03=1024). THERE MUST BE ONE ;
	2456	; ENTRY FOR EVERY SECTOR ON THE TRACK. THIS INFORMATION ;
	2457	; IS USED TO FIND THE REQUESTED SECTOR DURING READ/WRITE ;
	2458	; ACCESS. ;
	2459	; ;
	2460	; DATA VARIABLE -- DISK_POINTER ;
	2461	; DOUBLE WORD POINTER TO THE CURRENT SET OF DISKETTE PARAMETERS ;
	2462	; OUTPUT ;
	2463	; AH = STATUS OF OPERATION ;
	2464	; STATUS BITS ARE DEFINED IN THE EQUATES FOR ;
	2465	; DISKETTE_STATUS VARIABLE IN THE DATA SEGMENT OF THIS ;
	2466	; MODULE. ;
	2467	; CY = 0 SUCCESSFUL OPERATION (AH=0 ON RETURN) ;
	2468	; CY = 1 FAILED OPERATION (AH HAS ERROR REASON) ;
	2469	; FOR READ/WRITE/VERIFY ;
	2470	; DS,BX,DX,CH,CL PRESERVED ;
	2471	; AL = NUMBER OF SECTORS ACTUALLY READ ;
	2472	; ***** AL MAY NOT BE CORRECT IF TIME OUT ERROR OCCURS ;
	2473	; NOTE: IF AN ERROR IS REPORTED BY THE DISKETTE CODE, THE ;
	2474	; APPROPRIATE ACTION IS TO RESET THE DISKETTE, THEN RETRY ;
	2475	; THE OPERATION. ON READ ACCESSES, NO MOTOR START DELAY ;
	2476	; IS TAKEN, SO THAT THREE RETRIES ARE REQUIRED ON READS ;
	2477	; TO ENSURE THAT THE PROBLEM IS NOT DUE TO MOTOR ;
	2478	; START-UP. ;
	2479	;-----
	2480	ASSUME CS:CODE,DS:DATA,ES:DATA
EC59	2481	ORG 0EC59H
EC59	2482	DISKETTE_IO PROC FAR
EC59 FB	2483	STI ; INTERRUPTS BACK ON

LOC OBJ	LINE	SOURCE	
EC5A 53	2484	PUSH BX	; SAVE ADDRESS
EC5B 51	2485	PUSH CX	
EC5C 1E	2486	PUSH DS	; SAVE SEGMENT REGISTER VALUE
EC5D 56	2487	PUSH SI	; SAVE ALL REGISTERS DURING OPERATION
EC5E 57	2488	PUSH DI	
EC5F 55	2489	PUSH BP	
EC60 52	2490	PUSH DX	
EC61 8BEC	2491	MOV BP,SP	; SET UP POINTER TO HEAD PARAM
EC63 E8D812	2492	CALL DDS	
EC66 E81C00	2493	CALL J1	; CALL THE REST TO ENSURE DS RESTORED
EC69 B80400	2494	MOV BX,4	; GET THE MOTOR WAIT PARAMETER
EC6C E8FD01	2495	CALL GET_PARAM	
EC6F 88264000	2496	MOV MOTOR_COUNT,AH	; SET THE TIMER COUNT FOR THE MOTOR
EC73 8A264100	2497	MOV AH,DISKETTE_STATUS	; GET STATUS OF OPERATION
EC77 80FC01	2498	CMP AH,1	; SET THE CARRY FLAG TO INDICATE
EC7A F5	2499	CMC	; SUCCESS OR FAILURE
EC7B 5A	2500	POP DX	; RESTORE ALL REGISTERS
EC7C 5D	2501	POP BP	
EC7D 5F	2502	POP DI	
EC7E 5E	2503	POP SI	
EC7F 1F	2504	POP DS	
EC80 59	2505	POP CX	
EC81 5B	2506	POP BX	; RECOVER ADDRESS
EC82 CA0200	2507	RET 2	; THROW AWAY SAVED FLAGS
	2508	DISKETTE_IO	
	2509	ENDP	
EC85	2510	J1 PROC NEAR	
EC85 8AF0	2511	MOV DH,AL	; SAVE # SECTORS IN DH
EC87 80263F007F	2512	AND MOTOR_STATUS,07FH	; INDICATE A READ OPERATION
EC8C 0AE4	2513	OR AH,AH	; AH=0
EC8E 7427	2514	JZ DISK_RESET	
EC90 FECC	2515	DEC AH	; AH=1
EC92 7473	2516	JZ DISK_STATUS	
EC94 C606410000	2517	MOV DISKETTE_STATUS,0	; RESET THE STATUS INDICATOR
EC99 80FA04	2518	CHP DL,4	; TEST FOR DRIVE IN 0-3 RANGE
EC9C 7313	2519	JAE J3	; ERROR IF ABOVE
EC9E FECC	2520	DEC AH	; AH=2
ECA0 7469	2521	JZ DISK_READ	
ECA2 FECC	2522	DEC AH	; AH=3
ECA4 7503	2523	JNZ J2	; TEST_DISK_VERF
ECA6 E99500	2524	JMP DISK_WRITE	
ECA9	2525	J2: TEST_DISK_VERF	
ECA9 FECC	2526	DEC AH	; AH=4
ECAB 7467	2527	JZ DISK_VERF	
ECAD FECC	2528	DEC AH	; AH=5
ECAF 7467	2529	JZ DISK_FORMAT	
ECB1	2530	J3: BAD_COMMAND	
ECB1 C606410001	2531	MOV DISKETTE_STATUS,BAD_CMD	; ERROR CODE, NO SECTORS TRANSFERRED
ECB6 C3	2532	RET	; UNDEFINED OPERATION
	2533	J1 ENDP	
	2534		
	2535	;----- RESET THE DISKETTE SYSTEM	
	2536		
ECB7	2537	DISK_RESET PROC NEAR	
ECB7 BAF203	2538	MOV DX,03F2H	; ADAPTER CONTROL PORT
ECBA FA	2539	CLI	; NO INTERRUPTS
ECBB A03F00	2540	MOV AL,MOTOR_STATUS	; WHICH MOTOR IS ON
ECBE B104	2541	MOV CL,4	; SHIFT COUNT
ECC0 D2E0	2542	SAL AL,CL	; MOVE MOTOR VALUE TO HIGH NYBBLE
ECC2 A820	2543	TEST AL, 20H	; SELECT CORRESPONDING DRIVE
ECC4 750C	2544	JNZ J5	; JUMP IF MOTOR ONE IS ON
ECC6 A840	2545	TEST AL, 40H	
ECC8 7506	2546	JNZ J4	; JUMP IF MOTOR TWO IS ON
ECCA A880	2547	TEST AL, 80H	
ECCC 7406	2548	JZ J6	; JUMP IF MOTOR ZERO IS ON
ECCE FEC0	2549	INC AL	
ECDD	2550	J4: INC AL	
ECDD FEC0	2551	INC AL	
ECDD	2552	J5: INC AL	
ECDD FEC0	2553	INC AL	
ECDD	2554	J6: INC AL	
ECDD 0C08	2555	OR AL,8	; TURN ON INTERRUPT ENABLE
ECDD EE	2556	OUT DX,AL	; RESET THE ADAPTER
ECDD C6063E0000	2557	MOV SEEK_STATUS,0	; SET RECAL REQUIRED ON ALL DRIVES
ECDD C606410000	2558	MOV DISKETTE_STATUS,0	; SET OK STATUS FOR DISKETTE
ECE1 0C04	2559	OR AL,4	; TURN OFF RESET
ECE3 EE	2560	OUT DX,AL	; TURN OFF THE RESET

LOC OBJ	LINE	SOURCE	
ECE4 FB	2561	STI	; REENABLE THE INTERRUPTS
ECE5 E82A02	2562	CALL CHK_STAT_2	; DO SENSE INTERRUPT STATUS
	2563		; FOLLOWING RESET
ECE8 A04200	2564	MOV AL,NEC_STATUS	; IGNORE ERROR RETURN AND DO OWN TEST
ECB8 3CC0	2565	CMF AL,0C0H	; TEST FOR DRIVE READY TRANSITION
ECED 7406	2566	JZ J7	; EVERYTHING OK
ECF3 800E410020	2567	OR DISKETTE_STATUS,BAD_NEC	; SET ERROR CODE
ECF4 C3	2568	RET	
	2569		
	2570	;----- SEND SPECIFY COMMAND TO NEC	
	2571		
ECF5	2572	J7:	; DRIVE_READY
ECF5 B403	2573	MOV AH,03H	; SPECIFY COMMAND
ECF7 E84701	2574	CALL NEC_OUTPUT	; OUTPUT THE COMMAND
ECFA BB0100	2575	MOV BX,1	; FIRST BYTE PARM IN BLOCK
ECFD E86C01	2576	CALL GET_PARM	; TO THE NEC CONTROLLER
ED00 BB0300	2577	MOV BX,3	; SECOND BYTE PARM IN BLOCK
ED03 E86601	2578	CALL GET_PARM	; TO THE NEC CONTROLLER
ED06	2579	J8:	; RESET_RET
ED06 C3	2580	RET	; RETURN TO CALLER
	2581	DISK_RESET	ENDP
	2582		
	2583	;----- DISKETTE STATUS ROUTINE	
	2584		
ED07	2585	DISK_STATUS	PROC NEAR
ED07 A04100	2586	MOV AL,DISKETTE_STATUS	
ED0A C3	2587	RET	
	2588	DISK_STATUS	ENDP
	2589		
	2590	;----- DISKETTE READ	
	2591		
ED0B	2592	DISK_READ	PROC NEAR
ED0B B046	2593	MOV AL,046H	; READ COMMAND FOR DMA
ED0D	2594	J9:	; DISK_READ_CONT
ED0D E8B801	2595	CALL DMA_SETUP	; SET UP THE DMA
ED10 B4E6	2596	MOV AH,0E6H	; SET UP RD COMMAND FOR NEC CONTROLLER
ED12 EB36	2597	JMP SHORT RM_OPN	; GO DO THE OPERATION
	2598	DISK_READ	ENDP
	2599		
	2600	;----- DISKETTE VERIFY	
	2601		
ED14	2602	DISK_VERF	PROC NEAR
ED14 B042	2603	MOV AL,042H	; VERIFY COMMAND FOR DMA
ED16 EB55	2604	JMP J9	; DO AS IF DISK READ
	2605	DISK_VERF	ENDP
	2606		
	2607	;----- DISKETTE FORMAT	
	2608		
ED18	2609	DISK_FORMAT	PROC NEAR
ED18 800E3F0080	2610	OR MOTOR_STATUS,80H	; INDICATE WRITE OPERATION
ED1D B04A	2611	MOV AL,04AH	; WILL WRITE TO THE DISKETTE
ED1F E8A601	2612	CALL DMA_SETUP	; SET UP THE DMA
ED22 B44D	2613	MOV AH,04DH	; ESTABLISH THE FORMAT COMMAND
ED24 EB24	2614	JMP SHORT RM_OPN	; DO THE OPERATION
ED26	2615	J10:	; CONTINUATION OF RM_OPN FOR FMT
ED26 BB0700	2616	MOV BX,7	; GET THE
ED29 E84001	2617	CALL GET_PARM	; BYTES/SECTOR VALUE TO NEC
ED2C BB0900	2618	MOV BX,9	; GET THE
ED2F E83A01	2619	CALL GET_PARM	; SECTORS/TRACK VALUE TO NEC
ED32 BB0F00	2620	MOV BX,15	; GET THE
ED35 E83401	2621	CALL GET_PARM	; GAP LENGTH VALUE TO NEC
ED38 BB1100	2622	MOV BX,17	; GET THE FILLER BYTE
ED3B E9AB00	2623	JMP J16	; TO THE CONTROLLER
	2624	DISK_FORMAT	ENDP
	2625		
	2626	;----- DISKETTE WRITE ROUTINE	
	2627		
ED3E	2628	DISK_WRITE	PROC NEAR
ED3E 800E3F0080	2629	OR MOTOR_STATUS,80H	; INDICATE WRITE OPERATION
ED43 B04A	2630	MOV AL,04AH	; DMA WRITE COMMAND
ED45 E88001	2631	CALL DMA_SETUP	
ED48 B4C5	2632	MOV AH,0C5H	; NEC COMMAND TO WRITE TO DISKETTE
	2633	DISK_WRITE	ENDP
	2634		
	2635	;----- ALLOW WRITE ROUTINE TO FALL INTO RM_OPN	
	2636		
	2637	;-----	

```

2638      ; RW_OPN
2639      ; THIS ROUTINE PERFORMS THE READ/WRITE/VERIFY OPERATION :
2640      ;-----
2641      RW_OPN PROC NEAR
2642          JNC J11 ; TEST FOR DMA ERROR
2643          MOV     DISKETTE_STATUS,DMA_BOUNDARY ; SET ERROR
2644          MOV     AL,0 ; NO SECTORS TRANSFERRED
2645          RET     ; RETURN TO MAIN ROUTINE
2646      J11: ; DO_RW_OPN
2647          PUSH    AX ; SAVE THE COMMAND
2648
2649      ;----- TURN ON THE MOTOR AND SELECT THE DRIVE
2650
2651          PUSH    CX ; SAVE THE T/S PARMS
2652          MOV     CL,DL ; GET DRIVE NUMBER AS SHIFT COUNT
2653          MOV     AL,1 ; MASK FOR DETERMINING MOTOR BIT
2654          SAL     AL,CL ; SHIFT THE MASK BIT
2655          CLI     ; NO INTERRUPTS WHILE DETERMINING
2656          ; MOTOR STATUS
2657          MOV     MOTOR_COUNT,OFFH ; SET LARGE COUNT DURING OPERATION
2658          TEST    AL,MOTOR_STATUS ; TEST THAT MOTOR FOR OPERATING
2659          JNZ     J14 ; IF RUNNING, SKIP THE WAIT
2660          AND     MOTOR_STATUS,0F0H ; TURN OFF ALL MOTOR BITS
2661          OR      MOTOR_STATUS,AL ; TURN ON THE CURRENT MOTOR
2662          STI     ; INTERRUPTS BACK ON
2663          MOV     AL,10H ; MASK BIT
2664          SAL     AL,CL ; DEVELOP BIT MASK FOR MOTOR ENABLE
2665          OR      AL,DL ; GET DRIVE SELECT BITS IN
2666          OR      AL,0CH ; NO RESET, ENABLE DMA/INT
2667          PUSH    DX ; SAVE REG
2668          MOV     DX,03F2H ; CONTROL PORT ADDRESS
2669          OUT     DX,AL
2670          POP     DX ; RECOVER REGISTERS
2671
2672      ;----- WAIT FOR MOTOR IF WRITE OPERATION
2673
2674          TEST    MOTOR_STATUS,80H ; IS THIS A WRITE
2675          JZ      J14 ; NO, CONTINUE WITHOUT WAIT
2676          MOV     BX,20 ; GET THE MOTOR WAIT
2677          CALL    GET_PARM ; PARAMETER
2678          OR      AH,AH ; TEST FOR NO WAIT
2679      J12: ; TEST_WAIT_TIME
2680          JZ      J14 ; EXIT WITH TIME EXPIRED
2681          SUB     CX,CX ; SET UP 1/8 SECOND LOOP TIME
2682      J13:
2683          LOOP    J13 ; WAIT FOR THE REQUIRED TIME
2684          DEC     AH ; DECREMENT TIME VALUE
2685          JMP     J12 ; ARE WE DONE YET
2686      J14: ; MOTOR_RUNNING
2687          STI     ; INTERRUPTS BACK ON FOR BYPASS WAIT
2688          POP     CX
2689
2690      ;----- DO THE SEEK OPERATION
2691
2692          CALL    SEEK ; MOVE TO CORRECT TRACK
2693          POP     AX ; RECOVER COMMAND
2694          MOV     BH,AH ; SAVE COMMAND IN BH
2695          MOV     DH,0 ; SET NO SECTORS READ IN CASE OF ERROR
2696          JC      J17 ; IF ERROR, THEN EXIT AFTER MOTOR OFF
2697          MOV     SI,OFFSET J17 ; DUMMY RETURN ON STACK FOR NEC_OUTPUT
2698          PUSH    SI ; SO THAT IT WILL RETURN TO MOTOR OFF
2699          ; LOCATION
2700
2701      ;----- SEND OUT THE PARAMETERS TO THE CONTROLLER
2702
2703          CALL    NEC_OUTPUT ; OUTPUT THE OPERATION COMMAND
2704          MOV     AH,[BP+1] ; GET THE CURRENT HEAD NUMBER
2705          SAL     AH,1 ; MOVE IT TO BIT 2
2706          SAL     AH,1
2707          AND     AH,4 ; ISOLATE THAT BIT
2708          OR      AH,DL ; OR IN THE DRIVE NUMBER
2709          CALL    NEC_OUTPUT
2710
2711      ;----- TEST FOR FORMAT COMMAND
2712
2713          CMP     BH,040H ; IS THIS A FORMAT OPERATION
2714          JNE     J15 ; NO. CONTINUE WITH R/W/V

```

LOC OBJ	LINE	SOURCE
EDC1 E962FF	2715	JMP J10 ; IF SO, HANDLE SPECIAL
EDC4	2716	J15:
EDC4 8AE5	2717	MOV AH,CH ; CYLINDER NUMBER
EDC6 E87800	2718	CALL NEC_OUTPUT
EDC9 8A6601	2719	MOV AH,[BP+1] ; HEAD NUMBER FROM STACK
EDCC E87200	2720	CALL NEC_OUTPUT
EDCF 8AE1	2721	MOV AH,CL ; SECTOR NUMBER
EDD1 E86000	2722	CALL NEC_OUTPUT
EDD4 B80700	2723	MOV BX,7 ; BYTES/SECTOR PARM FROM BLOCK
EDD7 E89200	2724	CALL GET_PARM ; TO THE NEC
EDDA B80900	2725	MOV BX,9 ; EOT PARM FROM BLOCK
EDDD E88C00	2726	CALL GET_PARM ; TO THE NEC
EDE0 B80B00	2727	MOV BX,11 ; GAP LENGTH PARM FROM BLOCK
EDE3 E88600	2728	CALL GET_PARM ; TO THE NEC
EDE6 B80D00	2729	MOV BX,13 ; DTL PARM FROM BLOCK
EDE9	2730	J16: ; RM_OPN_FINISH
EDE9 E88000	2731	CALL GET_PARM ; TO THE NEC
EDEC 5E	2732	POP SI ; CAN NOW DISCARD THAT DUMMY
	2733	; RETURN ADDRESS
	2734	
	2735	;----- LET THE OPERATION HAPPEN
	2736	
EDED E84301	2737	CALL WAIT_INT ; WAIT FOR THE INTERRUPT
EDF0	2738	J17: ; MOTOR_OFF
EDF0 7245	2739	JC J21 ; LOOK FOR ERROR
EDF2 E87401	2740	CALL RESULTS ; GET THE NEC STATUS
EDF5 723F	2741	JC J20 ; LOOK FOR ERROR
	2742	
	2743	;----- CHECK THE RESULTS RETURNED BY THE CONTROLLER
	2744	
EDF7 FC	2745	CLD ; SET THE CORRECT DIRECTION
EDF8 BE4200	2746	MOV SI,OFFSET NEC_STATUS ; POINT TO STATUS FIELD
EDFB AC	2747	LODS NEC_STATUS ; GET ST0
EDFC 24C0	2748	AND AL,0C0H ; TEST FOR NORMAL TERMINATION
EDFE 743B	2749	JZ J22 ; OPN_OK
EE00 3C40	2750	CMPL AL,040H ; TEST FOR ABNORMAL TERMINATION
EE02 7529	2751	JNZ J18 ; NOT ABNORMAL, BAD NEC
	2752	
	2753	;----- ABNORMAL TERMINATION, FIND OUT WHY
	2754	
EE04 AC	2755	LODS NEC_STATUS ; GET ST1
EE05 D0E0	2756	SAL AL,1 ; TEST FOR EOT FOUND
EE07 B404	2757	MOV AH,RECORD_NOT_FND
EE09 7224	2758	JC J19 ; RM_FAIL
EE0B D0E0	2759	SAL AL,1
EE0D D0E0	2760	SAL AL,1 ; TEST FOR CRC ERROR
EE0F B410	2761	MOV AH,BAD_CRC
EE11 721C	2762	JC J19 ; RM_FAIL
EE13 D0E0	2763	SAL AL,1 ; TEST FOR DMA OVERRUN
EE15 B408	2764	MOV AH,BAD_DMA
EE17 7216	2765	JC J19 ; RM_FAIL
EE19 D0E0	2766	SAL AL,1
EE1B D0E0	2767	SAL AL,1 ; TEST FOR RECORD NOT FOUND
EE1D B404	2768	MOV AH,RECORD_NOT_FND
EE1F 720E	2769	JC J19 ; RM_FAIL
EE21 D0E0	2770	SAL AL,1
EE23 B403	2771	MOV AH,WRITE_PROTECT ; TEST FOR WRITE_PROTECT
EE25 7208	2772	JC J19 ; RM_FAIL
EE27 D0E0	2773	SAL AL,1 ; TEST MISSING ADDRESS MARK
EE29 B402	2774	MOV AH,BAD_ADDR_MARK
EE2B 7202	2775	JC J19 ; RM_FAIL
	2776	
	2777	;----- NEC MUST HAVE FAILED
	2778	
EE2D	2779	J18: ; RM-NEC-FAIL
EE2D B420	2780	MOV AH,BAD_NEC
EE2F	2781	J19: ; RM-FAIL
EE2F 08264100	2782	OR DISKETTE_STATUS,AH
EE33 E87801	2783	CALL NUM_TRANS ; HOW MANY WERE REALLY TRANSFERRED
EE36	2784	J20: ; RM_ERR
EE36 C3	2785	RET ; RETURN TO CALLER
EE37	2786	J21: ; RM_ERR_RES
EE37 E82F01	2787	CALL RESULTS ; FLUSH THE RESULTS BUFFER
EE3A C3	2788	RET
	2789	
	2790	;----- OPERATION WAS SUCCESSFUL
	2791	

LOC OBJ	LINE	SOURCE
EE3B	2792	J22: ; OPN_OK
EE3B E87001	2793	CALL NUM_TRANS ; HOW MANY GOT MOVED
EE3E 32E4	2794	XOR AH,AH ; NO ERRORS
EE40 C3	2795	RET
	2796	RM_OPN ENDP
	2797	-----
	2798	; NEC_OUTPUT
	2799	; THIS ROUTINE SENDS A BYTE TO THE NEC CONTROLLER AFTER TESTING
	2800	; FOR CORRECT DIRECTION AND CONTROLLER READY THIS ROUTINE WILL
	2801	; TIME OUT IF THE BYTE IS NOT ACCEPTED WITHIN A REASONABLE
	2802	; AMOUNT OF TIME, SETTING THE DISKETTE STATUS ON COMPLETION.
	2803	; INPUT
	2804	; (AH) BYTE TO BE OUTPUT
	2805	; OUTPUT
	2806	; CY = 0 SUCCESS
	2807	; CY = 1 FAILURE -- DISKETTE STATUS UPDATED
	2808	; IF A FAILURE HAS OCCURRED, THE RETURN IS MADE ONE LEVEL
	2809	; HIGHER THAN THE CALLER OF NEC_OUTPUT.
	2810	; THIS REMOVES THE REQUIREMENT OF TESTING AFTER EVERY
	2811	; CALL OF NEC_OUTPUT.
	2812	; (AL) DESTROYED
	2813	-----
EE41	2814	NEC_OUTPUT PROC NEAR
EE41 52	2815	PUSH DX ; SAVE REGISTERS
EE42 51	2816	PUSH CX
EE43 BAF403	2817	MOV DX,03F4H ; STATUS PORT
EE46 33C9	2818	XOR CX,CX ; COUNT FOR TIME OUT
EE48	2819	J23:
EE48 EC	2820	IN AL,DX ; GET STATUS
EE49 A840	2821	TEST AL,040H ; TEST DIRECTION BIT
EE4B 740C	2822	JZ J25 ; DIRECTION OK
EE4D E2F9	2823	LOOP J23
EE4F	2824	J24: ; TIME_ERROR
EE4F 800E410080	2825	OR DISKETTE_STATUS,TIME_OUT
EE54 59	2826	POP CX
EE55 5A	2827	POP DX ; SET ERROR CODE AND RESTORE REGS
EE56 58	2828	POP AX ; DISCARD THE RETURN ADDRESS
EE57 F9	2829	STC ; INDICATE ERROR TO CALLER
EE58 C3	2830	RET
EE59	2831	J25:
EE59 33C9	2832	XOR CX,CX ; RESET THE COUNT
EE5B	2833	J26:
EE5B EC	2834	IN AL,DX ; GET THE STATUS
EE5C A880	2835	TEST AL,080H ; IS IT READY
EE5E 7504	2836	JNZ J27 ; YES, GO OUTPUT
EE60 E2F9	2837	LOOP J26 ; COUNT DOWN AND TRY AGAIN
EE62 EBE8	2838	JMP J24 ; ERROR CONDITION
EE64	2839	J27: ; OUTPUT
EE64 8AC4	2840	MOV AL,AH ; GET BYTE TO OUTPUT
EE66 B2F5	2841	MOV DL,0F5H ; DATA PORT (3F5)
EE68 EE	2842	OUT DX,AL ; OUTPUT THE BYTE
EE69 59	2843	POP CX ; RECOVER REGISTERS
EE6A 5A	2844	POP DX
EE6B C3	2845	RET ; CY = 0 FROM TEST INSTRUCTION
	2846	NEC_OUTPUT ENDP
	2847	-----
	2848	; GET_PARM
	2849	; THIS ROUTINE FETCHES THE INDEXED POINTER FROM THE DISK_BASE
	2850	; BLOCK POINTED AT BY THE DATA VARIABLE DISK_POINTER. A BYTE FROM
	2851	; THAT TABLE IS THEN MOVED INTO AH, THE INDEX OF THAT BYTE BEING
	2852	; THE PARM IN BX
	2853	; ENTRY --
	2854	; BX = INDEX OF BYTE TO BE FETCHED * 2
	2855	; IF THE LOW BIT OF BX IS ON, THE BYTE IS IMMEDIATELY OUTPUT
	2856	; TO THE NEC CONTROLLER
	2857	; EXIT --
	2858	; AH = THAT BYTE FROM BLOCK
	2859	-----
EE6C	2860	GET_PARM PROC NEAR
EE6C 1E	2861	PUSH DS ; SAVE SEGMENT
EE6D 2BC0	2862	SUB AX,AX ; ZERO TO AX
EE6F 8ED8	2863	MOV DS,AX
	2864	ASSUME DS:ABS0
EE71 C5367800	2865	LDS SI,DISK_POINTER ; POINT TO BLOCK
EE75 D1EB	2866	SHR BX,1 ; DIVIDE BX BY 2, AND SET FLAG
	2867	; FOR EXIT
EE77 8A20	2868	MOV AH,[SI+BX] ; GET THE WORD

LOC OBJ	LINE	SOURCE
EE79 1F	2869	POP DS ; RESTORE SEGMENT
	2870	ASSUME DS:DATA
EE7A 72C5	2871	JC NEC_OUTPUT ; IF FLAG SET, OUTPUT TO CONTROLLER
EE7C C3	2872	RET ; RETURN TO CALLER
	2873	GET_PARM ENDP
	2874	-----
	2875	; SEEK :
	2876	; THIS ROUTINE WILL MOVE THE HEAD ON THE NAMED DRIVE TO THE :
	2877	; NAMED TRACK. IF THE DRIVE HAS NOT BEEN ACCESSED SINCE THE :
	2878	; DRIVE RESET COMMAND WAS ISSUED, THE DRIVE WILL BE RECALIBRATED. :
	2879	; INPUT :
	2880	; (DL) = DRIVE TO SEEK ON :
	2881	; (CH) = TRACK TO SEEK TO :
	2882	; OUTPUT :
	2883	; CY = 0 SUCCESS :
	2884	; CY = 1 FAILURE -- DISKETTE_STATUS SET ACCORDINGLY :
	2885	; (AX) DESTROYED :
	2886	-----
EE7D	2887	SEEK PROC NEAR
EE7D B001	2888	MOV AL,1 ; ESTABLISH MASK FOR RECAL TEST
EE7F 51	2889	PUSH CX ; SAVE INPUT VALUES
EE80 8ACA	2890	MOV CL,DL ; GET DRIVE VALUE INTO CL
EE82 D2C0	2891	ROL AL,CL ; SHIFT IT BY THE DRIVE VALUE
EE84 59	2892	POP CX ; RECOVER TRACK VALUE
EE85 84063E00	2893	TEST AL,SEEK_STATUS ; TEST FOR RECAL REQUIRED
EE89 7513	2894	JNZ J28 ; NO_RECAL
EE8B 0B063E00	2895	OR SEEK_STATUS,AL ; TURN ON THE NO RECAL BIT IN FLAG
EE8F B407	2896	MOV AH,07H ; RECALIBRATE COMMAND
EE91 E8ADFF	2897	CALL NEC_OUTPUT
EE94 8AE2	2898	MOV AH,DL
EE96 E8A8FF	2899	CALL NEC_OUTPUT ; OUTPUT THE DRIVE NUMBER
EE99 E87600	2900	CALL CHK_STAT_2 ; GET THE INTERRUPT AND SENSE INT STATUS
EE9C 7229	2901	JC J32 ; SEEK_ERROR
	2902	
	2903	;----- DRIVE IS IN SYNCH WITH CONTROLLER, SEEK TO TRACK
	2904	
EE9E	2905	J28:
EE9E B40F	2906	MOV AH,0FH ; SEEK COMMAND TO NEC
EEA0 E89EFF	2907	CALL NEC_OUTPUT
EEA3 8AE2	2908	MOV AH,DL ; DRIVE NUMBER
EEA5 E899FF	2909	CALL NEC_OUTPUT
EEA8 8AE5	2910	MOV AH,CH ; TRACK NUMBER
EEAA E894FF	2911	CALL NEC_OUTPUT
EEAD E86200	2912	CALL CHK_STAT_2 ; GET ENDING INTERRUPT AND
	2913	; SENSE STATUS
	2914	
	2915	;----- WAIT FOR HEAD SETTLE
	2916	
EEB0 9C	2917	PUSHF ; SAVE STATUS FLAGS
EEB1 BB1200	2918	MOV BX,18 ; GET HEAD SETTLE PARAMETER
EEB4 E0B5FF	2919	CALL GET_PARM
EEB7 51	2920	PUSH CX ; SAVE REGISTER
EEB8	2921	J29: ; HEAD_SETTLE
EEB8 B92602	2922	MOV CX,550 ; 1 MS LOOP
EEBB 0AE4	2923	OR AH,AH ; TEST FOR TIME EXPIRED
EEBD 7406	2924	JZ J31
EEBF	2925	J30:
EEBF E2FE	2926	LOOP J30 ; DELAY FOR 1 MS
EEC1 FECC	2927	DEC AH ; DECREMENT THE COUNT
EEC3 EBF3	2928	JMP J29 ; DO IT SOME MORE
EEC5	2929	J31:
EEC5 59	2930	POP CX ; RECOVER STATE
EEC6 90	2931	POPF
EEC7	2932	J32:
EEC7 C3	2933	RET ; RETURN TO CALLER
	2934	SEEK ENDP
	2935	-----
	2936	; DMA_SETUP :
	2937	; THIS ROUTINE SETS UP THE DMA FOR READ/WRITE/VERIFY OPERATIONS. :
	2938	; INPUT :
	2939	; (AL) = MODE BYTE FOR THE DMA :
	2940	; (ES:BX) - ADDRESS TO READ/WRITE THE DATA :
	2941	; OUTPUT :
	2942	; (AX) DESTROYED :
	2943	-----
EEC8	2944	DMA_SETUP PROC NEAR
EEC8 51	2945	PUSH CX ; SAVE THE REGISTER

LOC OBJ	LINE	SOURCE	
EEC9 FA	2946	CLI	; NO MORE INTERRUPTS
EECA E60C	2947	OUT DMA+12,AL	; SET THE FIRST/LAST F/F
EECC 50	2948	PUSH AX	
EECD 58	2949	POP AX	
EECE E60B	2950	OUT DMA+11,AL	; OUTPUT THE MODE BYTE
EED0 8CC0	2951	MOV AX,ES	; GET THE ES VALUE
EED2 B104	2952	MOV CL,4	; SHIFT COUNT
EED4 D3C0	2953	ROL AX,CL	; ROTATE LEFT
EED6 8AE8	2954	MOV CH,AL	; GET HIGHEST NYBBLE OF ES TO CH
EED8 24F0	2955	AND AL,0F0H	; ZERO THE LOW NYBBLE FROM SEGMENT
EEDA 03C3	2956	ADD AX,BX	; TEST FOR CARRY FROM ADDITION
EEDC 7302	2957	JNC J33	
EEDF FEC5	2958	INC CH	; CARRY MEANS HIGH 4 BITS MUST BE INC
EEEO	2959	J33:	
EEEO 50	2960	PUSH AX	; SAVE START ADDRESS
EEF1 E604	2961	OUT DMA+4,AL	; OUTPUT LOW ADDRESS
EEF3 8AC4	2962	MOV AL,AH	
EEF5 E604	2963	OUT DMA+4,AL	; OUTPUT HIGH ADDRESS
EEF7 8AC5	2964	MOV AL,CH	; GET HIGH 4 BITS
EEF9 240F	2965	AND AL,0FH	
EEFB E681	2966	OUT 081H,AL	; OUTPUT THE HIGH 4 BITS TO
	2967		; THE PAGE REGISTER
	2968		
	2969	i----- DETERMINE COUNT	
	2970		
EEDD 8AE6	2971	MOV AH,DH	; NUMBER OF SECTORS
EEEF 2AC0	2972	SUB AL,AL	; TIMES 256 INTO AX
EEF1 D1E8	2973	SHR AX,1	; SECTORS * 128 INTO AX
EEF3 50	2974	PUSH AX	
EEF4 B0600	2975	MOV BX,6	; GET THE BYTES/SECTOR RARM
EEF7 E872FF	2976	CALL GET_PARM	
EEFA 8ACC	2977	MOV CL,AH	; USE AS SHIFT COUNT (0=128, 1=256 ETC)
EEFC 58	2978	POP AX	
EEFD D3E0	2979	SHL AX,CL	; MULTIPLY BY CORRECT AMOUNT
EEFF 48	2980	DEC AX	; -1 FOR DMA VALUE
EF00 50	2981	PUSH AX	; SAVE COUNT VALUE
EF01 E605	2982	OUT DMA+5,AL	; LOW BYTE OF COUNT
EF03 8AC4	2983	MOV AL,AH	
EF05 E605	2984	OUT DMA+5,AL	; HIGH BYTE OF COUNT
EF07 FB	2985	STI	; INTERRUPTS BACK ON
EF08 59	2986	POP CX	; RECOVER COUNT VALUE
EF09 58	2987	POP AX	; RECOVER ADDRESS VALUE
EF0A 03C1	2988	ADD AX,CX	; ADD, TEST FOR 64K OVERFLOW
EF0C 59	2989	POP CX	; RECOVER REGISTER
EF0D B002	2990	MOV AL,2	; MODE FOR 8237
EF0F E60A	2991	OUT DMA+10,AL	; INITIALIZE THE DISKETTE CHANNEL
EF11 C3	2992	RET	; RETURN TO CALLER,
	2993		; CFL SET BY ABOVE IF ERROR
	2994	DMA_SETUP	ENDP
	2995	i-----	
	2996	; CHK_STAT_2	:
	2997	; THIS ROUTINE HANDLES THE INTERRUPT RECEIVED AFTER A	:
	2998	; RECALIBRATE, SEEK, OR RESET TO THE ADAPTER.	:
	2999	; THE INTERRUPT IS WAITED FOR, THE INTERRUPT STATUS SENSED,	:
	3000	; AND THE RESULT RETURNED TO THE CALLER.	:
	3001	; INPUT	:
	3002	; NONE	:
	3003	; OUTPUT	:
	3004	; CY = 0 SUCCESS	:
	3005	; CY = 1 FAILURE -- ERROR IS IN DISKETTE_STATUS	:
	3006	; (AX) DESTROYED	:
	3007	i-----	
EF12	3008	CHK_STAT_2	PROC NEAR
EF12 E81E00	3009	CALL WAIT_INT	; WAIT FOR THE INTERRUPT
EF15 7214	3010	JC J34	; IF ERROR, RETURN IT
EF17 B408	3011	MOV AH,08H	; SENSE INTERRUPT STATUS COMMAND
EF19 E825FF	3012	CALL NEC_OUTPUT	
EF1C E84A00	3013	CALL RESULTS	; READ IN THE RESULTS
EF1F 720A	3014	JC J34	; CHK2_RETURN
EF21 A04200	3015	MOV AL,NEC_STATUS	; GET THE FIRST STATUS BYTE
EF24 2460	3016	AND AL,060H	; ISOLATE THE BITS
EF26 3C60	3017	CHP AL,060H	; TEST FOR CORRECT VALUE
EF28 7402	3018	JZ J35	; IF ERROR, GO MARK IT
EF2A F8	3019	CLC	; GOOD RETURN
EF2B	3020	J34:	
EF2B C3	3021	RET	; RETURN TO CALLER
EF2C	3022	J35:	; CHK2_ERROR

LOC OBJ

LINE SOURCE

```

EF2C 800E410040      3023      OR      DISKETTE_STATUS,BAD_SEEK
EF31 F9              3024      STC                      ; ERROR RETURN CODE
EF32 C3              3025      RET
3026      CHK_STAT_2      ENDP
3027      ;-----
3028      ; WAIT_INT
3029      ;      THIS ROUTINE WAITS FOR AN INTERRUPT TO OCCUR. A TIME OUT
3030      ;      ROUTINE TAKES PLACE DURING THE WAIT, SO THAT AN ERROR MAY BE
3031      ;      RETURNED IF THE DRIVE IS NOT READY.
3032      ; INPUT
3033      ;      NONE
3034      ; OUTPUT
3035      ;      CY = 0 SUCCESS
3036      ;      CY = 1 FAILURE -- DISKETTE_STATUS IS SET ACCORDINGLY
3037      ;      (AX) DESTROYED
3038      ;-----
EF33              3039      WAIT_INT      PROC      NEAR
EF33 FB              3040      STI                      ; TURN ON INTERRUPTS, JUST IN CASE
EF34 53              3041      PUSH     BX
EF35 51              3042      PUSH     CX      ; SAVE REGISTERS
EF36 B302            3043      MOV     BL,2      ; CLEAR THE COUNTERS
EF38 33C9            3044      XOR     CX,CX      ; FOR 2 SECOND WAIT
EF3A              3045      J36:
EF3A F6063E0080      3046      TEST    SEEK_STATUS,INT_FLAG      ; TEST FOR INTERRUPT OCCURRING
EF3F 750C            3047      JNZ     J37
EF41 E2F7            3048      LOOP    J36      ; COUNT DOWN WHILE WAITING
EF43 FECD            3049      DEC     BL      ; SECOND LEVEL COUNTER
EF45 75F3            3050      JNZ     J36
EF47 800E410080      3051      OR     DISKETTE_STATUS,TIME_OUT      ; NOTHING HAPPENED
EF4C F9              3052      STC                      ; ERROR RETURN
EF4D              3053      J37:
EF4D 9C              3054      PUSHF     ; SAVE CURRENT CARRY
EF4E 80263E007F      3055      AND     SEEK_STATUS,NOT INT_FLAG      ; TURN OFF INTERRUPT FLAG
EF53 90              3056      POPF     ; RECOVER CARRY
EF54 59              3057      POP     CX
EF55 5B              3058      POP     BX      ; RECOVER REGISTERS
EF56 C3              3059      RET                      ; GOOD RETURN CODE COMES
3060                      ; FROM TEST INST
3061      WAIT_INT      ENDP
3062      ;-----
3063      ; DISK_INT
3064      ;      THIS ROUTINE HANDLES THE DISKETTE INTERRUPT
3065      ; INPUT
3066      ;      NONE
3067      ; OUTPUT
3068      ;      THE INTERRUPT FLAG IS SET IS SEEK_STATUS
3069      ;-----
EF57              3070      ORG     0EF57H
EF57              3071      DISK_INT      PROC      FAR
EF57 FB              3072      STI                      ; RE ENABLE INTERRUPTS
EF58 1E              3073      PUSH     DS
EF59 50              3074      PUSH     AX
EF5A E8E10F          3075      CALL    DDS
EF5D 800E3E0080      3076      OR     SEEK_STATUS,INT_FLAG
EF62 B020            3077      MOV     AL,20H      ; END OF INTERRUPT MARKER
EF64 E620            3078      OUT     20H,AL      ; INTERRUPT CONTROL PORT
EF66 5B              3079      POP     AX
EF67 1F              3080      POP     DS      ; RECOVER SYSTEM
EF68 CF              3081      IRET     ; RETURN FROM INTERRUPT
3082      DISK_INT      ENDP
3083      ;-----
3084      ; RESULTS
3085      ;      THIS ROUTINE WILL READ ANYTHING THAT THE NEC CONTROLLER HAS
3086      ;      TO SAY FOLLOWING AN INTERRUPT.
3087      ; INPUT
3088      ;      NONE
3089      ; OUTPUT
3090      ;      CY = 0 SUCCESSFUL TRANSFER
3091      ;      CY = 1 FAILURE -- TIME OUT IN WAITING FOR STATUS
3092      ;      NEC_STATUS AREA HAS STATUS BYTE LOADED INTO IT
3093      ;      (AH) DESTROYED
3094      ;-----
EF69              3095      RESULTS      PROC      NEAR
EF69 FC              3096      CLD
EF6A BF4200          3097      MOV     DI,OFFSET NEC_STATUS      ; POINTER TO DATA AREA
EF6D 51              3098      PUSH     CX      ; SAVE COUNTER
EF6E 52              3099      PUSH     DX

```

LOC OBJ	LINE	SOURCE
EF6F 53	3100	PUSH BX
EF70 B307	3101	MOV BL,7 ; MAX STATUS BYTES
	3102	
	3103	;----- WAIT FOR REQUEST FOR MASTER
	3104	
EF72	3105	J38: ; INPUT_LOOP
EF72 33C9	3106	XOR CX,CX ; COUNTER
EF74 BAF403	3107	MOV DX,03F4H ; STATUS PORT
EF77	3108	J39: ; WAIT FOR MASTER
EF77 EC	3109	IN AL,DX ; GET STATUS
EF78 A880	3110	TEST AL,080H ; MASTER READY
EF7A 750C	3111	JNZ J40A ; TEST_DIR
EF7C E2F9	3112	LOOP J39 ; WAIT_MASTER
EF7E 800E410080	3113	OR DISKETTE_STATUS,TIME_OUT
EF83	3114	J40: ; RESULTS_ERROR
EF83 F9	3115	STC ; SET ERROR RETURN
EF84 5B	3116	POP BX
EF85 5A	3117	POP DX
EF86 59	3118	POP CX
EF87 C3	3119	RET
	3120	
	3121	;----- TEST THE DIRECTION BIT
	3122	
EF88	3123	J40A: ;
EF88 EC	3124	IN AL,DX ; GET STATUS REG AGAIN
EF89 A840	3125	TEST AL,040H ; TEST DIRECTION BIT
EF8B 7507	3126	JNZ J42 ; OK TO READ STATUS
EF8D	3127	J41: ; NEC_FAIL
EF8D 800E410020	3128	OR DISKETTE_STATUS,BAD_NEC
EF92 EBEF	3129	JMP J40 ; RESULTS_ERROR
	3130	
	3131	;----- READ IN THE STATUS
	3132	
EF94	3133	J42: ; INPUT_STAT
EF94 42	3134	INC DX ; POINT AT DATA PORT
EF95 EC	3135	IN AL,DX ; GET THE DATA
EF96 8805	3136	MOV [DI],AL ; STORE THE BYTE
EF98 47	3137	INC DI ; INCREMENT THE POINTER
EF99 B90A00	3138	MOV CX,10 ; LOOP TO KILL TIME FOR NEC
EF9C E2FE	3139	J43: LOOP J43
EF9E 4A	3140	DEC DX ; POINT AT STATUS PORT
EF9F EC	3141	IN AL,DX ; GET STATUS
EFA0 A810	3142	TEST AL,010H ; TEST FOR NEC STILL BUSY
EFA2 7406	3143	JZ J44 ; RESULTS DONE
EFA4 FECB	3144	DEC BL ; DECREMENT THE STATUS COUNTER
EFA6 75CA	3145	JNZ J38 ; GO BACK FOR MORE
EFA8 EBE3	3146	JMP J41 ; CHIP HAS FAILED
	3147	
	3148	;----- RESULT OPERATION IS DONE
	3149	
EFAA	3150	J44: ;
EFAA 5B	3151	POP BX
EFA8 5A	3152	POP DX
EFA8 59	3153	POP CX ; RECOVER REGISTERS
EFA8 C3	3154	RET ; GOOD RETURN CODE FROM TEST INST
	3155	;
	3156	;-----
	3157	NUM_TRANS ;
	3158	THIS ROUTINE CALCULATES THE NUMBER OF SECTORS THAT ;
	3159	WERE ACTUALLY TRANSFERRED TO/FROM THE DISKETTE ;
	3160	INPUT ;
	3161	(CH) = CYLINDER OF OPERATION ;
	3162	(CL) = START SECTOR OF OPERATION ;
	3163	OUTPUT ;
	3164	(AL) = NUMBER ACTUALLY TRANSFERRED ;
	3165	NO OTHER REGISTERS MODIFIED ;
	3166	;
EFAE	3166	NUM_TRANS PROC NEAR
EFAE A04500	3167	MOV AL,NEC_STATUS+3 ; GET CYLINDER ENDED UP ON
EFB1 3AC5	3168	CHP AL,CH ; SAME AS WE STARTED
EFB3 A04700	3169	MOV AL,NEC_STATUS+5 ; GET ENDING SECTOR
EFB6 740A	3170	JZ J45 ; IF ON SAME CYL, THEN NO ADJUST
EFB8 BB0800	3171	MOV BX,8
EFBB E8AEFE	3172	CALL GET_PARH ; GET EOT VALUE
EFBE 8AC4	3173	MOV AL,AH ; INTO AL
EFCE FEC0	3174	INC AL ; USE EOT+1 FOR CALCULATION
EFCE	3175	J45: ;
EFCE 2AC1	3176	SUB AL,CL ; SUBTRACT START FROM END

LOC OBJ	LINE	SOURCE
EFC4 C3	3177	RET
	3178	NUM_TRANS ENDP
	3179	RESULTS ENDP
	3180	;
	3181	DISK_BASE
	3182	THIS IS THE SET OF PARAMETERS REQUIRED FOR DISKETTE OPERATION.
	3183	THEY ARE POINTED AT BY THE DATA VARIABLE DISK_POINTER. TO
	3184	MODIFY THE PARAMETERS, BUILD ANOTHER PARAMETER BLOCK AND POINT
	3185	DISK_POINTER TO IT.
	3186	;
EFC7	3187	ORG 0EFC7H
EFC7	3188	DISK_BASE LABEL BYTE
EFC7 CF	3189	DB 11001111B ; SRT=C, HD UNLOAD-OF - 1ST SPECIFY BYTE
EFC8 02	3190	DB 2 ; HD LOAD=1, MODE=DMA - 2ND SPFCIFY RYTF
EFC9 25	3191	DB MOTOR_WAIT ; WAIT AFTER OPN TIL MOTOR OFF
EFCA 02	3192	DB 2 ; 512 BYTES/SECTOR
EFCE 08	3193	DB 8 ; EOT (LAST SECTOR ON TRACK)
EFCC 2A	3194	DB 02AH ; GAP LENGTH
EFCD FF	3195	DB 0FFH ; DTL
EFCE 50	3196	DB 050H ; GAP LENGTH FOR FORMAT
EFCE F6	3197	DB 0F6H ; FILL BYTE FOR FORMAT
EFDD 19	3198	DB 25 ; HEAD SETTLE TIME (MILLISECONDS)
EFDD 04	3199	DB 4 ; MOTOR START TIME (1/8 SECONDS)
	3200	;
	3201	INT 17
	3202	PRINTER_IO
	3203	THIS ROUTINE PROVIDES COMMUNICATION WITH THE PRINTER
	3204	INPUT
	3205	(AH)=0 PRINT THE CHARACTER IN (AL)
	3206	ON RETURN, AH=1 IF CHARACTER COULD NOT BE PRINTED
	3207	(TIME OUT). OTHER BITS SET AS ON NORMAL STATUS CALL
	3208	(AH)=1 INITIALIZE THE PRINTER PORT
	3209	RETURNS WITH (AH) SET WITH PRINTER STATUS
	3210	(AH)=2 READ THE PRINTER STATUS INTO (AH)
	3211	7 6 5 4 3 2-1 0
	3212	TIME OUT
	3213	UNUSED
	3214	1 = I/O ERROR
	3215	1 = SELECTED
	3216	1 = OUT OF PAPER
	3217	1 = ACKNOWLEDGE
	3218	1 = NOT BUSY
	3219	;
	3220	(DX) = PRINTER TO BE USED (0,1,2) CORRESPONDING TO ACTUAL
	3221	VALUES IN PRINTER_BASE AREA
	3222	;
	3223	DATA AREA PRINTER_BASE CONTAINS THE BASE ADDRESS OF THE PRINTER
	3224	CARD(S) AVAILABLE (LOCATED AT BEGINNING OF DATA SEGMENT,
	3225	408H ABSOLUTE, 3 WORDS)
	3226	;
	3227	DATA AREA PRINT_TIM_OUT (BYTE) MAY BE CHANGED TO CAUSE DIFFERENT
	3228	TIME-OUT WAITS. DEFAULT=20
	3229	;
	3230	REGISTERS AH IS MODIFIED
	3231	ALL OTHERS UNCHANGED
	3232	;
	3233	ASSUME CS:CODE,DS:DATA
EFDD	3234	ORG 0EFD2H
EFDD	3235	PRINTER_IO PROC FAR
EFDD FB	3236	STI ; INTERRUPTS BACK ON
EFDD 1E	3237	PUSH DS ; SAVE SEGMENT
EFDD 52	3238	PUSH DX
EFDD 56	3239	PUSH SI
EFDD 51	3240	PUSH CX
EFDD 53	3241	PUSH BX
EFDD E6630F	3242	CALL DOS
EFDD 8BF2	3243	MOV SI,DX ; GET PRINTER PARM
EFDD 8A5C78	3244	MOV BL,PRINT_TIM_OUT[SI] ; LOAD TIME-OUT PARM
EFDD D1E6	3245	SHL SI,1 ; WORD OFFSET INTO TABLE
EFDD 8B5408	3246	MOV DX,PRINTER_BASE[SI] ; GET BASE ADDRESS FOR PRINTER CARD
EFDD 8BD2	3247	OR DX,DX ; TEST DX FOR ZERO,
	3248	INDICATING NO PRINTER
EFDD 740C	3249	JZ B1 ; RETURN
EFDD 0AE4	3250	OR AH,AH ; TEST FOR (AH)=0
EFDD 740E	3251	JZ B2 ; PRINT_AL
EFDD FECC	3252	DEC AH ; TEST FOR (AH)=1
EFDD 743F	3253	JZ B8 ; INIT_PRT

LOC OBJ	LINE	SOURCE	
EFF1 FECC	3254	DEC AH	; TEST FOR (AH)=2
EFF3 7428	3255	JZ B5	; PRINTER STATUS
EFF5	3256	B1:	; RETURN
EFF5 5B	3257	POP BX	
EFF6 59	3258	POP CX	
EFF7 5E	3259	POP SI	; RECOVER REGISTERS
EFF8 5A	3260	POP DX	; RECOVER REGISTERS
EFF9 1F	3261	POP DS	
EFFA CF	3262	IRET	
	3263		
	3264	;----- PRINT THE CHARACTER IN (AL)	
	3265		
EFFB	3266	B2:	
EFFB 50	3267	PUSH AX	; SAVE VALUE TO PRINT
EFFC EE	3268	OUT DX,AL	; OUTPUT CHAR TO PORT
EFFD 42	3269	INC DX	; POINT TO STATUS PORT
EFFE	3270	B3:	
EFFE 2BC9	3271	SUB CX,CX	; WAIT_BUSY
F000	3272	B3_1:	
F000 EC	3273	IN AL,DX	; GET STATUS
F001 8AE0	3274	MOV AH,AL	; STATUS TO AH ALSO
F003 A800	3275	TEST AL,80H	; IS THE PRINTER CURRENTLY BUSY
F005 750E	3276	JNZ B4	; OUT_STROBE
F007 E2F7	3277	LOOP B3_1	; TRY AGAIN
F009 FECB	3278	DEC BL	; DROP LOOP COUNT
F00B 75F1	3279	JNZ B3	; GO TILL TIMEOUT ENDS
F00D 80CC01	3280	OR AH,1	; SET ERROR FLAG
F010 80E4F9	3281	AND AH,0F9H	; TURN OFF THE OTHER BITS
F013 EB13	3282	JMP SHORT B7	; RETURN WITH ERROR FLAG SET
F015	3283	B4:	; OUT_STROBE
F015 B00D	3284	MOV AL,0DH	; SET THE STROBE HIGH
F017 42	3285	INC DX	; STROBE IS BIT 0 OF PORT C OF 8255
F018 EE	3286	OUT DX,AL	
F019 B00C	3287	MOV AL,0CH	; SET THE STROBE LOW
F01B EE	3288	OUT DX,AL	
F01C 58	3289	POP AX	; RECOVER THE OUTPUT CHAR
	3290		
	3291	;----- PRINTER STATUS	
	3292		
F01D	3293	B5:	
F01D 50	3294	PUSH AX	; SAVE AL REG
F01E	3295	B6:	
F01E 8B5408	3296	MOV DX,PRINTER_BASE[SI]	
F021 42	3297	INC DX	
F022 EC	3298	IN AL,DX	; GET PRINTER STATUS
F023 8AE0	3299	MOV AH,AL	
F025 80E4F8	3300	AND AH,0F8H	; TURN OFF UNUSED BITS
F028	3301	B7:	; STATUS_SET
F028 5A	3302	POP DX	; RECOVER AL REG
F029 8AC2	3303	MOV AL,DL	; GET CHARACTER INTO AL
F02B 80F448	3304	XOR AH,48H	; FLIP A COUPLE OF BITS
F02E EBC5	3305	JMP B1	; RETURN FROM ROUTINE
	3306		
	3307	;----- INITIALIZE THE PRINTER PORT	
	3308		
F030	3309	B8:	
F030 50	3310	PUSH AX	; SAVE AL
F031 42	3311	INC DX	; POINT TO OUTPUT PORT
F032 42	3312	INC DX	
F033 B008	3313	MOV AL,8	; SET INIT LINE LOW
F035 EE	3314	OUT DX,AL	
F036 B8E803	3315	MOV AX,1000	
F039	3316	B9:	; INIT_LOOP
F039 48	3317	DEC AX	; LOOP FOR RESET TO TAKE
F03A 75FD	3318	JNZ B9	; INIT_LOOP
F03C B00C	3319	MOV AL,0CH	; NO INTERRUPTS, NON AUTO LF,
	3320		; INIT HIGH
F03E EE	3321	OUT DX,AL	
F03F EB0D	3322	JMP B6	; PRT_STATUS_1
	3323	PRINTER_IO	
	3324	ENDP	
F041 62E1	3325	C2 DW C24	; RETURN ADDRESS FOR DUMMY STACK
	3326		
	3327	;--- INT 10 -----	
	3328	; VIDEO_IO	:
	3329	; THESE ROUTINES PROVIDE THE CRT INTERFACE	:
	3330	; THE FOLLOWING FUNCTIONS ARE PROVIDED:	:

```

3331 | (AH)=0 SET MODE (AL) CONTAINS MODE VALUE |
3332 | (AL)=0 40X25 BW (POWER ON DEFAULT) |
3333 | (AL)=1 40X25 COLOR |
3334 | (AL)=2 80X25 BW |
3335 | (AL)=3 80X25 COLOR |
3336 | GRAPHICS MODES |
3337 | (AL)=4 320X200 COLOR |
3338 | (AL)=5 320X200 BW |
3339 | (AL)=6 640X200 BW |
3340 | CRT MODE=7 80X25 B&W CARD (USED INTERNAL TO VIDEO ONLY) |
3341 | *** NOTE BW MODES OPERATE SAME AS COLOR MODES, BUT |
3342 | COLOR BURST IS NOT ENABLED |
3343 | (AH)=1 SET CURSOR TYPE |
3344 | (CH) = BITS 4-0 = START LINE FOR CURSOR |
3345 | ** HARDWARE WILL ALWAYS CAUSE BLIN |
3346 | ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC |
3347 | BLINKING OR NO CURSOR AT ALL |
3348 | (CL) = BITS 4-0 = END LINE FOR CURSOR |
3349 | (AH)=2 SET CURSOR POSITION |
3350 | (DH,DL) = ROW,COLUMN (0,0) IS UPPER LEFT |
3351 | (BH) = PAGE NUMBER (MUST BE 0 FOR GRAPHICS MODES) |
3352 | (AH)=3 READ CURSOR POSITION |
3353 | (BH) = PAGE NUMBER (MUST BE 0 FOR GRAPHICS MODES) |
3354 | ON EXIT (DH,DL) = ROW,COLUMN OF CURRENT CURSOR |
3355 | (CH,CL) = CURSOR MODE CURRENTLY SET |
3356 | (AH)=4 READ LIGHT PEN POSITION |
3357 | ON EXIT: |
3358 | (AH) = 0 -- LIGHT PEN SWITCH NOT DOWN/NOT TRIGGERED |
3359 | (AH) = 1 -- VALID LIGHT PEN VALUE IN REGISTERS |
3360 | (DH,DL) = ROW,COLUMN OF CHARACTER LP POSN |
3361 | (CH) = RASTER LINE (0-199) |
3362 | (BX) = PIXEL COLUMN (0-319,639) |
3363 | (AH)=5 SELECT ACTIVE DISPLAY PAGE (VALID ONLY FOR ALPHA MODES) |
3364 | (AL)=NEW PAGE VAL (0-7 FOR MODES 0&1, 0-3 FOR MODES 2&3) |
3365 | (AH)=6 SCROLL ACTIVE PAGE UP |
3366 | (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT BOTTOM |
3367 | OF WINDOW |
3368 | AL = 0 MEANS BLANK ENTIRE WINDOW |
3369 | (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL |
3370 | (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL |
3371 | (BH) = ATTRIBUTE TO BE USED ON BLANK LINE |
3372 | (AH)=7 SCROLL ACTIVE PAGE DOWN |
3373 | (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT TOP |
3374 | OF WINDOW |
3375 | AL = 0 MEANS BLANK ENTIRE WINDOW |
3376 | (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL |
3377 | (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL |
3378 | (BH) = ATTRIBUTE TO BE USED ON BLANK LINE |
3379 | |
3380 | CHARACTER HANDLING ROUTINES |
3381 | |
3382 | (AH) = 8 READ ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION |
3383 | (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY) |
3384 | ON EXIT: |
3385 | (AL) = CHAR READ |
3386 | (AH) = ATTRIBUTE OF CHARACTER READ (ALPHA MODES ONLY) |
3387 | (AH) = 9 WRITE ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION |
3388 | (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY) |
3389 | (CX) = COUNT OF CHARACTERS TO WRITE |
3390 | (AL) = CHAR TO WRITE |
3391 | (BL) = ATTRIBUTE OF CHARACTER (ALPHA)/COLOR OF CHAR |
3392 | (GRAPHICS) |
3393 | SEE NOTE ON WRITE DOT FOR BIT 7 OF BL = 1. |
3394 | (AH) = 10 WRITE CHARACTER ONLY AT CURRENT CURSOR POSITION |
3395 | (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY) |
3396 | (CX) = COUNT OF CHARACTERS TO WRITE |
3397 | (AL) = CHAR TO WRITE |
3398 | |
3399 | FOR READ/WRITE CHARACTER INTERFACE WHILE IN GRAPHICS MODE, THE |
3400 | CHARACTERS ARE FORMED FROM A CHARACTER GENERATOR IMAGE |
3401 | MAINTAINED IN THE SYSTEM ROM. ONLY THE 1ST 128 CHARS |
3402 | ARE CONTAINED THERE. TO READ/WRITE THE SECOND 128 |
3403 | CHARS, THE USER MUST INITIALIZE THE POINTER AT |
3404 | INTERRUPT 1FH (LOCATION 0007CH) TO POINT TO THE 1K BYTE |
3405 | TABLE CONTAINING THE CODE POINTS FOR THE SECOND |
3406 | 128 CHARS (128-255). |
3407 | |
3408 | FOR WRITE CHARACTER INTERFACE IN GRAPHICS MODE, THE REPLICATION |
3409 | FACTOR CONTAINED IN (CX) ON ENTRY WILL PRODUCE VALID |

```

```

3408 ; RESULTS ONLY FOR CHARACTERS CONTAINED ON THE SAME ROW. ;
3409 ; CONTINUATION TO SUCCEEDING LINES WILL NOT PRODUCE ;
3410 ; CORRECTLY. ;
3411 ; ;
3412 ; GRAPHICS INTERFACE ;
3413 ; (AH) = 11 SET COLOR PALETTE ;
3414 ; (BH) = PALETTE COLOR ID BEING SET (0-127) ;
3415 ; (BL) = COLOR VALUE TO BE USED WITH THAT COLOR ID ;
3416 ; NOTE: FOR THE CURRENT COLOR CARD, THIS ENTRY POINT ;
3417 ; HAS MEANING ONLY FOR 320X200 GRAPHICS. ;
3418 ; COLOR ID = 0 SELECTS THE BACKGROUND COLOR (0-15); ;
3419 ; COLOR ID = 1 SELECTS THE PALETTE TO BE USED: ;
3420 ; 0 = GREEN(1)/RED(2)/YELLOW(3) ;
3421 ; 1 = CYAN(1)/MAGENTA(2)/WHITE(3) ;
3422 ; IN 40X25 OR 80X25 ALPHA MODES, THE VALUE SET ;
3423 ; FOR PALETTE COLOR 0 INDICATES THE ;
3424 ; BORDER COLOR TO BE USED (VALUES 0-31, ;
3425 ; WHERE 16-31 SELECT THE HIGH INTENSITY ;
3426 ; BACKGROUND SET. ;
3427 ; (AH) = 12 WRITE DOT ;
3428 ; (DX) = ROW NUMBER ;
3429 ; (CX) = COLUMN NUMBER ;
3430 ; (AL) = COLOR VALUE ;
3431 ; IF BIT 7 OF AL = 1, THEN THE COLOR VALUE IS ;
3432 ; EXCLUSIVE OR'D WITH THE CURRENT CONTENTS OF ;
3433 ; THE DOT ;
3434 ; (AH) = 13 READ DOT ;
3435 ; (DX) = ROW NUMBER ;
3436 ; (CX) = COLUMN NUMBER ;
3437 ; (AL) RETURNS THE DOT READ ;
3438 ; ;
3439 ; ASCII TELETYPE ROUTINE FOR OUTPUT ;
3440 ; ;
3441 ; (AH) = 14 WRITE TELETYPE TO ACTIVE PAGE ;
3442 ; (AL) = CHAR TO WRITE ;
3443 ; (BL) = FOREGROUND COLOR IN GRAPHICS MODE ;
3444 ; NOTE -- SCREEN WIDTH IS CONTROLLED BY PREVIOUS MODE SET ;
3445 ; ;
3446 ; (AH) = 15 CURRENT VIDEO STATE ;
3447 ; RETURNS THE CURRENT VIDEO STATE ;
3448 ; (AL) = MODE CURRENTLY SET ( SEE AH=0 FOR EXPLANATION) ;
3449 ; (AH) = NUMBER OF CHARACTER COLUMNS ON SCREEN ;
3450 ; (BH) = CURRENT ACTIVE DISPLAY PAGE ;
3451 ; ;
3452 ; CS,SS,DS,ES,BX,CX,DX PRESERVED DURING CALL ;
3453 ; ALL OTHERS DESTROYED ;
3454 ; -----
3455 ; ASSUME CS:CODE,DS:DATA,ES:VIDEO_RAM
3456 ; ORG 0F045H
3457 MI LABEL WORD ; TABLE OF ROUTINES WITHIN VIDEO I/O
3458 DW OFFSET SET_MODE
3459 DW OFFSET SET_CTYPE
3460 DW OFFSET SET_CPOS
3461 DW OFFSET READ_CURSOR
3462 DW OFFSET READ_LPEN
3463 DW OFFSET ACT_DISP_PAGE
3464 DW OFFSET SCROLL_UP
3465 DW OFFSET SCROLL_DOWN
3466 DW OFFSET READ_AC_CURRENT
3467 DW OFFSET WRITE_AC_CURRENT
3468 DW OFFSET WRITE_C_CURRENT
3469 DW OFFSET SET_COLOR
3470 DW OFFSET WRITE_DOT
3471 DW OFFSET READ_DOT
3472 DW OFFSET WRITE_TTY
3473 DW OFFSET VIDEO_STATE
3474 M1L EQU $-M1
3475
3476 ; ORG 0F065H
3477 VIDEO_IO PROC NEAR
3478 STI ; INTERRUPTS BACK ON
3479 CLD ; SET DIRECTION FORWARD
3480 PUSH ES
3481 PUSH DS ; SAVE SEGMENT REGISTERS
3482 PUSH DX
3483 PUSH CX
3484 PUSH BX

```


LOC	OBJ	LINE	SOURCE
F06C	56	3485	PUSH SI
F06D	57	3486	PUSH DI
F06E	50	3487	PUSH AX
F06F	8AC4	3488	MOV AL, AH
F071	32E4	3489	XOR AH, AH
F073	D1E0	3490	SAL AX, 1
F075	8BF0	3491	MOV SI, AX
F077	3D2000	3492	CMP AX, M1L
F07A	7204	3493	JB M2
F07C	58	3494	POP AX
F07D	E94501	3495	JMP VIDEO_RETURN
F080		3496	M2: CALL DDS
F080	E0B0E	3497	MOV AX, 0B800H
F083	B800B8	3498	MOV DI, EQUIP_FLAG
F086	8B3E1000	3499	AND DI, 30H
F08A	81E73000	3500	CMP DI, 30H
F08E	83FF30	3501	JNE M3
F091	7502	3502	MOV AH, 0B0H
F093	B4B0	3503	M3: MOV ES, AX
F095		3504	POP AX
F095	8EC0	3505	MOV AH, CRT_MODE
F097	58	3506	JMP WORD PTR CS:[SI+OFFSET M1]
F098	8A264900	3507	VIDEO_TO
F09C	2EFA445F0	3508	ENDP
		3509	-----
		3510	SET_MODE
		3511	THIS ROUTINE INITIALIZES THE ATTACHMENT TO
		3512	THE SELECTED MODE. THE SCREEN IS BLANKED.
		3513	INPUT
		3514	(AL) = MODE SELECTED (RANGE 0-9)
		3515	OUTPUT
		3516	NONE
		3517	-----
		3518	TABLES FOR USE IN SETTING OF MODE
		3519	
F0A4		3520	ORG 0F0A4H
F0A4		3521	VIDEO_PARAMS LABEL BYTE
		3522	INIT_TABLE
F0A4 38		3523	DB 38H, 28H, 20H, 0AH, 1FH, 6, 19H
F0A5 28		3524	
F0A6 2D		3525	
F0A7 0A			
F0A8 1F			
F0A9 06			
F0AA 19			
F0AB 1C	3526	DB	1CH, 2, 7, 6, 7
F0AC 02			
F0AD 07			
F0AE 06			
F0AF 07			
F0B0 00	3527	DB	0, 0, 0, 0
F0B1 00			
F0B2 00			
F0B3 00			
0010	3528	M4 EQU	9-VIDEO_PARAMS
	3529		
F0B4 71	3530	DB	71H, 50H, 5AH, 0AH, 1FH, 6, 19H
F0B5 50			
F0B6 5A			
F0B7 0A			
F0B8 1F			
F0B9 06			
F0BA 19			
F0BB 1C	3531	DB	1CH, 2, 7, 6, 7
F0BC 02			
F0BD 07			
F0BE 06			
F0BF 07			
F0C0 00	3532	DB	0, 0, 0, 0
F0C1 00			
F0C2 00			
F0C3 00			
	3533		
F0C4 38	3534	DB	38H, 28H, 20H, 0AH, 7FH, 6, 64H
F0C5 28			

LOC OBJ	LINE	SOURCE
F0C6 2D		
F0C7 0A		
F0C8 7F		
F0C9 06		
F0CA 64		
F0CB 70	3535	DB 70H,2,1,6,7
F0CC 02		
F0CD 01		
F0CE 06		
F0CF 07		
F0D0 00	3536	DB 0,0,0,0
F0D1 00		
F0D2 00		
F0D3 00		
	3537	
F0D4 61	3538	DB 61H,50H,52H,0FH,19H,6,19H ; SET UP FOR 80X25 B&W CARD
F0D5 50		
F0D6 52		
F0D7 0F		
F0D8 19		
F0D9 06		
F0DA 19		
F0DB 19	3539	DB 19H,2,0DH,0BH,0CH
F0DC 02		
F0DD 0D		
F0DE 0B		
F0DF 0C		
F0E0 00	3540	DB 0,0,0,0
F0E1 00		
F0E2 00		
F0E3 00		
	3541	
F0E4	3542	M5 LABEL WORD ; TABLE OF REGEN LENGTHS
F0E4 0008	3543	DW 2048 ; 40X25
F0E6 0010	3544	DW 4096 ; 80X25
F0E8 0040	3545	DW 16384 ; GRAPHICS
F0EA 0040	3546	DW 16384
	3547	
	3548	;----- COLUMNS
	3549	
F0EC	3550	M6 LABEL BYTE
F0EC 28	3551	DB 40,40,80,80,40,40,80,80
F0ED 28		
F0EE 50		
F0EF 50		
F0F0 28		
F0F1 28		
F0F2 50		
F0F3 50		
	3552	
	3553	;----- C_REG_TAB
	3554	
F0F4	3555	M7 LABEL BYTE ; TABLE OF MODE SETS
F0F4 2C	3556	DB 2CH,28H,2DH,29H,2AH,2EH,1EH,29H
F0F5 28		
F0F6 2D		
F0F7 29		
F0F8 2A		
F0F9 2E		
F0FA 1E		
F0FB 29		
	3557	
F0FC	3558	SET_MODE PROC NEAR
F0FC BAD403	3559	MOV DX,03D4H ; ADDRESS OF COLOR CARD
F0FF B300	3560	MOV BL,0 ; MODE SET FOR COLOR CARD
F101 83FF30	3561	CMP DI,30H ; IS BW CARD INSTALLED
F104 7506	3562	JNE M8 ; OK WITH COLOR
F106 B007	3563	MOV AL,7 ; INDICATE BW CARD MODE
F108 B2B4	3564	MOV DI,0B4H ; ADDRESS OF BW CARD (384)
F10A FEC3	3565	INC BL ; MODE SET FOR BW CARD
F10C	3566	M8:
F10C 8AE0	3567	MOV AH,AL ; SAVE MODE IN AH
F10E A24900	3568	MOV CRT_MODE,AL ; SAVE IN GLOBAL VARIABLE
F111 89166300	3569	MOV ADDR_6045,DX ; SAVE ADDRESS OF BASE
F115 1E	3570	PUSH DS ; SAVE POINTER TO DATA SEGMENT
F116 50	3571	PUSH AX ; SAVE MODE
F117 52	3572	PUSH DX ; SAVE OUTPUT PORT VALUE

LOC	OBJ	LINE	SOURCE
F118	83C204	3573	ADD DX,4 ; POINT TO CONTROL REGISTER
F118	8AC3	3574	MOV AL,BL ; GET MODE SET FOR CARD
F110	EE	3575	OUT DX,AL ; RESET VIDEO
F11E	5A	3576	POP DX ; BACK TO BASE REGISTER
F11F	2BC0	3577	SUB AX,AX ; SET UP FOR ABS0 SEGMENT
F121	8E08	3578	MOV DS,AX ; ESTABLISH VECTOR TABLE ADDRESSING
		3579	ASSUME DS:ABS0
F123	C51E7400	3580	LDS BX,PARM_PTR ; GET POINTER TO VIDEO PARMS
F127	58	3581	POP AX ; RECOVER PARMS
		3582	ASSUME DS:CODE
F128	B91000	3583	MOV CX,M4 ; LENGTH OF EACH ROW OF TABLE
F128	80FC02	3584	CMF AH,2 ; DETERMINE WHICH ONE TO USE
F12E	7210	3585	JC M9 ; MODE IS 0 OR 1
F130	0309	3586	ADD BX,CX ; MOVE TO NEXT ROW OF INIT TABLE
F132	80FC04	3587	CMF AH,4
F135	7209	3588	JC M9 ; MOVE IS 2 OR 3
F137	0309	3589	ADD BX,CX ; MOVE TO GRAPHICS ROW OF INIT_TABLE
F139	80FC07	3590	CMF AH,7
F13C	7202	3591	JC M9 ; MODE IS 4,5, OR 6
F13E	0309	3592	ADD BX,CX ; MOVE TO BW CARD ROW OF INIT_TABLE
		3593	
		3594	;----- BX POINTS TO CORRECT ROW OF INITIALIZATION TABLE
		3595	
F140		3596	M9: ; OUT_INIT
F140	50	3597	PUSH AX ; SAVE MODE IN AH
F141	32E4	3598	XOR AH,AH ; AH WILL SERVE AS REGISTER
		3599	; NUMBER DURING LOOP
		3600	
		3601	;----- LOOP THROUGH TABLE, OUTPUTTING REG ADDRESS, THEN VALUE FROM TABLE
		3602	
F143		3603	M10: ; INIT LOOP
F143	8AC4	3604	MOV AL,AH ; GET 6845 REGISTER NUMBER
F145	EE	3605	OUT DX,AL
F146	42	3606	INC DX ; POINT TO DATA PORT
F147	FEC4	3607	INC AH ; NEXT REGISTER VALUE
F149	8A07	3608	MOV AL,[BX] ; GET TABLE VALUE
F14B	EE	3609	OUT DX,AL ; OUT TO CHIP
F14C	43	3610	INC BX ; NEXT IN TABLE
F14D	4A	3611	DEC DX ; BACK TO POINTER REGISTER
F14E	E2F3	3612	LOOP M10 ; DO THE WHOLE TABLE
F150	58	3613	POP AX ; GET MODE BACK
F151	1F	3614	POP DS ; RECOVER SEGMENT VALUE
		3615	ASSUME DS:DATA
		3616	
		3617	;----- FILL REGEN AREA WITH BLANK
		3618	
F152	33FF	3619	XOR DI,DI ; SET UP POINTER FOR REGEN
F154	893E4E00	3620	MOV CRT_START,DI ; START ADDRESS SAVED IN GLOBAL
F158	C606620000	3621	MOV ACTIVE_PAGE,0 ; SET PAGE VALUE
F15D	B90020	3622	MOV CX,8192 ; NUMBER OF WORDS IN COLOR CARD
F160	80FC04	3623	CMF AH,4 ; TEST FOR GRAPHICS
F163	720B	3624	JC M12 ; NO_GRAPHICS_INIT
F165	80FC07	3625	CMF AH,7 ; TEST FOR BW CARD
F168	7404	3626	JE M11 ; BW_CARD_INIT
F16A	33C0	3627	XOR AX,AX ; FILL FOR GRAPHICS MODE
F16C	EB05	3628	JMP SHORT M13 ; CLEAR_BUFFER
F16E		3629	M11: ; BW_CARD_INIT
F16E	B508	3630	MOV CH,08H ; BUFFER SIZE ON BW CARD
F170		3631	M12: ; NO_GRAPHICS_INIT
F170	B82007	3632	MOV AX,' '*256 ; FILL CHAR FOR ALPHA
F173		3633	M13: ; CLEAR_BUFFER
F173	F3	3634	REP STOSW ; FILL THE REGEN BUFFER WITH BLANKS
F174	AB		
		3635	
		3636	;----- ENABLE VIDEO AND CORRECT PORT SETTING
		3637	
F175	C70660000706	3638	MOV CURSOR_MODE,607H ; SET CURRENT CURSOR MODE
F17B	A04900	3639	MOV AL,CRT_MODE ; GET THE MODE
F17E	32E4	3640	XOR AH,AH ; INTO AX REGISTER
F180	8B0F	3641	MOV SI,AX ; TABLE POINTER, INDEXED BY MODE
F182	8B166300	3642	MOV DX,ADDR_6845 ; PREPARE TO OUTPUT TO
		3643	; VIDEO ENABLE PORT
F186	83C204	3644	ADD DX,4
F189	2E8A84F4F0	3645	MOV AL,CS:[SI+OFFSET M7]
F18E	EE	3646	OUT DX,AL ; SET VIDEO ENABLE PORT
F18F	A26500	3647	MOV CRT_MODE_SET,AL ; SAVE THAT VALUE
		3648	

```

LOC OBJ          LINE  SOURCE

3649      ;----- DETERMINE NUMBER OF COLUMNS, BOTH FOR ENTIRE DISPLAY
3650      ;----- AND THE NUMBER TO BE USED FOR TTY INTERFACE
3651
F192 2E8A84ECF0  3652      MOV     AL,CS:[SI + OFFSET M6]
F197 32E4        3653      XOR     AH,AH
F199 A34A00      3654      MOV     CRT_COLS,AX      ; NUMBER OF COLUMNS IN THIS SCREEN
3655
3656      ;----- SET CURSOR POSITIONS
3657
F19C 81E60E00    3658      AND     SI,0EH      ; WORD OFFSET INTO CLEAR LENGTH TABLE
F1A0 2E8B8CE4F0  3659      MOV     CX,CS:[SI + OFFSET M5] ; LENGTH TO CLEAR
F1A5 890E4C00    3660      MOV     CRT_LEN,CX   ; SAVE LENGTH OF CRT -- NOT USED FOR BH
F1A9 D90800      3661      MOV     CX,8        ; CLEAR ALL CURSOR POSITIONS
F1AC BF5000      3662      MOV     DI,OFFSET CURSOR_POSN
F1AF 1E          3663      PUSH    DS          ; ESTABLISH SEGMENT
F1B0 07          3664      POP     ES         ; ADDRESSING
F1B1 33C0        3665      XOR     AX,AX
F1B3 F3          3666      REP     STOSW      ; FILL WITH ZEROES
F1B4 AB
3667
3668      ;----- SET UP OVERSCAN REGISTER
3669
F1B5 42          3670      INC     DX          ; SET OVERSCAN PORT TO A DEFAULT
F1B6 B030        3671      MOV     AL,30H     ; VALUE OF 30H FOR ALL MODES
3672      ; EXCEPT 640X200
3673      ; SEE IF THE MODE IS 640X200 BW
F1B8 803E490006  3673      CMP     CRT_MODE,6
F1BD 7502        3674      JNZ     M14      ; IF IT ISNT 640X200, THEN GOTO REGULAR
F1BF B03F        3675      MOV     AL,3FH     ; IF IT IS 640X200, THEN PUT IN 3FH
F1C1            3676      M14:
F1C1 EE          3677      OUT     DX,AL      ; OUTPUT THE CORRECT VALUE TO 3D9 PORT
F1C2 A26600      3678      MOV     CRT_PALETTE,AL ; SAVE THE VALUE FOR FUTURE USE
3679
3680      ;----- NORMAL RETURN FROM ALL VIDEO RETURNS
3681
F1C5            3682      VIDEO_RETURN:
F1C5 5F          3683      POP     DI
F1C6 5E          3684      POP     SI
F1C7 5B          3685      POP     BX
3686      M15:
F1C8 59          3687      POP     CX      ; VIDEO_RETURN_C
F1C9 5A          3688      POP     DX
F1CA 1F          3689      POP     DS
F1CB 07          3690      POP     ES      ; RECOVER SEGMENTS
F1CC CF          3691      IRET      ; ALL DONE
3692      SET_MODE      ENDP
3693      ;-----
3694      ; SET_CTYPE
3695      ; THIS ROUTINE SETS THE CURSOR VALUE
3696      ; INPUT
3697      ; (CX) HAS CURSOR VALUE CH-START LINE, CL-STOP LINE
3698      ; OUTPUT
3699      ; NONE
3700      ;-----
F1CD            3701      SET_CTYPE      PROC      NEAR
F1CD B60A        3702      MOV     AH,10      ; 6845 REGISTER FOR CURSOR SET
F1CF 890E6600    3703      MOV     CURSOR_MODE,CX ; SAVE IN DATA AREA
F1D3 E80200      3704      CALL    M16        ; OUTPUT CX REG
F1D6 EBED        3705      JMP     VIDEO_RETURN
3706
3707      ;----- THIS ROUTINE OUTPUTS THE CX REGISTER TO THE 6845 REGS NAMED IN AH
3708
F1D8            3709      M16:
F1D8 8B166300    3710      MOV     DX,ADDR_6845 ; ADDRESS REGISTER
F1DC 8AC4        3711      MOV     AL,AH      ; GET VALUE
F1DE EE          3712      OUT     DX,AL      ; REGISTER SET
F1DF 42          3713      INC     DX          ; DATA REGISTER
F1E0 8AC5        3714      MOV     AL,CH      ; DATA
F1E2 EE          3715      OUT     DX,AL
F1E3 4A          3716      DEC     DX
F1E4 8AC4        3717      MOV     AL,AH
F1E6 FEC0        3718      INC     AL          ; POINT TO OTHER DATA REGISTER
F1E8 EE          3719      OUT     DX,AL      ; SET FOR SECOND REGISTER
F1E9 42          3720      INC     DX
F1EA 8AC1        3721      MOV     AL,CL      ; SECOND DATA VALUE
F1EC EE          3722      OUT     DX,AL
F1ED C3          3723      RET              ; ALL DONE
3724      SET_CTYPE      ENDP

```

LOC OBJ	LINE	SOURCE
	3725	-----
	3726	SET_CPOS :
	3727	THIS ROUTINE SETS THE CURRENT CURSOR :
	3728	POSITION TO THE NEW X-Y VALUES PASSED :
	3729	INPUT :
	3730	DX - ROW,COLUMN OF NEW CURSOR :
	3731	BH - DISPLAY PAGE OF CURSOR :
	3732	OUTPUT :
	3733	CURSOR IS SET AT 6845 IF DISPLAY PAGE :
	3734	IS CURRENT DISPLAY :
	3735	-----
F1EE	3736	SET_CPOS PROC NEAR
F1EE 8ACF	3737	MOV CL,BH
F1F0 32ED	3738	XOR CH,CH ; ESTABLISH LOOP COUNT
F1F2 D1E1	3739	SAL CX,1 ; WORD OFFSET
F1F4 8BF1	3740	MOV SI,CX ; USE INDEX REGISTER
F1F6 895450	3741	MOV [SI+OFFSET CURSOR_POSN],DX ; SAVE THE POINTER
F1F9 383E6200	3742	CHP ACTIVE_PAGE,BH
F1FD 7505	3743	JNZ M17 ; SET_CPOS_RETURN
F1FF 8BC2	3744	MOV AX,DX ; GET ROW/COLUMN TO AX
F201 E80200	3745	CALL M18 ; CURSOR_SET
F204	3746	M17: ; SET_CPOS_RETURN
F204 EBBF	3747	JMP VIDEO_RETURN
	3748	SET_CPOS ENDP
	3749	
	3750	----- SET CURSOR POSITION, AX HAS ROW/COLUMN FOR CURSOR
	3751	
F206	3752	M18 PROC NEAR
F206 E87C00	3753	CALL POSITION ; DETERMINE LOCATION IN REGEN BUFFER
F209 8BC8	3754	MOV CX,AX
F20B 030E4E00	3755	ADD CX,CRT_START ; ADD IN THE START ADDR FOR THIS PAGE
F20F D1F9	3756	SAR CX,1 ; DIVIDE BY 2 FOR CHAR ONLY COUNT
F211 B40E	3757	MOV AH,14 ; REGISTER NUMBER FOR CURSOR
F213 E8C2FF	3758	CALL M16 ; OUTPUT THE VALUE TO THE 6845
F216 C3	3759	RET
	3760	M18 ENDP
	3761	-----
	3762	ACT_DISP_PAGE :
	3763	THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING THE :
	3764	FULL USE OF THE RAM SET ASIDE FOR THE VIDEO ATTACHMENT :
	3765	INPUT :
	3766	AL HAS THE NEW ACTIVE DISPLAY PAGE :
	3767	OUTPUT :
	3768	THE 6845 IS RESET TO DISPLAY THAT PAGE :
	3769	-----
F217	3770	ACT_DISP_PAGE PROC NEAR
F217 A26200	3771	MOV ACTIVE_PAGE,AL ; SAVE ACTIVE PAGE VALUE
F21A 8B0E4C00	3772	MOV CX,CRT_LEN ; GET SAVED LENGTH OF REGEN BUFFER
F21E 98	3773	CBW ; CONVERT AL TO WORD
F21F 50	3774	PUSH AX ; SAVE PAGE VALUE
F220 F7E1	3775	MUL CX ; DISPLAY PAGE TIMES REGEN LENGTH
F222 A34E00	3776	MOV CRT_START,AX ; SAVE START ADDRESS FOR
	3777	LATER REQUIREMENTS
F225 8BC8	3778	MOV CX,AX ; START ADDRESS TO CX
F227 D1F9	3779	SAR CX,1 ; DIVIDE BY 2 FOR 6845 HANDLING
F229 B40C	3780	MOV AH,12 ; 6845 REGISTER FOR START ADDRESS
F22B E8AAFF	3781	CALL M16
F22E 5B	3782	POP BX ; RECOVER PAGE VALUE
F22F D1E3	3783	SAL BX,1 ; *2 FOR WORD OFFSET
F231 8B4750	3784	MOV AX,[BX + OFFSET CURSOR_POSN] ; GET CURSOR FOR THIS PAGE
F234 E8CFFF	3785	CALL M18 ; SET THE CURSOR POSITION
F237 EB8C	3786	JMP SHORT VIDEO_RETURN
	3787	ACT_DISP_PAGE ENDP
	3788	-----
	3789	READ_CURSOR :
	3790	THIS ROUTINE READS THE CURRENT CURSOR VALUE FROM THE :
	3791	6845, FORMATS IT, AND SENDS IT BACK TO THE CALLER :
	3792	INPUT :
	3793	BH - PAGE OF CURSOR :
	3794	OUTPUT :
	3795	DX - ROW, COLUMN OF THE CURRENT CURSOR POSITION :
	3796	CX - CURRENT CURSOR MODE :
	3797	-----
F239	3798	READ_CURSOR PROC NEAR
F239 8ADF	3799	MOV BL,BH
F23B 32FF	3800	XOR BH,BH
F23D D1E3	3801	SAL BX,1 ; WORD OFFSET

LOC OBJ	LINE	SOURCE
F23F 005750	3602	MOV DX,[BX+OFFSET CURSOR_POSN]
F242 0B0E6000	3603	MOV CX,CURSOR_MODE
F246 5F	3604	POP DI
F247 5E	3605	POP SI
F248 5B	3606	POP BX
F249 58	3607	POP AX
F24A 58	3608	POP AX
F24B 1F	3609	POP DS
F24C 07	3610	POP ES
F24D CF	3611	IRET
	3612	READ_CURSOR ENDP
	3613	-----
	3614	; SET COLOR :
	3615	; THIS ROUTINE WILL ESTABLISH THE BACKGROUND COLOR, THE OVERSCAN :
	3616	; COLOR, AND THE FOREGROUND COLOR SET FOR MEDIUM RESOLUTION :
	3617	; GRAPHICS :
	3618	; INPUT :
	3619	; (BH) HAS COLOR ID :
	3620	; IF BH=0, THE BACKGROUND COLOR VALUE IS SET
	3621	; FROM THE LOW BITS OF BL (0-31) :
	3622	; IF BH=1, THE PALETTE SELECTION IS MADE :
	3623	; BASED ON THE LOW BIT OF BL :
	3624	; 0=GREEN, RED, YELLOW FOR COLORS 1,2,3 :
	3625	; 1=BLUE, CYAN, MAGENTA FOR COLORS 1,2,3 :
	3626	; (BL) HAS THE COLOR VALUE TO BE USED :
	3627	; OUTPUT :
	3628	; THE COLOR SELECTION IS UPDATED :
	3629	-----
F24E	3630	SET_COLOR PROC NEAR
F24E 0B166300	3631	MOV DX,ADDR_6845 ; I/O PORT FOR PALETTE
F252 03C205	3632	ADD DX,5 ; OVERSCAN PORT
F255 A06600	3633	MOV AL,CRT_PALETTE ; GET THE CURRENT PALETTE VALUE
F258 DAFF	3634	OR BH,BH ; IS THIS COLOR 0?
F25A 750E	3635	JNZ M20 ; OUTPUT COLOR 1
	3636	
	3637	;----- HANDLE COLOR 0 BY SETTING THE BACKGROUND COLOR
	3638	
F25C 24E0	3639	AND AL,0E0H ; TURN OFF LOW 5 BITS OF CURRENT
F25E 80E31F	3640	AND BL,01FH ; TURN OFF HIGH 3 BITS OF INPUT VALUE
F261 0AC3	3641	OR AL,BL ; PUT VALUE INTO REGISTER
F263	3642	M19: ; OUTPUT THE PALETTE
F263 EE	3643	OUT DX,AL ; OUTPUT COLOR SELECTION TO 3D9 PORT
F264 A26600	3644	MOV CRT_PALETTE,AL ; SAVE THE COLOR VALUE
F267 E95BFF	3645	JMP VIDEO_RETURN
	3646	
	3647	;----- HANDLE COLOR 1 BY SELECTING THE PALETTE TO BE USED
	3648	
F26A	3649	M20:
F26A 24DF	3650	AND AL,0DFH ; TURN OFF PALETTE SELECT BIT
F26C D0EB	3651	SHR BL,1 ; TEST THE LOW ORDER BIT OF BL
F26E 73F3	3652	JNC M19 ; ALREADY DONE
F270 0C20	3653	OR AL,20H ; TURN ON PALETTE SELECT BIT
F272 EBEF	3654	JMP M19 ; GO DO IT
	3655	SET_COLOR ENDP
	3656	-----
	3657	; VIDEO STATE :
	3658	; RETURNS THE CURRENT VIDEO STATE IN AX :
	3659	; AH = NUMBER OF COLUMNS ON THE SCREEN :
	3660	; AL = CURRENT VIDEO MODE :
	3661	; BH = CURRENT ACTIVE PAGE :
	3662	-----
F274	3663	VIDEO_STATE PROC NEAR
F274 8A264A00	3664	MOV AH,BYTE PTR CRT_COLS ; GET NUMBER OF COLUMNS
F278 A04900	3665	MOV AL,CRT_MODE ; CURRENT MODE
F27B 8A3E6200	3666	MOV BH,ACTIVE_PAGE ; GET CURRENT ACTIVE PAGE
F27F 5F	3667	POP DI ; RECOVER REGISTERS
F280 5E	3668	POP SI
F281 59	3669	POP CX ; DISCARD SAVED BX
F282 E943FF	3670	JMP M15 ; RETURN TO CALLER
	3671	VIDEO_STATE ENDP
	3672	-----
	3673	; POSITION :
	3674	; THIS SERVICE ROUTINE CALCULATES THE REGEN :
	3675	; BUFFER ADDRESS OF A CHARACTER IN THE ALPHA MODE :
	3676	; INPUT :
	3677	; AX = ROW, COLUMN POSITION :
	3678	; OUTPUT :

```

3879 ; AX = OFFSET OF CHAR POSITION IN REGEN BUFFER :
3880 ;-----
F285 3881 POSITION PROC NEAR
F285 53 3882 PUSH BX ; SAVE REGISTER
F286 8808 3883 MOV BX,AX
F288 8AC4 3884 MOV AL,AH ; ROWS TO AL
F28A F6264A00 3885 MUL BYTE PTR CRT_COLS ; DETERMINE BYTES TO ROW
F28E 32FF 3886 XOR BH,BH
F290 03C3 3887 ADD AX,BX ; ADD IN COLUMN VALUE
F292 D1E0 3888 SAL AX,1 ; * 2 FOR ATTRIBUTE BYTES
F294 5B 3889 POP BX
F295 C3 3890 RET
3891 POSITION ENDP
3892 ;-----
3893 ; SCROLL UP :
3894 ; THIS ROUTINE MOVES A BLOCK OF CHARACTERS UP :
3895 ; ON THE SCREEN :
3896 ; INPUT :
3897 ; (AH) = CURRENT CRT MODE :
3898 ; (AL) = NUMBER OF ROWS TO SCROLL :
3899 ; (CX) = ROW/COLUMN OF UPPER LEFT CORNER :
3900 ; (DX) = ROW/COLUMN OF LOWER RIGHT CORNER :
3901 ; (BH) = ATTRIBUTE TO BE USED ON BLANKED LINE :
3902 ; (DS) = DATA SEGMENT :
3903 ; (ES) = REGEN BUFFER SEGMENT :
3904 ; OUTPUT :
3905 ; NONE -- THE REGEN BUFFER IS MODIFIED :
3906 ;-----
3907 ASSUME CS:CODE,DS:DATA,ES:DATA
F296 3908 SCROLL_UP PROC NEAR
F296 8A08 3909 MOV BL,AL ; SAVE LINE COUNT IN BL
F298 80FC04 3910 CMP AH,4 ; TEST FOR GRAPHICS MODE
F29B 7208 3911 JC N1 ; HANDLE SEPARATELY
F29D 80FC07 3912 CMP AH,7 ; TEST FOR BW CARD
F2A0 7403 3913 JE N1
F2A2 E9F001 3914 JMP GRAPHICS_UP
F2A5 3915 N1: ; UP_CONTINUE
F2A5 53 3916 PUSH BX ; SAVE FILL ATTRIBUTE IN BH
F2A6 8BC1 3917 MOV AX,CX ; UPPER LEFT POSITION
F2A8 E83700 3918 CALL SCROLL_POSITION ; DO SETUP FOR SCROLL
F2AB 7431 3919 JZ N7 ; BLANK_FIELD
F2AD 03F0 3920 ADD SI,AX ; FROM ADDRESS
F2AF 8AE6 3921 MOV AH,DH ; # ROWS IN BLOCK
F2B1 2AE3 3922 SUB AH,BL ; # ROWS TO BE MOVED
F2B3 3923 N2: ; ROW_LOOP
F2B3 E87200 3924 CALL N10 ; MOVE ONE ROW
F2B6 03F5 3925 ADD SI,BP
F2B8 03F0 3926 ADD DI,BP ; POINT TO NEXT LINE IN BLOCK
F2BA FECC 3927 DEC AH ; COUNT OF LINES TO MOVE
F2BC 75F5 3928 JNZ N2 ; ROW_LOOP
F2BE 3929 N3: ; CLEAR_ENTRY
F2BE 5B 3930 POP AX ; RECOVER ATTRIBUTE IN AH
F2BF B020 3931 MOV AL,' ' ; FILL WITH BLANKS
F2C1 3932 N4: ; CLEAR_LOOP
F2C1 E86D00 3933 CALL N11 ; CLEAR THE ROW
F2C4 03F0 3934 ADD DI,BP ; POINT TO NEXT LINE
F2C6 FECC 3935 DEC BL ; COUNTER OF LINES TO SCROLL
F2C8 75F7 3936 JNZ N4 ; CLEAR_LOOP
F2CA 3937 N5: ; SCROLL_END
F2CA E8710C 3938 CALL D05
F2CD 803E490007 3939 CMP CRT_MODE,7 ; IS THIS THE BLACK AND WHITE CARD
F2D2 7407 3940 JE N6 ; IF SO, SKIP THE MODE RESET
F2D4 A06500 3941 MOV AL,CRT_MODE_SET ; GET THE VALUE OF THE MODE SET
F2D7 BAD803 3942 MOV DX,0308H ; ALWAYS SET COLOR CARD PORT
F2DA EE 3943 OUT DX,AL
F2DB 3944 N6: ; VIDEO_RET_HERE
F2DB E9E7FE 3945 JMP VIDEO_RETURN
F2DE 3946 N7: ; BLANK_FIELD
F2DE 8ADE 3947 MOV BL,DH ; GET ROW COUNT
F2E0 E8DC 3948 JMP N3 ; GO CLEAR THAT AREA
3949 SCROLL_UP ENDP
3950
3951 ;----- HANDLE COMMON SCROLL SET UP HERE
3952
F2E2 3953 SCROLL_POSITION PROC NEAR
F2E2 803E490002 3954 CMP CRT_MODE,2 ; TEST FOR SPECIAL CASE HERE
F2E7 7218 3955 JB N9 ; HAVE TO HANDLE 80X25 SEPARATELY

```

LOC OBJ	LINE	SOURCE
F2E9 803E490003	3956	CMP CRT_MODE,3
F2EE 7711	3957	JA N9
	3958	
	3959	!----- 80X25 COLOR CARD SCROLL
	3960	
F2F0 52	3961	PUSH DX
F2F1 BADA03	3962	MOV DX,3DAH ; GUARANTEED TO BE COLOR CARD HERE
F2F4 50	3963	PUSH AX
F2F5	3964	
F2F5 EC	3965	N8: IN AL,DX ; WAIT_DISP_ENABLE
F2F6 A808	3966	TEST AL,8 ; GET PORT
F2F8 74FB	3967	JZ N8 ; WAIT FOR VERTICAL RETRACE
F2FA B025	3968	MOV AL,25H ; WAIT_DISP_ENABLE
F2FC B2D8	3969	MOV DL,0D8H ; DX=3DB
F2FE EE	3970	OUT DX,AL ; TURN OFF VIDEO
F2FF 58	3971	POP AX ; DURING VERTICAL RETRACE
F300 5A	3972	POP DX
F301	3973	
F301 E881FF	3974	N9: CALL POSITION ; CONVERT TO REGEN POINTER
F304 03064E00	3975	ADD AX,CRT_START ; OFFSET OF ACTIVE PAGE
F308 8BF8	3976	MOV DI,AX ; TO ADDRESS FOR SCROLL
F30A 8BF8	3977	MOV SI,AX ; FROM ADDRESS FOR SCROLL
F30C 2BD1	3978	SUB DX,CX ; DX = #ROWS, #COLS IN BLOCK
F30E FEC6	3979	INC DH
F310 FEC2	3980	INC DL ; INCREMENT FOR 0 ORIGIN
F312 32ED	3981	XOR CH,CH ; SET HIGH BYTE OF COUNT TO ZERO
F314 8B2E4A00	3982	MOV BP,CRT_COLS ; GET NUMBER OF COLUMNS IN DISPLAY
F318 03ED	3983	ADD BP,BP ; TIMES 2 FOR ATTRIBUTE BYTE
F31A 8AC3	3984	MOV AL,BL ; GET LINE COUNT
F31C F6264A00	3985	MUL BYTE PTR CRT_COLS ; DETERMINE OFFSET TO FROM ADDRESS
F320 03C0	3986	ADD AX,AX ; #2 FOR ATTRIBUTE BYTE
F322 06	3987	PUSH ES ; ESTABLISH ADDRESSING TO REGEN BUFFER
F323 1F	3988	POP DS ; FOR BOTH POINTERS
F324 80FB00	3989	CMPL BL,0 ; 0 SCROLL MEANS BLANK FIELD
F327 C3	3990	RET ; RETURN WITH FLAGS SET
	3991	SCROLL_POSITION ENDP
	3992	
	3993	!----- MOVE_ROW
	3994	
F328	3995	N10 PROC NEAR
F328 8ACA	3996	MOV CL,DL ; GET # OF COLS TO MOVE
F32A 56	3997	PUSH SI
F32B 57	3998	PUSH DI ; SAVE START ADDRESS
F32C F3	3999	REP MOVSW ; MOVE THAT LINE ON SCREEN
F32D A5		
F32E 5F	4000	POP DI
F32F 5E	4001	POP SI ; RECOVER ADDRESSES
F330 C3	4002	RET
	4003	N10 ENDP
	4004	
	4005	!----- CLEAR_ROW
	4006	
F331	4007	N11 PROC NEAR
F331 8ACA	4008	MOV CL,DL ; GET # COLUMNS TO CLEAR
F333 57	4009	PUSH DI
F334 F3	4010	REP STOSW ; STORE THE FILL CHARACTER
F335 AB		
F336 5F	4011	POP DI
F337 C3	4012	RET
	4013	N11 ENDP
	4014	!-----
	4015	! SCROLL_DOWN :
	4016	! THIS ROUTINE MOVES THE CHARACTERS WITHIN A :
	4017	! DEFINED BLOCK DOWN ON THE SCREEN, FILLING THE :
	4018	! TOP LINES WITH A DEFINED CHARACTER :
	4019	! INPUT :
	4020	! (AH) = CURRENT CRT MODE :
	4021	! (AL) = NUMBER OF LINES TO SCROLL :
	4022	! (CX) = UPPER LEFT CORNER OF REGION :
	4023	! (DX) = LOWER RIGHT CORNER OF REGION :
	4024	! (BH) = FILL CHARACTER :
	4025	! (DS) = DATA SEGMENT :
	4026	! (ES) = REGEN SEGMENT :
	4027	! OUTPUT :
	4028	! NONE -- SCREEN IS SCROLLED :
	4029	!-----
F338	4030	SCROLL_DOWN PROC NEAR

LOC	OBJ	LINE	SOURCE
F338	FD	4031	STD ; DIRECTION FOR SCROLL DOWN
F339	8A08	4032	MOV BL,AL ; LINE COUNT TO BL
F33B	80FC04	4033	CHP AH,4 ; TEST FOR GRAPHICS
F33E	7208	4034	JC NI2
F340	80FC07	4035	CHP AH,7 ; TEST FOR BW CARD
F343	7403	4036	JE NI2
F345	E9A601	4037	JMP GRAPHICS_DOWN
F348		4038	N12: ; CONTINUE DOWN
F348	53	4039	PUSH BX ; SAVE ATTRIBUTE IN BH
F349	8BC2	4040	MOV AX,DX ; LOWER RIGHT CORNER
F34B	E894FF	4041	CALL SCROLL_POSITION ; GET REGEN LOCATION
F34E	7420	4042	JZ NI6
F350	2BF0	4043	SUB SI,AX ; SI IS FROM ADDRESS
F352	8AE6	4044	MOV AH,DH ; GET TOTAL # ROWS
F354	2AE3	4045	SUB AH,BL ; COUNT TO MOVE IN SCROLL
F356		4046	N13: ;
F356	E8CFFF	4047	CALL NI0 ; MOVE ONE ROW
F359	2BF5	4048	SUB SI,BP
F35B	2BFD	4049	SUB DI,BP
F35D	FECC	4050	DEC AH
F35F	75F5	4051	JNZ NI3
F361		4052	N14: ;
F361	58	4053	POP AX ; RECOVER ATTRIBUTE IN AH
F362	8020	4054	MOV AL,' '
F364		4055	N15: ;
F364	EBCAFF	4056	CALL NI1 ; CLEAR ONE ROW
F367	2BFD	4057	SUB DI,BP ; GO TO NEXT ROW
F369	FECB	4058	DEC BL
F36B	75F7	4059	JNZ NI5
F36D	E95AFF	4060	JMP N5 ; SCROLL_END
F370		4061	N16: ;
F370	8ADE	4062	MOV BL,DH
F372	EBED	4063	JMP NI4
		4064	SCROLL_DOWN ENDP
		4065	-----
		4066	; READ_AC_CURRENT ;
		4067	; THIS ROUTINE READS THE ATTRIBUTE AND CHARACTER ;
		4068	; AT THE CURRENT CURSOR POSITION AND RETURNS THEM ;
		4069	; TO THE CALLER ;
		4070	;INPUT ;
		4071	; (AH) = CURRENT CRT MODE ;
		4072	; (BH) = DISPLAY PAGE (ALPHA MODES ONLY) ;
		4073	; (DS) = DATA SEGMENT ;
		4074	; (ES) = REGEN SEGMENT ;
		4075	;OUTPUT ;
		4076	; (AL) = CHAR READ ;
		4077	; (AH) = ATTRIBUTE READ ;
		4078	-----
		4079	ASSUME CS:CODE,DS:DATA,ES:DATA
F374		4080	READ_AC_CURRENT PROC NEAR
F374	80FC04	4081	CHP AH,4 ; IS THIS GRAPHICS
F377	7208	4082	JC P1
F379	80FC07	4083	CHP AH,7 ; IS THIS BW CARD
F37C	7403	4084	JE P1
F37E	E9A802	4085	JMP GRAPHICS_READ
F381		4086	P1: ; READ_AC_CONTINUE
F381	E81A00	4087	CALL FIND_POSITION
F384	8BF3	4088	MOV SI,BX ; ESTABLISH ADDRESSING IN SI
		4089	
		4090	----- WAIT FOR HORIZONTAL RETRACE
		4091	
F386	8B166300	4092	MOV DX,ADDR_6845 ; GET BASE ADDRESS
F38A	83C206	4093	ADD DX,6 ; POINT AT STATUS PORT
F38D	06	4094	PUSH ES
F38E	1F	4095	POP DS ; GET SEGMENT FOR QUICK ACCESS
F38F		4096	P2: ; WAIT FOR RETRACE LOW
F38F	EC	4097	IN AL,DX ; GET STATUS
F390	A801	4098	TEST AL,1 ; IS HORZ RETRACE LOW
F392	75FB	4099	JNZ P2 ; WAIT UNTIL IT IS
F394	FA	4100	CLI ; NO MORE INTERRUPTS
F395		4101	P3: ; WAIT FOR RETRACE HIGH
F395	EC	4102	IN AL,DX ; GET STATUS
F396	A801	4103	TEST AL,1 ; IS IT HIGH
F398	74FB	4104	JZ P3 ; WAIT UNTIL IT IS
F39A	AD	4105	LODSW ; GET THE CHAR/ATTR
F39B	E927FE	4106	JMP VIDEO_RETURN
		4107	READ_AC_CURRENT ENDP

```

4108
F39E 4109 FIND_POSITION PROC NEAR
F39E 8ACF 4110 MOV CL,BH ; DISPLAY PAGE TO CX
F3A0 32ED 4111 XOR CH,CH
F3A2 8BF1 4112 MOV SI,CX ; MOVE TO SI FOR INDEX
F3A4 D1E6 4113 SAL SI,1 ; # 2 FOR WORD OFFSET
F3A6 8B4450 4114 MOV AX,[SI+ OFFSET CURSOR_POSN] ; GET ROW/COLUMN OF THAT PAGE
F3A9 330B 4115 XOR BX,BX ; SET START ADDRESS TO ZERO
F3AB E306 4116 JCXZ P5 ; NO_PAGE
F3AD 4117 P4: ; PAGE_LOOP
F3AD 031E4C00 4118 ADD BX,CRT_LEN ; LENGTH OF BUFFER
F3B1 E2FA 4119 LOOP P4
F3B3 4120 P5: ; NO_PAGE
F3B3 E8CFFE 4121 CALL POSITION ; DETERMINE LOCATION IN REGEN
F3B6 0308 4122 ADD BX,AX ; ADD TO START OF REGEN
F3B8 C3 4123 RET
4124 FIND_POSITION ENDP
4125 ;-----
4126 ; WRITE_AC_CURRENT ;
4127 ; THIS ROUTINE WRITES THE ATTRIBUTE ;
4128 ; AND CHARACTER AT THE CURRENT CURSOR ;
4129 ; POSITION ;
4130 ; INPUT ;
4131 ; (AH) = CURRENT CRT MODE ;
4132 ; (BH) = DISPLAY PAGE ;
4133 ; (CX) = COUNT OF CHARACTERS TO WRITE ;
4134 ; (AL) = CHAR TO WRITE ;
4135 ; (BL) = ATTRIBUTE OF CHAR TO WRITE ;
4136 ; (DS) = DATA SEGMENT ;
4137 ; (ES) = REGEN SEGMENT ;
4138 ; OUTPUT ;
4139 ; NONE ;
4140 ;-----
F3B9 4141 WRITE_AC_CURRENT PROC NEAR
F3B9 80FC04 4142 CMP AH,4 ; IS THIS GRAPHICS
F3BC 7208 4143 JC P6
F3BE 80FC07 4144 CMP AH,7 ; IS THIS BW CARD
F3C1 7403 4145 JE P6
F3C3 E9B201 4146 JMP GRAPHICS_WRITE
F3C6 4147 P6: ; WRITE_AC_CONTINUE
F3C6 8AE3 4148 MOV AH,BL ; GET ATTRIBUTE TO AH
F3C8 50 4149 PUSH AX ; SAVE ON STACK
F3C9 51 4150 PUSH CX ; SAVE WRITE COUNT
F3CA E8D1FF 4151 CALL FIND_POSITION
F3CD 8BF8 4152 MOV DI,BX ; ADDRESS TO DI REGISTER
F3CF 59 4153 POP CX ; WRITE COUNT
F3D0 5B 4154 POP BX ; CHARACTER IN BX REG
F3D1 4155 P7: ; WRITE_LOOP
4156
4157 ;----- WAIT FOR HORIZONTAL RETRACE
4158
F3D1 8B166300 4159 MOV DX,ADDR_6845 ; GET BASE ADDRESS
F3D5 83C206 4160 ADD DX,6 ; POINT AT STATUS PORT
F3D8 4161 P8:
F3D8 EC 4162 IN AL,DX ; GET STATUS
F3D9 A801 4163 TEST AL,1 ; IS IT LOW
F3DB 75FB 4164 JNZ P8 ; WAIT UNTIL IT IS
F3DD FA 4165 CLI ; NO MORE INTERRUPTS
F3DE 4166 P9:
F3DE EC 4167 IN AL,DX ; GET STATUS
F3DF A801 4168 TEST AL,1 ; IS IT HIGH
F3E1 74FB 4169 JZ P9 ; WAIT UNTIL IT IS
F3E3 8BC3 4170 MOV AX,BX ; RECOVER THE CHAR/ATTR
F3E5 AB 4171 STOSW ; PUT THE CHAR/ATTR
F3E6 FB 4172 STI ; INTERRUPTS BACK ON
F3E7 E2E8 4173 LOOP P7 ; AS MANY TIMES AS REQUESTED
F3E9 E9D9FD 4174 JMP VIDEO_RETURN
4175 WRITE_AC_CURRENT ENDP
4176 ;-----
4177 ; WRITE_C_CURRENT ;
4178 ; THIS ROUTINE WRITES THE CHARACTER AT ;
4179 ; THE CURRENT CURSOR POSITION, ATTRIBUTE ;
4180 ; UNCHANGED ;
4181 ; INPUT ;
4182 ; (AH) = CURRENT CRT MODE ;
4183 ; (BH) = DISPLAY PAGE ;
4184 ; (CX) = COUNT OF CHARACTERS TO WRITE ;

```

LOC OBJ

LINE SOURCE

```

4185 ; (AL) = CHAR TO WRITE ;
4186 ; (DS) = DATA SEGMENT ;
4187 ; (ES) = REGEN SEGMENT ;
4188 ; OUTPUT ;
4189 ; NONE ;
4190 ;-----
F3EC 4191 WRITE_C_CURRENT PROC NEAR
F3EC 80FC04 4192 CMP AH,4 ; IS THIS GRAPHICS
F3EF 7208 4193 JC P10
F3F1 80FC07 4194 CMP AH,7 ; IS THIS BW CARD
F3F4 7403 4195 JE P10
F3F6 E97F01 4196 JMP GRAPHICS_WRITE
F3F9 4197 P10:
F3F9 50 4198 PUSH AX ; SAVE ON STACK
F3FA 51 4199 PUSH CX ; SAVE WRITE COUNT
F3FB E8A0FF 4200 CALL FIND_POSITION
F3FE 8BFB 4201 MOV DI,BX ; ADDRESS TO DI
F400 59 4202 POP CX ; WRITE COUNT
F401 5B 4203 POP BX ; BL HAS CHAR TO WRITE
F402 4204 P11: ; WRITE_LOOP
4205
4206 ;----- WAIT FOR HORIZONTAL RETRACE
4207
F402 8B166300 4208 MOV DX,ADDR_6845 ; GET BASE ADDRESS
F406 83C206 4209 ADD DX,6 ; POINT AT STATUS PORT
F409 4210 P12:
F409 EC 4211 IN AL,DX ; GET STATUS
F40A A801 4212 TEST AL,1 ; IS IT LOW
F40C 75FB 4213 JNZ P12 ; WAIT UNTIL IT IS
F40E FA 4214 CLI ; NO MORE INTERRUPTS
F40F 4215 P13:
F40F EC 4216 IN AL,DX ; GET STATUS
F410 A801 4217 TEST AL,1 ; IS IT HIGH
F412 74FB 4218 JZ P13 ; WAIT UNTIL IT IS
F414 8AC3 4219 MOV AL,BL ; RECOVER CHAR
F416 AA 4220 STOSB ; PUT THE CHAR/ATTR
F417 FB 4221 STI ; INTERRUPTS BACK ON
F418 47 4222 INC DI ; BUMP POINTER PAST ATTRIBUTE
F419 E2E7 4223 LOOP P11 ; AS MANY TIMES AS REQUESTED
F41B E9A7FD 4224 JMP VIDEO_RETURN
4225 WRITE_C_CURRENT ENDP
4226 ;-----
4227 ; READ DOT -- WRITE DOT ;
4228 ; THESE ROUTINES WILL WRITE A DOT, OR READ THE DOT AT ;
4229 ; THE INDICATED LOCATION ;
4230 ; ENTRY -- ;
4231 ; DX = ROW (0-199) (THE ACTUAL VALUE DEPENDS ON THE MODE) ;
4232 ; CX = COLUMN ( 0-639) ( THE VALUES ARE NOT RANGE CHECKED ) ;
4233 ; AL = DOT VALUE TO WRITE (1,2 OR 4 BITS DEPENDING ON MODE, ;
4234 ; REQ'D FOR WRITE DOT ONLY, RIGHT JUSTIFIED) ;
4235 ; BIT 7 OF AL=1 INDICATES XOR THE VALUE INTO THE LOCATION ;
4236 ; DS = DATA SEGMENT ;
4237 ; ES = REGEN SEGMENT ;
4238 ; ;
4239 ; EXIT ;
4240 ; AL = DOT VALUE READ, RIGHT JUSTIFIED, READ ONLY ;
4241 ;-----
F41E 4242 ASSUME CS:CODE,DS:DATA,ES:DATA
F41E 4243 READ_DOT PROC NEAR
F41E E83100 4244 CALL R3 ; DETERMINE BYTE POSITION OF DOT
F421 268A04 4245 MOV AL,ES:[SI] ; GET THE BYTE
F424 22C4 4246 AND AL,AH ; MASK OFF THE OTHER BITS IN THE BYTE
F426 D2E0 4247 SHL AL,CL ; LEFT JUSTIFY THE VALUE
F428 8ACE 4248 MOV CL,DH ; GET NUMBER OF BITS IN RESULT
F42A D2C0 4249 ROL AL,CL ; RIGHT JUSTIFY THE RESULT
F42C E996FD 4250 JMP VIDEO_RETURN ; RETURN FROM VIDEO IO
4251 READ_DOT ENDP
4252
4253 WRITE_DOT PROC NEAR
F42F 4254 PUSH AX ; SAVE DOT VALUE
F430 50 4255 PUSH AX ; TWICE
F431 E81E00 4256 CALL R3 ; DETERMINE BYTE POSITION OF THE DOT
F434 D2E8 4257 SHR AL,CL ; SHIFT TO SET UP THE BITS FOR OUTPUT
F436 22C4 4258 AND AL,AH ; STRIP OFF THE OTHER BITS
F438 268A0C 4259 MOV CL,ES:[SI] ; GET THE CURRENT BYTE
F43B 5B 4260 POP BX ; RECOVER XOR FLAG
F43C F6C380 4261 TEST BL,80H ; IS IT ON

```

LOC OBJ	LINE	SOURCE
F43F 750D	4262	JNZ R2 ; YES, XOR THE DOT
F441 F6D4	4263	NOT AH ; SET THE MASK TO REMOVE THE
F443 22CC	4264	AND CL,AH ; INDICATED BITS
F445 0AC1	4265	OR AL,CL ; OR IN THE NEW VALUE OF THOSE BITS
F447	4266	R1: ; FINISH_DOT
F447 268004	4267	MOV ES:[SI],AL ; RESTORE THE BYTE IN MEMORY
F44A 58	4268	POP AX
F44B E977FD	4269	JMP VIDEO_RETURN ; RETURN FROM VIDEO IO
F44E	4270	R2: ; XOR_DOT
F44E 32C1	4271	XOR AL,CL ; EXCLUSIVE OR THE DOTS
F450 EBF5	4272	JMP R1 ; FINISH UP THE WRITING
	4273	WRITE_DOT END
	4274	;
	4275	; THIS SUBROUTINE DETERMINES THE REGEN BYTE LOCATION :
	4276	; OF THE INDICATED ROW COLUMN VALUE IN GRAPHICS MODE. :
	4277	; ENTRY -- :
	4278	; DX = ROW VALUE (0-199) :
	4279	; CX = COLUMN VALUE (0-639) :
	4280	; EXIT -- :
	4281	; SI = OFFSET INTO REGEN BUFFER FOR BYTE OF INTEREST :
	4282	; AH = MASK TO STRIP OFF THE BITS OF INTEREST :
	4283	; CL = BITS TO SHIFT TO RIGHT JUSTIFY THE MASK IN AH :
	4284	; DH = # BITS IN RESULT :
	4285	;
F452	4286	R3 PROC NEAR
F452 53	4287	PUSH BX ; SAVE BX DURING OPERATION
F453 50	4288	PUSH AX ; WILL SAVE AL DURING OPERATION
	4289	
	4290	;
	4291	;----- DETERMINE 1ST BYTE IN IDICATED ROW BY MULTIPLYING ROW VALUE BY 40
	4292	;----- (LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW
F454 B028	4293	MOV AL,40
F456 52	4294	PUSH DX ; SAVE ROW VALUE
F457 80E2FE	4295	AND DL,0FEH ; STRIP OFF ODD/EVEN BIT
F45A F6E2	4296	MUL DL ; AX HAS ADDRESS OF 1ST BYTE
	4297	; OF INDICATED ROW
F45C 5A	4298	POP DX ; RECOVER IT
F45D F6C201	4299	TEST DL,1 ; TEST FOR EVEN/ODD
F460 7403	4300	JZ R4 ; JUMP IF EVEN ROW
F462 050020	4301	ADD AX,2000H ; OFFSET TO LOCATION OF ODD ROWS
F465	4302	R4: ; EVEN_ROW
F465 8BF0	4303	MOV SI,AX ; MOVE POINTER TO SI
F467 58	4304	POP AX ; RECOVER AL VALUE
F468 8BD1	4305	MOV DX,CX ; COLUMN VALUE TO DX
	4306	
	4307	;
	4308	;----- DETERMINE GRAPHICS MODE CURRENTLY IN EFFECT
	4309	;
	4310	; SET UP THE REGISTERS ACCORDING TO THE MODE :
	4311	; CH = MASK FOR LOW OF COLUMN ADDRESS (7/3 FOR HIGH/MED RES) :
	4312	; CL = # OF ADDRESS BITS IN COLUMN VALUE (3/2 FOR H/M) :
	4313	; BL = MASK TO SELECT BITS FROM POINTED BYTE (80H/COH FOR H/M) :
	4314	; BH = NUMBER OF VALID BITS IN POINTED BYTE (1/2 FOR H/M) :
	4315	;
	4316	;
F46A BBC002	4317	MOV BX,2C0H
F46D B90203	4318	MOV CX,302H ; SET PARMS FOR MED RES
F470 803E490006	4319	CHP CRT_MODE,6
F475 7206	4320	JC R5 ; HANDLE IF MED ARE
F477 BB8001	4321	MOV BX,180H
F47A B90307	4322	MOV CX,703H ; SET PARMS FOR HIGH RES
	4323	
	4324	;
	4325	;----- DETERMINE BIT OFFSET IN BYTE FROM COLUMN MASK
F47D	4326	R5:
F47D 22EA	4327	AND CH,DL ; ADDRESS OF PEL WITHIN BYTE TO CH
	4328	
	4329	;
	4330	;----- DETERMINE BYTE OFFSET FOR THIS LOCATION IN COLUMN
F47F D3EA	4331	SHR DX,CL ; SHIFT BY CORRECT AMOUNT
F481 03F2	4332	ADD SI,DX ; INCREMENT THE POINTER
F483 8AF7	4333	MOV DH,BH ; GET THE # OF BITS IN RESULT TO DH
	4334	
	4335	;
	4336	;----- MULTIPLY BH (VALID BITS IN BYTE) BY CH (BIT OFFSET)
F485 2AC9	4337	SUB CL,CL ; ZERO INTO STORAGE LOCATION
F487	4338	R6:

LOC OBJ	LINE	SOURCE
F467 D0C8	4339	ROR AL,1 ; LEFT JUSTIFY THE VALUE
	4340	; IN AL (FOR WRITE)
F469 02CD	4341	ADD CL,CH ; ADD IN THE BIT OFFSET VALUE
F46B FECF	4342	DEC BH ; LOOP CONTROL
F46D 75F8	4343	JNZ R6 ; ON EXIT, CL HAS SHIFT COUNT
	4344	; TO RESTORE BITS
F46F 8AE3	4345	MOV AH,BL ; GET MASK TO AH
F491 D2EC	4346	SHR AH,CL ; MOVE THE MASK TO CORRECT LOCATION
F493 5B	4347	POP BX ; RECOVER REG
F494 C3	4348	RET ; RETURN WITH EVERYTHING SET UP
	4349	R3 ENDP
	4350	;------
	4351	; SCROLL UP ;
	4352	; THIS ROUTINE SCROLLS UP THE INFORMATION ON THE CRT ;
	4353	; ENTRY ;
	4354	; CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL ;
	4355	; DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL ;
	4356	; BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS ;
	4357	; BH = FILL VALUE FOR BLANKED LINES ;
	4358	; AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE ;
	4359	; FIELD) ;
	4360	; DS = DATA SEGMENT ;
	4361	; ES = REGEN SEGMENT ;
	4362	; EXIT ;
	4363	; NOTHING, THE SCREEN IS SCROLLED ;
	4364	;------
F495	4365	GRAPHICS_UP PROC NEAR
F495 8AD8	4366	MOV BL,AL ; SAVE LINE COUNT IN BL
F497 8BC1	4367	MOV AX,CX ; GET UPPER LEFT POSITION INTO AX REG
	4368	
	4369	;----- USE CHARACTER SUBROUTINE FOR POSITIONING
	4370	;----- ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
	4371	
F499 E86902	4372	CALL GRAPH_POSN
F49C 8BF8	4373	MOV DI,AX ; SAVE RESULT AS DESTINATION ADDRESS
	4374	
	4375	;----- DETERMINE SIZE OF WINDOW
	4376	
F49E 2BD1	4377	SUB DX,CX
F4A0 81C20101	4378	ADD DX,101H ; ADJUST VALUES
F4A4 D0E6	4379	SAL DH,1 ; MULTIPLY # ROWS BY 4
	4380	; SINCE 8 VERT DOTS/CHAR
F4A6 D0E6	4381	SAL DH,1 ; AND EVEN/ODD ROWS
	4382	
	4383	;----- DETERMINE CRT MODE
	4384	
F4A8 803E490006	4385	CMP CRT_MODE,6 ; TEST FOR MEDIUM RES
F4AD 7304	4386	JNC R7 ; FIND_SOURCE
	4387	
	4388	;----- MEDIUM RES UP
	4389	
F4AF D0E2	4390	SAL DL,1 ; # COLUMNS # 2, SINCE 2 BYTES/CHAR
F4B1 D1E7	4391	SAL DI,1 ; OFFSET #2 SINCE 2 BYTES/CHAR
	4392	
	4393	;----- DETERMINE THE SOURCE ADDRESS IN THE BUFFER
	4394	
F4B3	4395	R7: ; FIND_SOURCE
F4B3 06	4396	PUSH ES ; GET SEGMENTS BOTH POINTING TO REGEN
F4B4 1F	4397	POP DS
F4B5 2AED	4398	SUB CH,CH ; ZERO TO HIGH OF COUNT REG
F4B7 D0E3	4399	SAL BL,1 ; MULTIPLY NUMBER OF LINES BY 4
F4B9 D0E3	4400	SAL BL,1
F4BB 742D	4401	JZ R11 ; IF ZERO, THEN BLANK ENTIRE FIELD
F4BD 8AC3	4402	MOV AL,BL ; GET NUMBER OF LINES IN AL
F4BF B450	4403	MOV AH,80 ; 80 BYTES/ROW
F4C1 F6E4	4404	MUL AH ; DETERMINE OFFSET TO SOURCE
F4C3 8BF7	4405	MOV SI,DI ; SET UP SOURCE
F4C5 03F0	4406	ADD SI,AX ; ADD IN OFFSET TO IT
F4C7 8AE6	4407	MOV AH,DH ; NUMBER OF ROWS IN FIELD
F4C9 2AE3	4408	SUB AH,BL ; DETERMINE NUMBER TO MOVE
	4409	
	4410	;----- LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
	4411	
F4CB	4412	R8: ; ROW_LOOP
F4CB E88000	4413	CALL R17 ; MOVE ONE ROW
F4CE 81EEB01F	4414	SUB SI,2000H-80 ; MOVE TO NEXT ROW
F4D2 81EFB01F	4415	SUB DI,2000H-80

LOC OBJ	LINE	SOURCE	
F4D6 FECC	4416	DEC AH	; NUMBER OF ROWS TO MOVE
F4D8 75F1	4417	JNZ R8	; CONTINUE TILL ALL MOVED
	4418		
	4419	;----- FILL IN THE VACATED LINE(S)	
	4420		
F4DA	4421	R9:	; CLEAR_ENTRY
F4DA 8AC7	4422	MOV AL,BH	; ATTRIBUTE TO FILL WITH
F4DC	4423	R10:	
F4DC E8800	4424	CALL R18	; CLEAR THAT ROW
F4DF 81EFB01F	4425	SUB DI,2000H-80	; POINT TO NEXT LINE
F4E3 FECB	4426	DEC BL	; NUMBER OF LINES TO FILL
F4E5 75F5	4427	JNZ R10	; CLEAR_LOOP
F4E7 E90BFC	4428	JMP VIDEO_RETURN	; EVERYTHING DONE
F4EA	4429	R11:	; BLANK_FIELD
F4EA 8ADE	4430	MOV BL,DH	; SET BLANK COUNT TO
	4431		; EVERYTHING IN FIELD
F4EC EBEC	4432	JMP R9	; CLEAR THE FIELD
	4433	GRAPHICS_UP ENDP	
	4434	;-----	
	4435	; SCROLL DOWN	:
	4436	; THIS ROUTINE SCROLLS DOWN THE INFORMATION ON THE CRT	:
	4437	; ENTRY	:
	4438	; CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL	:
	4439	; DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL	:
	4440	; BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS	:
	4441	; BH = FILL VALUE FOR BLANKED LINES	:
	4442	; AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE	:
	4443	; FIELD)	:
	4444	; DS = DATA SEGMENT	:
	4445	; ES = REGEN SEGMENT	:
	4446	; EXIT	:
	4447	; NOTHING, THE SCREEN IS SCROLLED	:
	4448	;-----	
F4EE	4449	GRAPHICS_DOWN PROC NEAR	
F4EE FD	4450	STD	; SET DIRECTION
F4EF 8AD8	4451	MOV BL,AL	; SAVE LINE COUNT IN BL
F4F1 8BC2	4452	MOV AX,DX	; GET LOWER RIGHT POSITION INTO AX REG
	4453		
	4454	;----- USE CHARACTER SUBROUTINE FOR POSITIONING	
	4455	;----- ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE	
	4456		
F4F3 E80F02	4457	CALL GRAPH_POSN	
F4F6 8BF8	4458	MOV DI,AX	; SAVE RESULT AS DESTINATION ADDRESS
	4459		
	4460	;----- DETERMINE SIZE OF WINDOW	
	4461		
F4F8 2BD1	4462	SUB DX,CX	
F4FA 81C20101	4463	ADD DX,101H	; ADJUST VALUES
F4FE D0E6	4464	SAL DH,1	; MULTIPLY # ROWS BY 4
	4465		; SINCE 8 VERT DOTS/CHAR
F500 D0E6	4466	SAL DH,1	; AND EVEN/ODD ROWS
	4467		
	4468	;----- DETERMINE CRT MODE	
	4469		
F502 803E490006	4470	CMF CRT_MODE,6	; TEST FOR MEDIUM RES
F507 7305	4471	JNC R12	; FIND_SOURCE_DOWN
	4472		
	4473	;----- MEDIUM RES DOWN	
	4474		
F509 D0E2	4475	SAL DL,1	; # COLUMNS * 2, SINCE
	4476		; 2 BYTES/CHAR (OFFSET OK)
F50B D1E7	4477	SAL DI,1	; OFFSET #2 SINCE 2 BYTES/CHAR
F50D 47	4478	INC DI	; POINT TO LAST BYTE
	4479		
	4480	;----- DETERMINE THE SOURCE ADDRESS IN THE BUFFER	
	4481		
F50E	4482	R12:	; FIND_SOURCE_DOWN
F50E 06	4483	PUSH ES	; BOTH SEGMENTS TO REGEN
F50F 1F	4484	POP DS	
F510 2AED	4485	SUB CH,CH	; ZERO TO HIGH OF COUNT REG
F512 81C7F000	4486	ADD DI,240	; POINT TO LAST ROW OF PIXELS
F516 D0E3	4487	SAL BL,1	; MULTIPLY NUMBER OF LINES BY 4
F518 D0E3	4488	SAL BL,1	
F51A 742E	4489	JZ R16	; IF ZERO, THEN BLANK ENTIRE FIELD
F51C 8AC3	4490	MOV AL,BL	; GET NUMBER OF LINES IN AL
F51E B450	4491	MOV AH,60	; 60 BYTES/ROW
F520 F6E4	4492	MUL AH	; DETERMINE OFFSET TO SOURCE

LOC OBJ	LINE	SOURCE
F522 8BF7	4493	MOV SI,DI ; SET UP SOURCE
F524 2BF0	4494	SUB SI,AX ; SUBTRACT THE OFFSET
F526 8AE6	4495	MOV AH,DH ; NUMBER OF ROWS IN FIELD
F528 2AE3	4496	SUB AH,BL ; DETERMINE NUMBER TO MOVE
	4497	
	4498	;----- LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
	4499	
F52A	4500	R13: ; ROW_LOOP_DOWN
F52A E82100	4501	CALL R17 ; MOVE ONE ROW
F52D 81EE5020	4502	SUB SI,2000H+80 ; MOVE TO NEXT ROW
F531 81EF5020	4503	SUB DI,2000H+80
F535 FECC	4504	DEC AH ; NUMBER OF ROWS TO MOVE
F537 75F1	4505	JNZ R13 ; CONTINUE TILL ALL MOVED
	4506	
	4507	;----- FILL IN THE VACATED LINE(S)
	4508	
F539	4509	R14: ; CLEAR_ENTRY_DOWN
F539 8AC7	4510	MOV AL,BH ; ATTRIBUTE TO FILL WITH
F53B	4511	R15: ; CLEAR_LOOP_DOWN
F53B E82900	4512	CALL R18 ; CLEAR A ROW
F53E 81EF5020	4513	SUB DI,2000H+80 ; POINT TO NEXT LINE
F542 FECB	4514	DEC BL ; NUMBER OF LINES TO FILL
F544 75F5	4515	JNZ R15 ; CLEAR_LOOP_DOWN
F546 FC	4516	CLD ; RESET THE DIRECTION FLAG
F547 E97BFC	4517	JMP VIDEO_RETURN ; EVERYTHING DONE
F54A	4518	R16: ; BLANK_FIELD_DOWN
F54A 8ADE	4519	MOV BL,DH ; SET BLANK COUNT TO
	4520	; EVERYTHING IN FIELD
F54C EBEB	4521	JMP R14 ; CLEAR THE FIELD
	4522	GRAPHICS_DOWN ENDP
	4523	
	4524	;----- ROUTINE TO MOVE ONE ROW OF INFORMATION
	4525	
F54E	4526	R17 PROC NEAR
F54E 8ACA	4527	MOV CL,DL ; NUMBER OF BYTES IN THE ROW
F550 56	4528	PUSH SI
F551 57	4529	PUSH DI ; SAVE POINTERS
F552 F3	4530	REP MOVSB ; MOVE THE EVEN FIELD
F553 A4		
F554 5F	4531	POP DI
F555 5E	4532	POP SI
F556 81C60020	4533	ADD SI,2000H
F55A 81C70020	4534	ADD DI,2000H ; POINT TO THE ODD FIELD
F55E 56	4535	PUSH SI
F55F 57	4536	PUSH DI ; SAVE THE POINTERS
F560 8ACA	4537	MOV CL,DL ; COUNT BACK
F562 F3	4538	REP MOVSB ; MOVE THE ODD FIELD
F563 A4		
F564 5F	4539	POP DI
F565 5E	4540	POP SI ; POINTERS BACK
F566 C3	4541	RET ; RETURN TO CALLER
	4542	R17 ENDP
	4543	
	4544	;----- CLEAR A SINGLE ROW
	4545	
F567	4546	R18 PROC NEAR
F567 8ACA	4547	MOV CL,DL ; NUMBER OF BYTES IN FIELD
F569 57	4548	PUSH DI ; SAVE POINTER
F56A F3	4549	REP STOSB ; STORE THE NEW VALUE
F56B AA		
F56C 5F	4550	POP DI ; POINTER BACK
F56D 81C70020	4551	ADD DI,2000H ; POINT TO ODD FIELD
F571 57	4552	PUSH DI
F572 8ACA	4553	MOV CL,DL
F574 F3	4554	REP STOSB ; FILL THE ODD FIELD
F575 AA		
F576 5F	4555	POP DI
F577 C3	4556	RET ; RETURN TO CALLER
	4557	R18 ENDP
	4558	;-----
	4559	; GRAPHICS WRITE ;
	4560	; THIS ROUTINE WRITES THE ASCII CHARACTER TO THE ;
	4561	; CURRENT POSITION ON THE SCREEN. ;
	4562	; ENTRY ;
	4563	; AL = CHARACTER TO WRITE ;
	4564	; BL = COLOR ATTRIBUTE TO BE USED FOR FOREGROUND COLOR ;
	4565	; IF BIT 7 IS SET, THE CHAR IS XOR'D INTO THE REGEN ;

```

4566 ;      BUFFER (0 IS USED FOR THE BACKGROUND COLOR)      :
4567 ;      CX = NUMBER OF CHARS TO WRITE                      :
4568 ;      DS = DATA SEGMENT                                :
4569 ;      ES = REGEN SEGMENT                                  :
4570 ; EXIT                                                      :
4571 ;      NOTHING IS RETURNED                                :
4572 ;                                                          :
4573 ; GRAPHICS READ                                           :
4574 ;      THIS ROUTINE READS THE ASCII CHARACTER AT THE CURRENT :
4575 ;      CURSOR POSITION ON THE SCREEN BY MATCHING THE DOTS ON :
4576 ;      THE SCREEN TO THE CHARACTER GENERATOR CODE POINTS :
4577 ; ENTRY                                                    :
4578 ;      NONE ( 0 IS ASSUMED AS THE BACKGROUND COLOR)       :
4579 ; EXIT                                                      :
4580 ;      AL = CHARACTER READ AT THAT POSITION (0 RETURNED IF :
4581 ;      NONE FOUND)                                         :
4582 ;                                                          :
4583 ; FOR BOTH ROUTINES, THE IMAGES USED TO FORM CHARS ARE :
4584 ; CONTAINED IN ROM FOR THE 1ST 128 CHARS. TO ACCESS CHARS :
4585 ; IN THE SECOND HALF, THE USER MUST INITIALIZE THE VECTOR AT :
4586 ; INTERRUPT 1FH (LOCATION 0007CH) TO POINT TO THE USER :
4587 ; SUPPLIED TABLE OF GRAPHIC IMAGES (8X8 BOXES).         :
4588 ; FAILURE TO DO SO WILL CAUSE IN STRANGE RESULTS         :
4589 ;-----
4590 ;      ASSUME CS:CODE,DS:DATA,ES:DATA
F578 4591 GRAPHICS_WRITE PROC NEAR
F578 B400 4592     MOV     AH,0      ; ZERO TO HIGH OF CODE POINT
F57A 50 4593     PUSH    AX       ; SAVE CODE POINT VALUE
4594
4595 ;----- DETERMINE POSITION IN REGEN BUFFER TO PUT CODE POINTS
4596
F57B E88401 4597     CALL    S26      ; FIND LOCATION IN REGEN BUFFER
F57E 8BF8 4598     MOV     DI,AX    ; REGEN POINTER IN DI
4599
4600 ;----- DETERMINE REGION TO GET CODE POINTS FROM
4601
F580 58 4602     POP     AX       ; RECOVER CODE POINT
F581 3C80 4603     CMP     AL,80H  ; IS IT IN SECOND HALF
F583 7306 4604     JAE     SI     ; YES
4605
4606 ;----- IMAGE IS IN FIRST HALF, CONTAINED IN ROM
4607
F585 BE6EFA 4608     MOV     SI,0FA6EH ; CRT_CHAR_GEN (OFFSET OF IMAGES)
F588 0E 4609     PUSH    CS       ; SAVE SEGMENT ON STACK
F589 E80F 4610     JMP     SHORT S2 ; DETERMINE_MODE
4611
4612 ;----- IMAGE IS IN SECOND HALF, IN USER RAM
4613
4614 S1:
4615     SUB     AL,80H    ; EXTEND_CHAR
4616     PUSH    DS       ; ZERO ORIGIN FOR SECOND HALF
4617     SUB     SI,SI     ; SAVE DATA POINTER
4618     MOV     DS,SI    ; ESTABLISH VECTOR ADDRESSING
4619     ASSUME  DS:ABS0
4620     LDS     SI,EXT_PTR ; GET THE OFFSET OF THE TABLE
4621     MOV     DX,DS     ; GET THE SEGMENT OF THE TABLE
4622     ASSUME  DS:DATA
4623     POP     DS       ; RECOVER DATA SEGMENT
4624     PUSH    DX       ; SAVE TABLE SEGMENT ON STACK
4625
4626 ;----- DETERMINE GRAPHICS MODE IN OPERATION
4627
F59A 4628 S2:
F59A D1E0 4629     SAL     AX,1     ; DETERMINE_MODE
F59C D1E0 4630     SAL     AX,1     ; MULTIPLY CODE POINT
F59E D1E0 4631     SAL     AX,1     ; VALUE BY 8
F5A0 03F0 4632     ADD     SI,AX   ; SI HAS OFFSET OF DESIRED CODES
F5A2 803E490006 4633     CMP     CRT_MODE,6
F5A7 1F 4634     POP     DS     ; RECOVER TABLE POINTER SEGMENT
F5A8 722C 4635     JC      S7      ; TEST FOR MEDIUM RESOLUTION MODE
4636
4637 ;----- HIGH RESOLUTION MODE
4638
F5AA 4639 S3:
F5AA 57 4640     PUSH    DI       ; HIGH_CHAR
F5AB 56 4641     PUSH    SI       ; SAVE REGEN POINTER
F5AC B604 4642     MOV     DH,4     ; SAVE CODE POINTER
                     ; NUMBER OF TIMES THROUGH LOOP

```


LOC OBJ	LINE	SOURCE	
F5AE	4643	S4:	
F5AE AC	4644	LODSB	: GET BYTE FROM CODE POINTS
F5AF F6C380	4645	TEST BL,80H	: SHOULD WE USE THE FUNCTION
F5B2 7516	4646	JNZ S6	: TO PUT CHAR IN
F5B4 AA	4647	STOSB	: STORE IN REGEN BUFFER
F5B5 AC	4648	LODSB	
F5B6	4649	S5:	
F5B6 268085FF1F	4650	MOV ES:[DI+2000H-1],AL	: STORE IN SECOND HALF
F5BB 83C74F	4651	ADD DI,79	: MOVE TO NEXT ROW IN REGEN
F5BE FECE	4652	DEC DH	: DONE WITH LOOP
F5C0 75EC	4653	JNZ S4	
F5C2 5E	4654	POP SI	
F5C3 5F	4655	POP DI	: RECOVER REGEN POINTER
F5C4 47	4656	INC DI	: POINT TO NEXT CHAR POSITION
F5C5 E2E3	4657	LOOP S3	: MORE CHARS TO WRITE
F5C7 E9FBFB	4658	JMP VIDEO_RETURN	
F5CA	4659	S6:	
F5CA 263205	4660	XOR AL,ES:[DI]	: EXCLUSIVE OR WITH CURRENT
F5CD AA	4661	STOSB	: STORE THE CODE POINT
F5CE AC	4662	LODSB	: AGAIN FOR ODD FIELD
F5CF 263285FF1F	4663	XOR AL,ES:[DI+2000H-1]	
F5D4 EBE0	4664	JMP S5	: BACK TO MAINSTREAM
	4665		
	4666	:----- MEDIUM RESOLUTION WRITE	
	4667		
F5D6	4668	S7:	: MED_RES_WRITE
F5D6 8AD3	4669	MOV DL,BL	: SAVE HIGH COLOR BIT
F5D8 D1E7	4670	SAL DI,1	: OFFSET*2 SINCE 2 BYTES/CHAR
F5DA E8D100	4671	CALL S19	: EXPAND BL TO FULL WORD OF COLOR
F5DD	4672	S8:	: MED_CHAR
F5DD 57	4673	PUSH DI	: SAVE REGEN POINTER
F5DE 56	4674	PUSH SI	: SAVE THE CODE POINTER
F5DF B604	4675	MOV DH,4	: NUMBER OF LOOPS
F5E1	4676	S9:	
F5E1 AC	4677	LODSB	: GET CODE POINT
F5E2 E8DE00	4678	CALL S21	: DOUBLE UP ALL THE BITS
F5E5 23C3	4679	AND AX,BX	: CONVERT THEM TO FOREGROUND
	4680		: COLOR (0 BACK)
F5E7 F6C280	4681	TEST DL,80H	: IS THIS XOR FUNCTION
F5EA 7407	4682	JZ S10	: NO, STORE IT IN AS IT IS
F5EC 263225	4683	XOR AH,ES:[DI]	: DO FUNCTION WITH HALF
F5EF 26324501	4684	XOR AL,ES:[DI+1]	: AND WITH OTHER HALF
F5F3	4685	S10:	
F5F3 268825	4686	MOV ES:[DI],AH	: STORE FIRST BYTE
F5F6 26884501	4687	MOV ES:[DI+1],AL	: STORE SECOND BYTE
F5FA AC	4688	LODSB	: GET CODE POINT
F5FB E8C500	4689	CALL S21	
F5FE 23C3	4690	AND AX,BX	: CONVERT TO COLOR
F600 F6C280	4691	TEST DL,80H	: AGAIN, IS THIS XOR FUNCTION
F603 740A	4692	JZ S11	: NO, JUST STORE THE VALUES
F605 2632A50020	4693	XOR AH,ES:[DI+2000H]	: FUNCTION WITH FIRST HALF
F60A 2632850120	4694	XOR AL,ES:[DI+2001H]	: AND WITH SECOND HALF
F60F	4695	S11:	
F60F 2688A50020	4696	MOV ES:[DI+2000H],AH	
F614 2688850120	4697	MOV ES:[DI+2000H+1],AL	: STORE IN SECOND PORTION OF BUFFER
F619 83C750	4698	ADD DI,80	: POINT TO NEXT LOCATION
F61C FECE	4699	DEC DH	
F61E 75C1	4700	JNZ S9	: KEEP GOING
F620 5E	4701	POP SI	: RECOVER CODE PONTER
F621 5F	4702	POP DI	: RECOVER REGEN POINTER
F622 47	4703	INC DI	: POINT TO NEXT CHAR POSITION
F623 47	4704	INC DI	
F624 E2B7	4705	LOOP S8	: MORE TO WRITE
F626 E9FCFB	4706	JMP VIDEO_RETURN	
	4707	GRAPHICS_WRITE ENDP	
	4708	:-----	
	4709	: GRAPHICS_READ :	
	4710	:-----	
F629	4711	GRAPHICS_READ PROC NEAR	
F629 E80600	4712	CALL S26	: CONVERTED TO OFFSET IN REGEN
F62C 8BF0	4713	MOV SI,AX	: SAVE IN SI
F62E 83EC08	4714	SUB SP,8	: ALLOCATE SPACE TO SAVE THE
	4715		: READ CODE POINT
F631 8BEC	4716	MOV BP,SP	: POINTER TO SAVE AREA
	4717		
	4718	:----- DETERMINE GRAPHICS MODES	
	4719		

LOC	OBJ	LINE	SOURCE
F633	803E490006	4720	CHP CRT_MODE,6
F638	06	4721	PUSH ES
F639	1F	4722	POP DS ; POINT TO REGEN SEGMENT
F63A	721A	4723	JC S13 ; MEDIUM RESOLUTION
		4724	
		4725	;----- HIGH RESOLUTION READ
		4726	
		4727	;----- GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT
		4728	
F63C	B604	4729	MOV DH,4 ; NUMBER OF PASSES
F63E		4730	S12:
F63E	8A04	4731	MOV AL,[SI] ; GET FIRST BYTE
F640	8B4600	4732	MOV [BP],AL ; SAVE IN STORAGE AREA
F643	45	4733	INC BP ; NEXT LOCATION
F644	8A840020	4734	MOV AL,[SI+2000H] ; GET LOWER REGION BYTE
F648	8B4600	4735	MOV [BP],AL ; ADJUST AND STORE
F64B	45	4736	INC BP
F64C	83C650	4737	ADD SI,80 ; POINTER INTO REGEN
F64F	FECE	4738	DEC DH ; LOOP CONTROL
F651	75EB	4739	JNZ S12 ; DO IT SOME MORE
F653	EB1790	4740	JMP S15 ; GO MATCH THE SAVED CODE POINTS
		4741	
		4742	;----- MEDIUM RESOLUTION READ
		4743	
F656		4744	S13:
F656	D1E6	4745	SAL SI,1 ; MED_RES_READ
F658	B604	4746	MOV DH,4 ; OFFSET*2 SINCE 2 BYTES/CHAR
F65A		4747	S14:
F65A	E88800	4748	CALL S23 ; GET PAIR BYTES FROM REGEN
		4749	;
F65D	81C60020	4750	ADD SI,2000H ; INTO SINGLE SAVE
F661	E80100	4751	CALL S23 ; GO TO LOWER REGION
F664	81EEB01F	4752	SUB SI,2000H-80 ; GET THIS PAIR INTO SAVE
F668	FECE	4753	DEC DH ; ADJUST POINTER BACK INTO UPPER
F66A	75EE	4754	JNZ S14 ; KEEP GOING UNTIL ALL 8 DONE
		4755	
		4756	;----- SAVE AREA HAS CHARACTER IN IT, MATCH IT
		4757	
F66C		4758	S15:
F66C	BF6EFA90	4759	MOV DI,OFFSET CRT_CHAR_GEN ; FIND_CHAR
F670	0E	4760	PUSH CS ; ESTABLISH ADDRESSING
F671	07	4761	POP ES ; CODE POINTS IN CS
F672	83ED08	4762	SUB BP,8 ; ADJUST POINTER TO BEGINNING
		4763	;
F675	8BF5	4764	MOV SI,BP ; OF SAVE AREA
F677	FC	4765	CLD ; ENSURE DIRECTION
F678	B000	4766	MOV AL,0 ; CURRENT CODE POINT BEING MATCHED
F67A		4767	S16:
F67A	16	4768	PUSH SS ; ESTABLISH ADDRESSING TO STACK
F67B	1F	4769	POP DS ; FOR THE STRING COMPARE
F67C	BA8000	4770	MOV DX,128 ; NUMBER TO TEST AGAINST
F67F		4771	S17:
F67F	56	4772	PUSH SI ; SAVE SAVE AREA POINTER
F680	57	4773	PUSH DI ; SAVE CODE POINTER
F681	B90800	4774	MOV CX,8 ; NUMBER OF BYTES TO MATCH
F684	F3	4775	REPE CMPSB ; COMPARE THE 8 BYTES
F685	A6		
F686	5F	4776	POP DI ; RECOVER THE POINTERS
F687	5E	4777	POP SI
F688	741E	4778	JZ S18 ; IF ZERO FLAG SET, THEN MATCH OCCURRED
F68A	FEC0	4779	INC AL ; NO MATCH, MOVE ON TO NEXT
F68C	83C708	4780	ADD DI,8 ; NEXT CODE POINT
F68F	4A	4781	DEC DX ; LOOP CONTROL
F690	75ED	4782	JNZ S17 ; DO ALL OF THEM
		4783	
		4784	;----- CHAR NOT MATCHED, MIGHT BE IN USER SUPPLIED SECOND HALF
		4785	
F692	3C00	4786	CHP AL,0 ; AL <> 0 IF ONLY 1ST HALF SCANNED
F694	7412	4787	JE S18 ; IF = 0, THEN ALL HAS BEEN SCANNED
F696	2BC0	4788	SUB AX,AX
F698	8ED8	4789	MOV DS,AX ; ESTABLISH ADDRESSING TO VECTOR
		4790	ASSUME DS:ABS0
F69A	C43E7C00	4791	LES DI,EXT_PTR ; GET POINTER
F69E	8CC0	4792	MOV AX,ES ; SEE IF THE POINTER REALLY EXISTS
F6A0	0BC7	4793	OR AX,DI ; IF ALL 0, THEN DOESN'T EXIST
F6A2	7404	4794	JZ S18 ; NO SENSE LOOKING
F6A4	B080	4795	MOV AL,128 ; ORIGIN FOR SECOND HALF

LOC OBJ	LINE	SOURCE
F6A6 EBD2	4796	JMP S16 ; GO BACK AND TRY FOR IT
	4797	ASSUME DS:DATA
	4798	
	4799	;----- CHARACTER IS FOUND (AL=0 IF NOT FOUND)
	4800	
F6A8	4801	S18:
F6A8 83C408	4802	ADD SP,8 ; READJUST THE STACK, THROW AWAY SAVE
F6AB E917FB	4803	JMP VIDEO_RETURN ; ALL DONE
	4804	GRAPHICS_READ ENDP
	4805	;-----
	4806	; EXPAND_HED_COLOR ;
	4807	; THIS ROUTINE EXPANDS THE LOW 2 BITS IN BL TO ;
	4808	; FILL THE ENTIRE BX REGISTER ;
	4809	; ENTRY ;
	4810	; BL = COLOR TO BE USED (LOW 2 BITS) ;
	4811	; EXIT ;
	4812	; BX = COLOR TO BE USED (8 REPLICATIONS OF THE ;
	4813	; 2 COLOR BITS) ;
	4814	;-----
F6AE	4815	S19 PROC NEAR
F6AE 80E303	4816	AND BL,3 ; ISOLATE THE COLOR BITS
F6B1 8AC3	4817	MOV AL,BL ; COPY TO AL
F6B3 51	4818	PUSH CX ; SAVE REGISTER
F6B4 B90300	4819	MOV CX,3 ; NUMBER OF TIMES TO DO THIS
F6B7	4820	S20:
F6B7 D0E0	4821	SAL AL,1
F6B9 D0E0	4822	SAL AL,1 ; LEFT SHIFT BY 2
F6BB 0AD8	4823	OR BL,AL ; ANOTHER COLOR VERSION INTO BL
F6BD E2F8	4824	LOOP S20 ; FILL ALL OF BL
F6BF 8AFB	4825	MOV BH,BL ; FILL UPPER PORTION
F6C1 59	4826	POP CX ; REGISTER BACK
F6C2 C3	4827	RET ; ALL DONE
	4828	S19 ENDP
	4829	;-----
	4830	; EXPAND_BYTE ;
	4831	; THIS ROUTINE TAKES THE BYTE IN AL AND DOUBLES ;
	4832	; ALL OF THE BITS, TURNING THE 8 BITS INTO ;
	4833	; 16 BITS. THE RESULT IS LEFT IN AX ;
	4834	;-----
F6C3	4835	S21 PROC NEAR
F6C3 52	4836	PUSH DX ; SAVE REGISTERS
F6C4 51	4837	PUSH CX
F6C5 53	4838	PUSH BX
F6C6 2BD2	4839	SUB DX,DX ; RESULT REGISTER
F6C8 B90100	4840	MOV CX,1 ; MASK REGISTER
F6CB	4841	S22:
F6CB 8BD8	4842	MOV BX,AX ; BASE INTO TEMP
F6CD 23D9	4843	AND BX,CX ; USE MASK TO EXTRACT A BIT
F6CF 0BD3	4844	OR DX,BX ; PUT INTO RESULT REGISTER
F6D1 D1E0	4845	SHL AX,1
F6D3 D1E1	4846	SHL CX,1 ; SHIFT BASE AND MASK BY 1
F6D5 8BD8	4847	MOV BX,AX ; BASE TO TEMP
F6D7 23D9	4848	AND BX,CX ; EXTRACT THE SAME BIT
F6D9 0BD3	4849	OR DX,BX ; PUT INTO RESULT
F6DB D1E1	4850	SHL CX,1 ; SHIFT ONLY MASK NOW,
	4851	; MOVING TO NEXT BASE
F6DD 73EC	4852	JNC S22 ; USE MASK BIT COMING OUT TO TERMINATE
F6DF 8BC2	4853	MOV AX,DX ; RESULT TO PARM REGISTER
F6E1 5B	4854	POP BX
F6E2 59	4855	POP CX ; RECOVER REGISTERS
F6E3 5A	4856	POP DX
F6E4 C3	4857	RET ; ALL DONE
	4858	S21 ENDP
	4859	;-----
	4860	; HED_READ_BYTE ;
	4861	; THIS ROUTINE WILL TAKE 2 BYTES FROM THE REGEN ;
	4862	; BUFFER, COMPARE AGAINST THE CURRENT FOREGROUND ;
	4863	; COLOR, AND PLACE THE CORRESPONDING ON/OFF BIT ;
	4864	; PATTERN INTO THE CURRENT POSITION IN THE SAVE ;
	4865	; AREA ;
	4866	; ENTRY ;
	4867	; SI,DS = POINTER TO REGEN AREA OF INTEREST ;
	4868	; BX = EXPANDED FOREGROUND COLOR ;
	4869	; BP = POINTER TO SAVE AREA ;
	4870	; EXIT ;
	4871	; BP IS INCREMENT AFTER SAVE ;
	4872	;-----

LOC OBJ	LINE	SOURCE
F6E5	4873	S23 PROC NEAR
F6E5 8A24	4874	MOV AH,[SI]
F6E7 8A4401	4875	MOV AL,[SI+1]
F6EA B900C0	4876	MOV CX,0C000H
F6ED B200	4877	MOV DL,0
F6EF	4878	S24: ; RESULT REGISTER
F6EF 85C1	4879	TEST AX,CX
F6F1 F8	4880	CLC
F6F2 7401	4881	JZ S25
F6F4 F9	4882	STC
F6F5 D0D2	4883	S25: RCL DL,1
F6F7 D1E9	4884	SHR CX,1
F6F9 D1E9	4885	SHR CX,1
F6FB 73F2	4886	JNC S24
F6FD 885600	4887	MOV [BP],DL
F700 45	4888	INC BP
F701 C3	4889	RET
	4890	S23 ENDP
	4891	-----
	4892	; V4_POSITION :
	4893	; THIS ROUTINE TAKES THE CURSOR POSITION :
	4894	; CONTAINED IN THE MEMORY LOCATION, AND :
	4895	; CONVERTS IT INTO AN OFFSET INTO THE :
	4896	; REGEN BUFFER, ASSUMING ONE BYTE/CHAR. :
	4897	; FOR MEDIUM RESOLUTION GRAPHICS, :
	4898	; THE NUMBER MUST BE DOUBLED. :
	4899	; ENTRY :
	4900	; NO REGISTERS, MEMORY LOCATION :
	4901	; CURSOR_POSN IS USED :
	4902	; EXIT :
	4903	; AX CONTAINS OFFSET INTO REGEN BUFFER :
	4904	-----
F702	4905	S26 PROC NEAR
F702 A15000	4906	MOV AX,CURSOR_POSN
F705	4907	GRAPH_POSN LABEL NEAR
F705 53	4908	PUSH BX
F706 8BD8	4909	MOV BX,AX
F708 8AC4	4910	MOV AL,AH
F70A F626A00	4911	MUL BYTE PTR CRT_COLS
F70E D1E0	4912	SHL AX,1
F710 D1E0	4913	SHL AX,1
F712 2AFF	4914	SUB BH,BH
F714 03C3	4915	ADD AX,BX
F716 5B	4916	POP BX
F717 C3	4917	RET
	4918	S26 ENDP
	4919	-----
	4920	; WRITE_TTY :
	4921	; THIS INTERFACE PROVIDES A TELETYPE LIKE INTERFACE TO THE VIDEO :
	4922	; CARD. THE INPUT CHARACTER IS WRITTEN TO THE CURRENT CURSOR :
	4923	; POSITION, AND THE CURSOR IS MOVED TO THE NEXT POSITION. IF THE :
	4924	; CURSOR LEAVES THE LAST COLUMN OF THE FIELD, THE COLUMN IS SET :
	4925	; TO ZERO, AND THE ROW VALUE IS INCREMENTED. IF THE ROW VALUE :
	4926	; LEAVES THE FIELD, THE CURSOR IS PLACED ON THE LAST ROW, FIRST :
	4927	; COLUMN, AND THE ENTIRE SCREEN IS SCROLLED UP ONE LINE. WHEN :
	4928	; THE SCREEN IS SCROLLED UP, THE ATTRIBUTE FOR FILLING THE NEWLY :
	4929	; BLANKED LINE IS READ FROM THE CURSOR POSITION ON THE PREVIOUS :
	4930	; LINE BEFORE THE SCROLL, IN CHARACTER MODE. IN GRAPHICS MODE, :
	4931	; THE 0 COLOR IS USED. :
	4932	; ENTRY :
	4933	; (AH) = CURRENT CRT MODE :
	4934	; (AL) = CHARACTER TO BE WRITTEN :
	4935	; NOTE THAT BACK SPACE, CAR RET, BELL AND LINE FEED ARE HANDLED :
	4936	; AS COMMANDS RATHER THAN AS DISPLAYABLE GRAPHICS :
	4937	; (BL) = FOREGROUND COLOR FOR CHAR WRITE IF CURRENTLY IN A :
	4938	; GRAPHICS MODE :
	4939	; EXIT :
	4940	; ALL REGISTERS SAVED :
	4941	-----
	4942	ASSUME CS:CODE,DS:DATA
F718	4943	WRITE_TTY PROC NEAR
F718 50	4944	PUSH AX
F719 50	4945	PUSH AX
F71A B403	4946	MOV AH,3
F71C 8A3E6200	4947	MOV BH,ACTIVE_PAGE
F720 CD10	4948	INT 10H
F722 5B	4949	POP AX

LOC OBJ

LINE SOURCE

```

4950
4951 ;----- DX NOW HAS THE CURRENT CURSOR POSITION
4952
F723 3C08      4953      CMP     AL,8           ; IS IT A BACKSPACE
F725 7452      4954      JE      U8           ; BACK_SPACE
F727 3C0D      4955      CMP     AL,0DH        ; IS IT CARRIAGE RETURN
F729 7457      4956      JE      U9           ; CAR_RET
F72B 3C0A      4957      CMP     AL,0AH        ; IS IT A LINE FEED
F72D 7457      4958      JE      U10          ; LINE_FEED
F72F 3C07      4959      CMP     AL,07H        ; IS IT A BELL
F731 745A      4960      JE      U11          ; BELL
4961
4962 ;----- WRITE THE CHAR TO THE SCREEN
4963
4964
F733 B40A      4965      MOV     AH,10          ; WRITE CHAR ONLY
F735 B90100     4966      MOV     CX,1           ; ONLY ONE CHAR
F73B CD10      4967      INT     10H          ; WRITE THE CHAR
4968
4969 ;----- POSITION THE CURSOR FOR NEXT CHAR
4970
F73A FEC2      4971      INC     DL
F73C 3A164A00   4972      CMP     DL,BYTE PTR CRT_COLS ; TEST FOR COLUMN OVERFLOW
F740 7533      4973      JNZ     U7           ; SET_CURSOR
F742 B200      4974      MOV     DL,0           ; COLUMN FOR CURSOR
F744 80FE18     4975      CMP     DH,24          ;
F747 752A      4976      JNZ     U6           ; SET_CURSOR_INC
4977
4978 ;----- SCROLL REQUIRED
4979
F749           4980      UI:
F749 B402      4981      MOV     AH,2
F74B CD10      4982      INT     10H          ; SET THE CURSOR
4983
4984 ;----- DETERMINE VALUE TO FILL WITH DURING SCROLL
4985
F74D A04900     4986      MOV     AL,CRT_MODE        ; GET THE CURRENT MODE
F750 3C04      4987      CMP     AL,4
F752 7206      4988      JC      U2           ; READ-CURSOR
F754 3C07      4989      CMP     AL,7
F756 B700      4990      MOV     BH,0           ; FILL WITH BACKGROUND
F758 7506      4991      JNE     U3           ; SCROLL-UP
F75A           4992      U2:
F75A B408      4993      MOV     AH,8
F75C CD10      4994      INT     10H          ; READ CHAR/ATTR AT CURRENT CURSOR
F75E 8AFC      4995      MOV     BH,AH        ; STORE IN BH
F760           4996      U3:
F760 B80106     4997      MOV     AX,601H        ; SCROLL ONE LINE
F763 2BC9      4998      SUB     CX,CX          ; UPPER LEFT CORNER
F765 B618      4999      MOV     DH,24          ; LOWER RIGHT ROW
F767 8A164A00   5000      MOV     DL,BYTE PTR CRT_COLS ; LOWER RIGHT COLUMN
F76B FECA      5001      DEC     DL
F76D           5002      U4:
F76D CD10      5003      INT     10H          ; VIDEO-CALL-RETURN
F76F           5004      U5:
F76F 58         5005      POP     AX           ; SCROLL UP THE SCREEN
F770 E952FA     5006      JMP     VIDEO_RETURN ; TTY-RETURN
F773           5007      U6:
F773 FEC6      5008      INC     DH           ; RESTORE THE CHARACTER
F775           5009      U7:
F775 B402      5010      MOV     AH,2           ; RETURN TO CALLER
F777 EBF4      5011      JMP     U4           ; SET-CURSOR-INC
5012           ; NEXT ROW
5013           ; SET-CURSOR
5014           ; ESTABLISH THE NEW CURSOR
5015
5016 ;----- BACK SPACE FOUND
5017
F779           5018      U8:
F779 80FA00     5019      CMP     DL,0           ; ALREADY AT END OF LINE
F77C 74F7      5020      JE      U7           ; SET_CURSOR
F77E FECA      5021      DEC     DL           ; NO -- JUST MOVE IT BACK
F780 EBF3      5022      JMP     U7           ; SET_CURSOR
5023
5024 ;----- CARRIAGE RETURN FOUND
5025
F782           5026      U9:
F782 B200      5027      MOV     DL,0           ; MOVE TO FIRST COLUMN
F784 EBEF      5028      JMP     U7           ; SET_CURSOR
5029

```

LOC OBJ	LINE	SOURCE
	5027	;----- LINE FEED FOUND
	5028	
F786	5029	U10:
F786 80FE18	5030	CMP DH,24 ; BOTTOM OF SCREEN
F789 75E8	5031	JNE U6 ; YES, SCROLL THE SCREEN
F78B EBBC	5032	JMP U1 ; NO, JUST SET THE CURSOR
	5033	
	5034	;----- BELL FOUND
	5035	
F78D	5036	U11:
F78D B302	5037	MOV BL,2 ; SET UP COUNT FOR BEEP
F78F E871EE	5038	CALL BEEP ; SOUND THE POO BELL
F792 EBDB	5039	JMP U5 ; TTY_RETURN
	5040	WRITE_TTY ENDP
	5041	;-----
	5042	; LIGHT PEN
	5043	; THIS ROUTINE TESTS THE LIGHT PEN SWITCH AND THE LIGHT
	5044	; PEN TRIGGER. IF BOTH ARE SET, THE LOCATION OF THE LIGHT
	5045	; PEN IS DETERMINED, OTHERWISE, A RETURN WITH NO
	5046	; INFORMATION IS MADE.
	5047	; ON EXIT
	5048	; (AH) = 0 IF NO LIGHT PEN INFORMATION IS AVAILABLE
	5049	; BX,CX,DX ARE DESTROYED
	5050	; (AH) = 1 IF LIGHT PEN IS AVAILABLE
	5051	; (DH,DL) = ROW,COLUMN OF CURRENT LIGHT PEN
	5052	; POSITION
	5053	; (CH) = RASTER POSITION
	5054	; (BX) = BEST GUESS AT PIXEL HORIZONTAL POSITION
	5055	;-----
	5056	ASSUME CS:CODE,DS:DATA
	5057	;----- SUBTRACT_TABLE
F794	5058	V1 LABEL BYTE
F794 03	5059	DB 3,3,5,5,3,3,3,4 ;
F795 03		
F796 05		
F797 05		
F798 03		
F799 03		
F79A 03		
F79B 04		
F79C		
	5060	READ_LPEN PROC NEAR
	5061	
	5062	;----- WAIT FOR LIGHT PEN TO BE DEPRESSED
	5063	
F79C B400	5064	MOV AH,0 ; SET NO LIGHT PEN RETURN CODE
F79E 8B166300	5065	MOV DX,ADDR_6845 ; GET BASE ADDRESS OF 6845
F7A2 83C206	5066	ADD DX,6 ; POINT TO STATUS REGISTER
F7A5 EC	5067	IN AL,DX ; GET STATUS REGISTER
F7A6 A804	5068	TEST AL,4 ; TEST LIGHT PEN SWITCH
F7A8 757E	5069	JNZ V6 ; NOT SET, RETURN
	5070	
	5071	;----- NOW TEST FOR LIGHT PEN TRIGGER
	5072	
F7AA A802	5073	TEST AL,2 ; TEST LIGHT PEN TRIGGER
F7AC 7503	5074	JNZ V7A ; RETURN WITHOUT RESETING TRIGGER
F7AE E98100	5075	JMP V7
	5076	
	5077	;----- TRIGGER HAS BEEN SET, READ THE VALUE IN
	5078	
F7B1	5079	V7A:
F7B1 B410	5080	MOV AH,16 ; LIGHT PEN REGISTERS ON 6845
	5081	
	5082	;----- INPUT REGS POINTED TO BY AH, AND CONVERT TO ROW COLUMN IN DX
	5083	
F7B3 8B166300	5084	MOV DX,ADDR_6845 ; ADDRESS REGISTER FOR 6845
F7B7 8AC4	5085	MOV AL,AH ; REGISTER TO READ
F7B9 EE	5086	OUT DX,AL ; SET IT UP
F7BA 42	5087	INC DX ; DATA REGISTER
F7BB EC	5088	IN AL,DX ; GET THE VALUE
F7BC 8AE8	5089	MOV CH,AL ; SAVE IN CX
F7BE 4A	5090	DEC DX ; ADDRESS REGISTER
F7BF FEC4	5091	INC AH
F7C1 8AC4	5092	MOV AL,AH ; SECOND DATA REGISTER
F7C3 EE	5093	OUT DX,AL
F7C4 42	5094	INC DX ; POINT TO DATA REGISTER
F7C5 EC	5095	IN AL,DX ; GET SECOND DATA VALUE
F7C6 8AE5	5096	MOV AH,CH ; AX HAS INPUT VALUE

LOC OBJ

LINE SOURCE

```

5097
5098 ;----- AX HAS THE VALUE READ IN FROM THE 6845
5099
F7C8 8A1E4900 5100     MOV     BL,CRT_MODE
F7CC 2AFF      5101     SUB     BH,BH                ; MODE VALUE TO BX
F7CE 2E8A9F94F7 5102     MOV     BL,CS:V1[BX]          ; DETERMINE AMOUNT TO SUBTRACT
F7D3 2BC3      5103     SUB     AX,BX                ; TAKE IT AWAY
F7D5 8B1E4E00 5104     MOV     BX,CRT_START
F7D9 D1EB      5105     SHR     BX,1
F7DB 2BC3      5106     SUB     AX,BX
F7DD 7902      5107     JNS     V2                ; IF POSITIVE, DETERMINE MODE
F7DF 2BC0      5108     SUB     AX,AX                ; <0 PLAYS AS 0
5109
5110 ;----- DETERMINE MODE OF OPERATION
5111
F7E1          5112     V2:                ; DETERMINE_MODE
F7E1 B103      5113     MOV     CL,3                ; SET #8 SHIFT COUNT
F7E3 803E490004 5114     CMP     CRT_MODE,4          ; DETERMINE IF GRAPHICS OR ALPHA
F7E8 722A      5115     JB      V4                ; ALPHA_PEN
F7EA 803E490007 5116     CMP     CRT_MODE,7
F7EF 7423      5117     JE      V4                ; ALPHA_PEN
5118
5119 ;----- GRAPHICS MODE
5120
F7F1 B228      5121     MOV     DL,40                ; DIVISOR FOR GRAPHICS
F7F3 F6F2      5122     DIV     DL                ; DETERMINE ROW(AL) AND COLUMN(AH)
5123                ; AL RANGE 0-99, AH RANGE 0-39
5124
5125 ;----- DETERMINE GRAPHIC ROM POSITION
5126
F7F5 8AE8      5127     MOV     CH,AL                ; SAVE ROW VALUE IN CH
F7F7 02ED      5128     ADD     CH,CH                ; *2 FOR EVEN/ODD FIELD
F7F9 8ADC      5129     MOV     BL,AH                ; COLUMN VALUE TO BX
F7FB 2AFF      5130     SUB     BH,BH                ; MULTIPLY BY 8 FOR MEDIUM RES
F7FD 803E490006 5131     CMP     CRT_MODE,6          ; DETERMINE MEDIUM OR HIGH RES
F802 7504      5132     JNE     V3                ; NOT_HIGH_RES
F804 B104      5133     MOV     CL,4                ; SHIFT VALUE FOR HIGH RES
F806 D0E4      5134     SAL     AH,1                ; COLUMN VALUE TIMES 2 FOR HIGH RES
F808          5135     V3:                ; NOT_HIGH_RES
F808 D3E3      5136     SHL     BX,CL                ; MULTIPLY *16 FOR HIGH RES
5137
5138 ;----- DETERMINE ALPHA CHAR POSITION
5139
F80A 8AD4      5140     MOV     DL,AH                ; COLUMN VALUE FOR RETURN
F80C 8AF0      5141     MOV     DH,AL                ; ROW VALUE
F80E D0EE      5142     SHR     DH,1                ; DIVIDE BY 4
F810 D0EE      5143     SHR     DH,1                ; FOR VALUE IN 0-24 RANGE
F812 EB12      5144     JMP     SHORT V5                ; LIGHT_PEN_RETURN_SET
5145
5146 ;----- ALPHA MODE ON LIGHT PEN
5147
F814          5148     V4:                ; ALPHA_PEN
F814 F6364A00 5149     DIV     BYTE PTR CRT_COLS          ; DETERMINE ROW,COLUMN VALUE
F818 8AF0      5150     MOV     DH,AL                ; ROWS TO DH
F81A 8AD4      5151     MOV     DL,AH                ; COLS TO DL
F81C D2E0      5152     SAL     AL,CL                ; MULTIPLY ROWS * 8
F81E 8AE8      5153     MOV     CH,AL                ; GET RASTER VALUE TO RETURN REG
F820 8ADC      5154     MOV     BL,AH                ; COLUMN VALUE
F822 32FF      5155     XOR     BH,BH                ; TO BX
F824 D3E3      5156     SAL     BX,CL
F826          5157     V5:                ; LIGHT_PEN_RETURN_SET
F826 B401      5158     MOV     AH,1                ; INDICATE EVERTHING SET
F828          5159     V6:                ; LIGHT_PEN_RETURN
F828 52        5160     PUSH    DX                ; SAVE RETURN VALUE (IN CASE)
F829 8B166300 5161     MOV     DX,ADDR_6845          ; GET BASE ADDRESS
F82D 83C207    5162     ADD     DX,7                ; POINT TO RESET PARM
F830 EE        5163     OUT     DX,AL                ; ADDRESS, NOT DATA, IS IMPORTANT
F831 5A        5164     POP     DX                ; RECOVER VALUE
F832          5165     V7:                ; RETURN_NO_RESET
F832 5F        5166     POP     DI
F833 5E        5167     POP     SI
F834 1F        5168     POP     DS                ; DISCARD SAVED BX,CX,DX
F835 1F        5169     POP     DS
F836 1F        5170     POP     DS
5171
F837 1F        5172     POP     DS
F838 07        5173     POP     ES

```

F839 CF

F841

F841

F841 FB

F842 1E

F843 E8F806

F846 A11300

F849 1F

F84A CF

F84D

F84D

F84D FB

F84E 1E

F84F E8EC06

F852 A11000

F855 1F

F856 CF

```

5174             IRET
5175 READ_LPEN      ENDP
5176
5177 ;--- INT 12 -----
5178 ; MEMORY_SIZE_DET
5179 ; THIS ROUTINE DETERMINES THE AMOUNT OF MEMORY IN THE SYSTEM
5180 ; AS REPRESENTED BY THE SWITCHES ON THE PLANAR. NOTE THAT THE
5181 ; SYSTEM MAY NOT BE ABLE TO USE I/O MEMORY UNLESS THERE IS A FULL
5182 ; COMPLEMENT OF 64K BYTES ON THE PLANAR.
5183 ; INPUT
5184 ; NO REGISTERS
5185 ; THE MEMORY_SIZE VARIABLE IS SET DURING POWER ON DIAGNOSTICS
5186 ; ACCORDING TO THE FOLLOWING HARDWARE ASSUMPTIONS:
5187 ; PORT 60 BITS 3,2 = 00 - 16K BASE RAM
5188 ;                   01 - 32K BASE RAM
5189 ;                   10 - 48K BASE RAM
5190 ;                   11 - 64K BASE RAM
5191 ; PORT 62 BITS 3-0 INDICATE AMOUNT OF I/O RAM IN 32K INCREMENTS
5192 ; E.G., 0000 - NO RAM IN I/O CHANNEL
5193 ;                   0010 - 64K RAM IN I/O CHANNEL, ETC.
5194 ; OUTPUT
5195 ; (AX) = NUMBER OF CONTIGUOUS 1K BLOCKS OF MEMORY
5196 ;-----
5197 ASSUME CS:CODE,DS:DATA
5198 ORG 0F841H
5199 MEMORY_SIZE_DET PROC FAR
5200             STI                     ; INTERRUPTS BACK ON
5201             PUSH DS                 ; SAVE SEGMENT
5202             CALL DDS
5203             MOV AX,MEMORY_SIZE      ; GET VALUE
5204             POP DS                  ; RECOVER SEGMENT
5205             IRET                    ; RETURN TO CALLER
5206 MEMORY_SIZE_DET ENDP
5207
5208 ;--- INT 11 -----
5209 ; EQUIPMENT DETERMINATION
5210 ; THIS ROUTINE ATTEMPTS TO DETERMINE WHAT OPTIONAL
5211 ; DEVICES ARE ATTACHED TO THE SYSTEM.
5212 ; INPUT
5213 ; NO REGISTERS
5214 ; THE EQUIP_FLAG VARIABLE IS SET DURING THE POWER ON
5215 ; DIAGNOSTICS USING THE FOLLOWING HARDWARE ASSUMPTIONS:
5216 ; PORT 60 = LOW ORDER BYTE OF EQUIPMENT
5217 ; PORT 3FA = INTERRUPT ID REGISTER OF 8250
5218 ; BITS 7-3 ARE ALWAYS 0
5219 ; PORT 376 = OUTPUT PORT OF PRINTER -- 8255 PORT THAT
5220 ; CAN BE READ AS WELL AS WRITTEN
5221 ; OUTPUT
5222 ; (AX) IS SET, BIT SIGNIFICANT, TO INDICATE ATTACHED I/O
5223 ; BIT 15,14 = NUMBER OF PRINTERS ATTACHED
5224 ; BIT 13 NOT USED
5225 ; BIT 12 = GAME I/O ATTACHED
5226 ; BIT 11,10,9 = NUMBER OF RS232 CARDS ATTACHED
5227 ; BIT 8 UNUSED
5228 ; BIT 7,6 = NUMBER OF DISKETTE DRIVES
5229 ;                   00=1, 01=2, 10=3, 11=4 ONLY IF BIT 0 = 1
5230 ; BIT 5,4 = INITIAL VIDEO MODE
5231 ;                   00 - UNUSED
5232 ;                   01 - 40X25 BW USING COLOR CARD
5233 ;                   10 - 80X25 BW USING COLOR CARD
5234 ;                   11 - 80X25 BW USING BW CARD
5235 ; BIT 3,2 = PLANAR RAM SIZE (00=16K,01=32K,10=48K,11=64K)
5236 ; BIT 1 NOT USED
5237 ; BIT 0 = IPL FROM DISKETTE -- THIS BIT INDICATES THAT
5238 ; THERE ARE DISKETTE DRIVES ON THE SYSTEM
5239 ;
5240 ; NO OTHER REGISTERS AFFECTED
5241 ;-----
5242 ASSUME CS:CODE,DS:DATA
5243 ORG 0F84DH
5244 EQUIPMENT      PROC FAR
5245             STI                     ; INTERRUPTS BACK ON
5246             PUSH DS                 ; SAVE SEGMENT REGISTER
5247             CALL DDS
5248             MOV AX,EQUIP_FLAG      ; GET THE CURRENT SETTINGS
5249             POP DS                  ; RECOVER SEGMENT
5250             IRET                    ; RETURN TO CALLER

```



```

5251 EQUIPMENT      ENDP
5252
5253 ;--- INT 15 -----
5254 ; CASSETTE I/O :
5255 ; (AH) = 0  TURN CASSETTE MOTOR ON :
5256 ; (AH) = 1  TURN CASSETTE MOTOR OFF :
5257 ; (AH) = 2  READ 1 OR MORE 256 BYTE BLOCKS FROM CASSETTE :
5258 ; (ES,BX) = POINTER TO DATA BUFFER :
5259 ; (CX) = COUNT OF BYTES TO READ :
5260 ; ON EXIT :
5261 ; (ES,BX) = POINTER TO LAST BYTE READ + 1 :
5262 ; (DX) = COUNT OF BYTES ACTUALLY READ :
5263 ; (CY) = 0 IF NO ERROR OCCURRED :
5264 ; = 1 IF ERROR OCCURRED :
5265 ; (AH) = ERROR RETURN IF (CY)= 1 :
5266 ; = 01 IF CRC ERROR WAS DETECTED :
5267 ; = 02 IF DATA TRANSITIONS ARE LOST :
5268 ; = 04 IF NO DATA WAS FOUND :
5269 ; (AH) = 3 WRITE 1 OR MORE 256 BYTE BLOCKS TO CASSETTE :
5270 ; (ES,BX) = POINTER TO DATA BUFFER :
5271 ; (CX) = COUNT OF BYTES TO WRITE :
5272 ; ON EXIT :
5273 ; (EX,BX) = POINTER TO LAST BYTE WRITTEN + 1 :
5274 ; (CX) = 0 :
5275 ; (AH) = ANY OTHER THAN ABOVE VALUES CAUSES (CY)= 1 :
5276 ; AND (AH)= 80 TO BE RETURNED (INVALID COMMAND). :
5277 ;-----
5278 ASSUME DS:DATA,ES:NOTHING,SS:NOTHING,CS:CODE
5279 ORG 0F859H
5280 CASSETTE_IO PROC FAR
5281 STI ; INTERRUPTS BACK ON
5282 PUSH DS ; ESTABLISH ADDRESSING TO DATA
5283 CALL DDS
5284 AND BIOS_BREAK, 7FH ; MAKE SURE BREAK FLAG IS OFF
5285 CALL W1 ; CASSETTE_IO_CONT
5286 POP DS
5287 RET 2 ; INTERRUPT RETURN
5288 CASSETTE_IO ENDP
5289 W1 PROC NEAR
5290 ;-----
5291 ; PURPOSE: :
5292 ; TO CALL APPROPRIATE ROUTINE DEPENDING ON REG AH :
5293 ; :
5294 ; AH ROUTINE :
5295 ;-----
5296 ; 0 MOTOR ON :
5297 ; 1 MOTOR OFF :
5298 ; 2 READ CASSETTE BLOCK :
5299 ; 3 WRITE CASSETTE BLOCK :
5300 ;-----
5301 OR AH,AH ; TURN ON MOTOR?
5302 JZ MOTOR_ON ; YES, DO IT
5303 DEC AH ; TURN OFF MOTOR?
5304 JZ MOTOR_OFF ; YES, DO IT
5305 DEC AH ; READ CASSETTE BLOCK?
5306 JZ READ_BLOCK ; YES, DO IT
5307 DEC AH ; WRITE CASSETTE BLOCK?
5308 JNZ W2 ; NOT_DEFINED
5309 JMP WRITE_BLOCK ; YES, DO IT
5310 W2: ; COMMAND NOT DEFINED
5311 MOV AH,080H ; ERROR, UNDEFINED OPERATION
5312 STC ; ERROR FLAG
5313 RET
5314 W1 ENDP
5315 MOTOR_ON PROC NEAR
5316 ;-----
5317 ; PURPOSE: :
5318 ; TO TURN ON CASSETTE MOTOR :
5319 ;-----
5320 IN AL,PORT_B ; READ CASSETTE OUTPUT
5321 AND AL,NOT_08H ; CLEAR BIT TO TURN ON MOTOR
5322 W3:
5323 OUT PORT_B,AL ; WRITE IT OUT
5324 SUB AH,AH ; CLEAR AH
5325 RET
5326 MOTOR_ON ENDP
5327 MOTOR_OFF PROC NEAR

```

```

5328 ;-----
5329 ; PURPOSE:
5330 ; TO TURN CASSETTE MOTOR OFF
5331 ;-----
F88A E461 5332 IN AL,PORT_B ; READ CASSETTE OUTPUT
F88C 0C08 5333 OR AL,08H ; SET BIT TO TURN OFF
F88E EBF5 5334 JMP W3 ; WRITE IT, CLEAR ERROR, RETURN
5335 MOTOR_OFF ENDP
F890 5336 READ_BLOCK PROC NEAR
5337 ;-----
5338 ; PURPOSE:
5339 ; TO READ 1 OR MORE 256 BYTE BLOCKS FROM CASSETTE
5340 ;
5341 ; ON ENTRY:
5342 ; ES IS SEGMENT FOR MEMORY BUFFER (FOR COMPACT CODE)
5343 ; BX POINTS TO START OF MEMORY BUFFER
5344 ; CX CONTAINS NUMBER OF BYTES TO READ
5345 ; ON EXIT:
5346 ; BX POINTS 1 BYTE PAST LAST BYTE PUT IN MEM
5347 ; CX CONTAINS DECREMENTED BYTE COUNT
5348 ; DX CONTAINS NUMBER OF BYTES ACTUALLY READ
5349 ;
5350 ; CARRY FLAG IS CLEAR IF NO ERROR DETECTED
5351 ; CARRY FLAG IS SET IF CRC ERROR DETECTED
5352 ;-----
F890 53 5353 PUSH BX ; SAVE BX
F891 51 5354 PUSH CX ; SAVE CX
F892 56 5355 PUSH SI ; SAVE SI
F893 BE0700 5356 MOV SI, 7 ; SET UP RETRY COUNT FOR LEADER
F896 E8BF01 5357 CALL BEGIN_OP ; BEGIN BY STARTING MOTOR
F899 5358 W4: ; SEARCH FOR LEADER
F899 E462 5359 IN AL,PORT_C ; GET INITIAL VALUE
F89B 2410 5360 AND AL,010H ; MASK OFF EXTRANEIOUS BITS
F89D A26B00 5361 MOV LAST_VAL,AL ; SAVE IN LOC LAST_VAL
F8A0 BA7A3F 5362 MOV DX,16250 ; # OF TRANSITIONS TO LOOK FOR
F8A3 5363 W5: ; WAIT_FOR_EDGE
F8A3 F606710080 5364 TEST BIOS_BREAK, 80H ; CHECK FOR BREAK KEY
F8A8 7503 5365 JNZ W6A ; JUMP IF NO BREAK KEY
5366 ; JUMP IF BREAK KEY HIT
F8AA 5367 W6:
F8AA 4A 5368 DEC DX
F8AB 7503 5369 JNZ W7 ; JUMP IF BEGINNING OF LEADER
F8AD 5370 W6A:
F8AD E98400 5371 JMP W17 ; JUMP IF NO LEADER FOUND
F8B0 5372 W7:
F8B0 E8C600 5373 CALL READ_HALF_BIT ; IGNORE FIRST EDGE
F8B3 E3EE 5374 JCXZ W5 ; JUMP IF NO EDGE DETECTED
F8B5 BA7803 5375 MOV DX,0378H ; CHECK FOR HALF BITS
F8B8 B90002 5376 MOV CX,200H ; MUST HAVE AT LEAST THIS MANY ONE SIZE
5377 ; PULSES BEFORE CHCKNG FOR SYNC BIT (0)
F8BB E421 5378 IN AL, 021H ; INTERRUPT MASK REGISTER
F8BD 0C01 5379 OR AL,1 ; DISABLE TIMER INTERRUPTS
F8BF E621 5380 OUT 021H, AL
F8C1 5381 W8: ; SEARCH-LDR
F8C1 F606710080 5382 TEST BIOS_BREAK, 80H ; CHECK FOR BREAK KEY
F8C6 756C 5383 JNZ W17 ; JUMP IF BREAK KEY HIT
F8C8 51 5384 PUSH CX ; SAVE REG CX
F8C9 E8AD00 5385 CALL READ_HALF_BIT ; GET PULSE WIDTH
F8CC 0BC9 5386 OR CX, CX ; CHECK FOR TRANSITION
F8CE 59 5387 POP CX ; RESTORE ONE BIT COUNTER
F8CF 74C8 5388 JZ W4 ; JUMP IF NO TRANSITION
F8D1 3B03 5389 CMP DX,BX ; CHECK PULSE WIDTH
F8D3 E304 5390 JCXZ W9 ; IF CX=0 THEN WE CAN LOOK
5391 ; FOR SYNC BIT (0)
F8D5 73C2 5392 JNC W4 ; JUMP IF ZERO BIT (NOT GOOD LEADER)
F8D7 E2E8 5393 LOOP W8 ; DEC CX AND READ ANOTHER HALF ONE BIT
F8D9 5394 W9: ; FIND-SYNC
F8D9 72E6 5395 JC W8 ; JUMP IF ONE BIT (STILL LEADER)
5396
5397 ;----- A SYNCH BIT HAS BEEN FOUND. READ SYN CHARACTER:
5398
F8DB E69B00 5399 CALL READ_HALF_BIT ; SKIP OTHER HALF OF SYNC BIT (0)
F8DE E8AA00 5400 CALL READ_BYTE ; READ SYN BYTE
F8E1 3C16 5401 CMP AL, 16H ; SYNCHRONIZATION CHARACTER
F8E3 7549 5402 JNE W16 ; JUMP IF BAD LEADER FOUND.
5403
5404 ;----- GOOD CRC SO READ DATA BLOCK(S)

```

LOC OBJ

LINE

SOURCE

```

5405
F8E5 5E      5406      POP      SI          ; RESTORE REGS
F8E6 59      5407      POP      CX
F8E7 5B      5408      POP      BX
5409
5410      ; READ 1 OR MORE 256 BYTE BLOCKS FROM CASSETTE      :
5411      ;                                                    :
5412      ; ON ENTRY:                                          :
5413      ;     ES IS SEGMENT FOR MEMORY BUFFER (FOR COMPACT CODE) :
5414      ;     BX POINTS TO START OF MEMORY BUFFER             :
5415      ;     CX CONTAINS NUMBER OF BYTES TO READ             :
5416      ; ON EXIT:                                           :
5417      ;     BX POINTS 1 BYTE PAST LAST BYTE PUT IN MEM     :
5418      ;     CX CONTAINS DECREMENTED BYTE COUNT             :
5419      ;     DX CONTAINS NUMBER OF BYTES ACTUALLY READ       :
5420      ;-----
F8E8 51      5421      PUSH     CX          ; SAVE BYTE COUNT
F8E9          5422      W10:          ; COME HERE BEFORE EACH
5423          ; 256 BYTE BLOCK IS READ
F8E9 C7066900FFFF 5424      MOV      CRC_REG,0FFFFH ; INIT CRC REG
F8EF BA0001    5425      MOV      DX,256    ; SET DX TO DATA BLOCK SIZE
F8F2          5426      W11:          ; RD_BLK
F8F2 F606710080 5427      TEST     BIOS_BREAK, 80H ; CHECK FOR BREAK KEY
F8F7 7523      5428      JNZ      W13          ; JUMP IF BREAK KEY HIT
F8F9 E84F00    5429      CALL     READ_BYTE ; READ BYTE FROM CASSETTE
F8FC 721E      5430      JC       W13          ; CY SET INDICATES NO DATA TRANSITIONS
F8FE E305      5431      JCXZ     W12          ; IF WE'VE ALREADY REACHED
5432          ; END OF MEMORY BUFFER
5433          ; SKIP REST OF BLOCK
F900 268007    5434      MOV      ES:[BX],AL ; STORE DATA BYTE AT BYTE PTR
F903 43        5435      INC      BX          ; INC BUFFER PTR
F904 49        5436      DEC      CX          ; DEC BYTE COUNTER
F905          5437      W12:          ; LOOP UNTIL DATA BLOCK HAS BEEN
5438          ; READ FROM CASSETTE.
F905 4A        5439      DEC      DX          ; DEC BLOCK CNT
F906 7FEA      5440      JG       W11          ; RD_BLK
F908 E84000    5441      CALL     READ_BYTE ; NOW READ TWO CRC BYTES
F90B E83000    5442      CALL     READ_BYTE
F90E 2AE4      5443      SUB      AH,AH          ; CLEAR AH
F910 813E6900F1D 5444      CMP      CRC_REG,1D0FH ; IS THE CRC CORRECT
F916 7506      5445      JNE      W14          ; IF NOT EQUAL CRC IS BAD
F918 E306      5446      JCXZ     W15          ; IF BYTE COUNT IS ZERO
5447          ; THEN WE HAVE READ ENOUGH
5448          ; SO WE WILL EXIT
F91A EBCD      5449      JMP      W10          ; STILL MORE, SO READ ANOTHER BLOCK
F91C          5450      W13:          ; MISSING-DATA
5451          ; NO DATA TRANSITIONS SO
F91C B401      5452      MOV      AH,01H      ; SET AH=02 TO INDICATE
5453          ; DATA TIMEOUT
F91E          5454      W14:          ; BAD-CRC
F91E FEC4      5455      INC      AH          ; EXIT EARLY ON ERROR
5456          ; SET AH=01 TO INDICATE CRC ERROR
F920          5457      W15:          ; RD-BLK-EX
F920 5A        5458      POP      DX          ; CALCULATE COUNT OF
F921 2B01      5459      SUB      DX,CX      ; DATA BYTES ACTUALLY READ
5460          ; RETURN COUNT IN REG DX
F923 50        5461      PUSH     AX          ; SAVE AX (RET CODE)
F924 F6C490    5462      TEST     AH, 90H      ; CHECK FOR ERRORS
F927 7513      5463      JNZ      W18          ; JUMP IF ERROR DETECTED
F929 E81F00    5464      CALL     READ_BYTE ; READ TRAILER
F92C EB0E      5465      JMP      SHORT W18 ; SKIP TO TURN OFF MOTOR
F92E          5466      W16:          ; BAD-LEADER
F92E 4E        5467      DEC      SI          ; CHECK RETRIES
F92F 7403      5468      JZ       W17          ; JUMP IF TOO MANY RETRIES
F931 E965FF    5469      JMP      W4          ; JUMP IF NOT TOO MANY RETRIES
F934          5470      W17:          ; NO VALID DATA FOUND
5471
5472      ;----- NO DATA FROM CASSETTE ERROR, I.E. TIMEOUT
5473
F934 5E        5474      POP      SI          ; RESTORE REGS
F935 59        5475      POP      CX          ; RESTORE REGS
F936 5B        5476      POP      BX
F937 2B02      5477      SUB      DX,DX      ; ZERO NUMBER OF BYTES READ
F939 B404      5478      MOV      AH,04H      ; TIME OUT ERROR (NO LEADER)
F93B 50        5479      PUSH     AX
F93C          5480      W18:          ; HOT-OFF

```

LOC OBJ	LINE	SOURCE	
F93C E421	5481	IN AL, 021H	; RE_ENABLE INTERRUPTS
F93E 24FE	5482	AND AL, 0FFH- 1	
F940 E621	5483	OUT 021H, AL	
F942 E045FF	5484	CALL MOTOR_OFF	; TURN OFF MOTOR
F945 58	5485	POP AX	; RESTORE RETURN CODE
F946 80FC01	5486	CMF AH,01H	; SET CARRY IF ERROR (AH>0)
F949 F5	5487	CMC	
F94A C3	5488	RET	; FINISHED
	5489	READ_BLOCK ENDP	
	5490	;-----	
	5491	; PURPOSE:	:
	5492	; TO READ A BYTE FROM CASSETTE	:
	5493	; ON EXIT	:
	5494	; REG AL CONTAINS READ DATA BYTE	:
	5495	;-----	
F94B	5496	READ_BYTE PROC NEAR	
F94B 53	5497	PUSH BX	; SAVE REGS BX,CX
F94C 51	5498	PUSH CX	
F94D B108	5499	MOV CL,8H	; SET BIT COUNTER FOR 8 BITS
F94F	5500	W19:	; BYTE-ASH
F94F 51	5501	PUSH CX	; SAVE CX
	5502	;-----	
	5503	; READ DATA BIT FROM CASSETTE	:
	5504	;-----	
F950 E02600	5505	CALL READ_HALF_BIT	; READ ONE PULSE
F953 E320	5506	JCXZ W21	; IF CX=0 THEN TIMEOUT
	5507		; BECAUSE OF NO DATA TRANSITIONS
F955 53	5508	PUSH BX	; SAVE 1ST HALF BIT'S
	5509		; PULSE WIDTH (IN BX)
F956 E02000	5510	CALL READ_HALF_BIT	; READ COMPLEMENTARY PULSE
F959 58	5511	POP AX	; COMPUTE DATA BIT
F95A E319	5512	JCXZ W21	; IF CX=0 THEN TIMEOUT DUE TO
	5513		; NO DATA TRANSITIONS
F95C 0308	5514	ADD BX,AX	; PERIOD
F95E 01FBF006	5515	CMF BX, 06F0H	; CHECK FOR ZERO BIT
F962 F5	5516	CMC	; CARRY IS SET IF ONE BIT
F963 9F	5517	LAHF	; SAVE CARRY IN AH
F964 59	5518	POP CX	; RESTORE CX
	5519		; NOTE:
	5520		; MS BIT OF BYTE IS READ FIRST.
	5521		; REG CH IS SHIFTED LEFT WITH
	5522		; CARRY BEING INSERTED INTO LS
	5523		; BIT OF CH.
	5524		; AFTER ALL 8 BITS HAVE BEEN
	5525		; READ, THE MS BIT OF THE DATA BYTE
	5526		; WILL BE IN THE MS BIT OF REG CH
F965 D0D5	5527	RCL CH,1	; ROTATE REG CH LEFT WITH CARRY TO
	5528		; LS BIT OF REG CH
F967 9E	5529	SAHF	; RESTORE CARRY FOR CRC ROUTINE
F968 E0D900	5530	CALL CRC_GEN	; GENERATE CRC FOR BIT
F96B FEC9	5531	DEC CL	; LOOP TILL ALL 8 BITS OF DATA
	5532		; ASSEMBLED IN REG CH
F96D 75E0	5533	JNZ W19	; BYTE-ASH
F96F 8AC5	5534	MOV AL,CH	; RETURN DATA BYTE IN REG AL
F971 F8	5535	CLC	
F972	5536	W20:	; RD-BYT-EX
F972 59	5537	POP CX	; RESTORE REGS CX,BX
F973 5B	5538	POP BX	
F974 C3	5539	RET	; FINISHED
F975	5540	W21:	; NO-DATA
F975 59	5541	POP CX	; RESTORE CX
F976 F9	5542	STC	; INDICATE ERROR
F977 EBF9	5543	JMP W20	; RD_BYT_EX
	5544	READ_BYTE ENDP	
	5545	;-----	
	5546	; PURPOSE:	:
	5547	; TO COMPUTE TIME TILL NEXT DATA	:
	5548	; TRANSITION (EDGE)	:
	5549	; ON ENTRY:	:
	5550	; EDGE_CNT CONTAINS LAST EDGE COUNT	:
	5551	; ON EXIT:	:
	5552	; AX CONTAINS OLD LAST EDGE COUNT	:
	5553	; BX CONTAINS PULSE WIDTH (HALF BIT)	:
	5554	;-----	
	5555	READ_HALF_BIT PROC NEAR	
F979	5556	MOV CX, 100	; SET TIME TO WAIT FOR BIT
F979 B96400	5557	MOV AH, LAST_VAL	; GET PRESENT INPUT VALUE

LOC	OBJ	LINE	SOURCE
F980		5558	W22:
F980 E462		5559	IN AL,PORT_C ; RD-H-BIT
F982 2410		5560	AND AL,010H ; INPUT DATA BIT
F984 3AC4		5561	CMF AL,AH ; MASK OFF EXTRANEOUS BITS
F986 E1F8		5562	LOOPE W22 ; SAME AS BEFORE?
F988 AC2B00		5563	MOV LAST_VAL,AL ; LOOP TILL IT CHANGES
F98B B000		5564	MOV AL,0 ; UPDATE LAST_VAL WITH NEW VALUE
F98D E643		5565	OUT TIM_CTL,AL ; READ TMR'S COUNTER COMMAND
F98F 8B1E700		5566	MOV BX,EDGE_CNT ; LATCH COUNTER
F993 E440		5567	IN AL,TIMER0 ; BX GETS LAST EDGE COUNT
F995 8AE0		5568	MOV AH,AL ; GET LS BYTE
F997 E440		5569	IN AL,TIMER0 ; SAVE IN AH
F999 86C4		5570	XCHG AL,AH ; GET MS BYTE
F99B 2BD8		5571	SUB BX,AX ; XCHG AL,AH
F99D A36700		5572	MOV EDGE_CNT,AX ; SET BX EQUAL TO HALF BIT PERIOD
F9A0 C3		5573	RET ; UPDATE EDGE COUNT;
		5574	READ_HALF_BIT ENDP
		5575	-----
		5576	; PURPOSE :
		5577	; WRITE 1 OR MORE 256 BYTE BLOCKS TO CASSETTE. :
		5578	; THE DATA IS PADDED TO FILL OUT THE LAST 256 BYTE BLOCK. :
		5579	; ON ENTRY: :
		5580	; BX POINTS TO MEMORY BUFFER ADDRESS :
		5581	; CX CONTAINS NUMBER OF BYTES TO WRITE :
		5582	; ON EXIT: :
		5583	; BX POINTS 1 BYTE PAST LAST BYTE WRITTEN TO CASSETTE :
		5584	; CX IS ZERO :
		5585	-----
F9A1		5586	WRITE_BLOCK PROC NEAR
F9A1 53		5587	PUSH BX
F9A2 51		5588	PUSH CX
F9A3 E461		5589	IN AL,PORT_B ; DISABLE SPEAKER
F9A5 24FD		5590	AND AL,NOT 02H
F9A7 0C01		5591	OR AL,01H ; ENABLE TIMER
F9A9 E661		5592	OUT PORT_B,AL
F9AB B0B6		5593	MOV AL,0B6H ; SET UP TIMER -- MODE 3 SQUARE WAVE
F9AD E643		5594	OUT TIM_CTL,AL
F9AF E8A600		5595	CALL BEGIN_OP ; START MOTOR AND DELAY
F9B2 B8A004		5596	MOV AX,1164 ; SET NORMAL BIT SIZE
F9B5 E88500		5597	CALL W31 ; SET_TIMER
F9B8 B90008		5598	MOV CX,0800H ; SET CX FOR LEADER BYTE COUNT
F9BB		5599	W23: ; WRITE LEADER
F9BB F9		5600	STC ; WRITE ONE BITS
F9BC E86800		5601	CALL WRITE_BIT
F9BF E2FA		5602	LOOP W23 ; LOOP 'TIL LEADER IS WRITTEN
F9C1 F8		5603	CLC ; WRITE SYNC BIT (0)
F9C2 E86200		5604	CALL WRITE_BIT
F9C5 59		5605	POP CX ; RESTORE REGS CX,BX
F9C6 5B		5606	POP BX
F9C7 B016		5607	MOV AL,16H ; WRITE SYN CHARACTER
F9C9 E84400		5608	CALL WRITE_BYTE
		5609	-----
		5610	; PURPOSE :
		5611	; WRITE 1 OR MORE 256 BYTE BLOCKS TO CASSETTE :
		5612	; ON ENTRY: :
		5613	; BX POINTS TO MEMORY BUFFER ADDRESS :
		5614	; CONTAINS NUMBER OF BYTES TO WRITE :
		5615	; ON EXIT: :
		5616	; BX POINTS 1 BYTE PAST LAST BYTE WRITTEN TO CASSETTE :
		5617	; CX IS ZERO :
		5618	-----
F9CC		5619	WR_BLOCK:
F9CC C7066900FFFF		5620	MOV CRC_REG,0FFFFH ; INIT CRC
F9D2 BA0001		5621	MOV DX,256 ; FOR 256 BYTES
F9D5		5622	W24: ; WR-BLK
F9D5 268A07		5623	MOV AL,ES:[BX] ; READ BYTE FROM MEM
F9DB E83500		5624	CALL WRITE_BYTE ; WRITE IT TO CASSETTE
F9DB E302		5625	JCJZ W25 ; UNLESS CX=0, ADVANCE PTRS & DEC COUNT
F9DD 43		5626	INC BX ; INC BUFFER POINTER
F9DE 49		5627	DEC CX ; DEC BYTE COUNTER
F9DF		5628	W25: ; SKIP-ADV
F9DF 4A		5629	DEC DX ; DEC BLOCK CNT
F9E0 7FF3		5630	JG W24 ; LOOP TILL 256 BYTE BLOCK
		5631	; IS WRITTEN TO TAPE
		5632	-----
		5633	; WRITE CRC :
		5634	; WRITE 1'S COMPLEMENT OF CRC REG TO CASSETTE :

```

5635 ; WHICH IS CHECKED FOR CORRECTNESS WHEN THE BLOCK IS READ :
5636 ; REG AX IS MODIFIED :
5637 ;-----
F9E2 A16900 5638 MOV AX,CRC_REG ; WRITE THE ONE'S COMPLEMENT OF THE
5639 ; TWO BYTE CRC TO TAPE
F9E5 F700 5640 NOT AX ; FOR 1'S COMPLEMENT
F9E7 50 5641 PUSH AX ; SAVE IT
F9E8 86E0 5642 XCHG AH,AL ; WRITE MS BYTE FIRST
F9EA E02300 5643 CALL WRITE_BYTE ; WRITE IT
F9ED 58 5644 POP AX ; GET IT BACK
F9EE E81F00 5645 CALL WRITE_BYTE ; NOW WRITE LS BYTE
F9F1 0BC9 5646 OR CX,CX ; IS BYTE COUNT EXHAUSTED?
F9F3 75D7 5647 JNZ WR_BLOCK ; JUMP IF NOT DONE YET
F9F5 51 5648 PUSH CX ; SAVE REG CX
F9F6 B92000 5649 MOV CX, 32 ; WRITE OUT TRAILER BITS
F9F9 5650 W26: ; TRAIL-LOOP
F9F9 F9 5651 STC
F9FA E02A00 5652 CALL WRITE_BIT
F9FD E2FA 5653 LOOP W26 ; WRITE UNTIL TRAILER WRITTEN
F9FF 59 5654 POP CX ; RESTORE REG CX
FA00 B0B0 5655 MOV AL, 0B0H ; TURN TIMER2 OFF
FA02 E643 5656 OUT TIM_CTL, AL
FA04 B00100 5657 MOV AX, 1
FA07 E83300 5658 CALL W31 ; SET_TIMER
FA0A E870FE 5659 CALL MOTOR_OFF ; TURN MOTOR OFF
FA0D 2BC0 5660 SUB AX,AX ; NO ERRORS REPORTED ON WRITE OP
FA0F C3 5661 RET ; FINISHED
5662 WRITE_BLOCK ENDP
5663 ;-----
5664 ; WRITE A BYTE TO CASSETTE. :
5665 ; BYTE TO WRITE IS IN REG AL. :
5666 ;-----
FA10 5667 WRITE_BYTE PROC NEAR
FA10 51 5668 PUSH CX ; SAVE REGS CX,AX
FA11 50 5669 PUSH AX
FA12 8AE8 5670 MOV CH,AL ; AL=BYTE TO WRITE.
5671 ; (MS BIT WRITTEN FIRST)
FA14 B108 5672 MOV CL,8 ; FOR 8 DATA BITS IN BYTE.
5673 ; NOTE: TWO EDGES PER BIT
FA16 5674 W27: ; DISASSEMBLE THE DATA BIT
FA16 D0D5 5675 RCL CH,1 ; ROTATE MS BIT INTO CARRY
FA18 9C 5676 PUSHF ; SAVE FLAGS.
5677 ; NOTE: DATA BIT IS IN CARRY
FA19 E80B00 5678 CALL WRITE_BIT ; WRITE DATA BIT
FA1C 9D 5679 POPF ; RESTORE CARRY FOR CRC CALC
FA1D E82400 5680 CALL CRC_GEN ; COMPUTE CRC ON DATA BIT
FA20 FEC9 5681 DEC CL ; LOOP TILL ALL 8 BITS DONE
FA22 75F2 5682 JNZ W27 ; JUMP IF NOT DONE YET
FA24 58 5683 POP AX ; RESTORE REGS AX,CX
FA25 59 5684 POP CX
FA26 C3 5685 RET ; WE ARE FINISHED
5686 WRITE_BYTE ENDP
5687 ;-----
5688 ; PURPOSE: :
5689 ; TO WRITE A DATA BIT TO CASSETTE :
5690 ; CARRY FLAG CONTAINS DATA BIT :
5691 ; I.E. IF SET DATA BIT IS A ONE :
5692 ; IF CLEAR DATA BIT IS A ZERO :
5693 ; :
5694 ; NOTE: TWO EDGES ARE WRITTEN PER BIT :
5695 ; ONE BIT HAS 500 USEC BETWEEN EDGES :
5696 ; FOR A 1000 USEC PERIOD (1 MILLISEC) :
5697 ; :
5698 ; ZERO BIT HAS 250 USEC BETWEEN EDGES :
5699 ; FOR A 500 USEC PERIOD (.5 MILLISEC) :
5700 ; CARRY FLAG IS DATA BIT :
5701 ;-----
FA27 5702 WRITE_BIT PROC NEAR
5703 5704 MOV AX,1104 ; ASSUME IT'S A '1'
FA27 B8A004 5705 JC W28 ; SET AX TO NOMINAL ONE SIZE
FA2A 7203 5706 MOV AX,592 ; JUMP IF ONE BIT
FA2C B85002 5707 ; NO, SET TO NOMINAL ZERO SIZE
FA2F 5708 W28: ; WRITE-BIT-AX
FA2F 50 5709 PUSH AX ; WRITE BIT WITH PERIOD EQ TO VALUE AX
FA30 5709 W29:
FA30 E462 5710 IN AL,PORT_C ; INPUT TIMER_0 OUTPUT
FA32 2420 5711 AND AL,020H

```

LOC OBJ	LINE	SOURCE
FA34 74FA	5712	JZ W29 ; LOOP TILL HIGH
FA36	5713	
FA36 E462	5714	W30: IN AL,PORT_C ; NOW WAIT TILL TIMER'S OUTPUT IS LOW
FA38 2420	5715	AND AL,020H
FA3A 75FA	5716	JNZ W30
	5717	
	5718	; RELOAD TIMER WITH PERIOD
FA3C 58	5719	POP AX ; FOR NEXT DATA BIT
FA3D	5720	W31: ; RESTORE PERIOD COUNT
FA3D E642	5721	OUT 042H, AL ; SET TIMER
FA3F 8AC4	5722	MOV AL, AH ; SET LOW BYTE OF TIMER 2
FA41 E642	5723	OUT 042H, AL ; SET HIGH BYTE OF TIMER 2
FA43 C3	5724	RET
	5725	WRITE_BIT ENDP
	5726	;
	5727	; UPDATE CRC REGISTER WITH NEXT DATA BIT ;
	5728	; CRC IS USED TO DETECT READ ERRORS ;
	5729	; ASSUMES DATA BIT IS IN CARRY ;
	5730	; ;
	5731	; REG AX IS MODIFIED ;
	5732	; FLAGS ARE MODIFIED ;
	5733	;
FA44	5734	CRC_GEN PROC NEAR
FA44 A16900	5735	MOV AX,CRC_REG
	5736	
	5737	; THE FOLLOWING INSTRUCTIONS
	5738	; WILL SET THE OVERFLOW FLAG
	5739	; IF CARRY AND MS BIT OF CRC
	5740	; ARE UNEQUAL
FA47 D1D8	5740	RCR AX,1
FA49 D1D0	5741	RCL AX,1
FA4B FB	5742	CLC
FA4C 71D4	5743	JND W32 ; CLEAR CARRY
	5744	; SKIP IF NO OVERFLOW
	5745	; IF DATA BIT XORED WITH
	5746	; CRC REG BIT 15 IS ONE
FA4E 351008	5746	XOR AX,0810H ; THEN XOR CRC REG WITH 0801H
FA51 F9	5747	STC ; SET CARRY
FA52	5748	
FA52 D1D0	5749	W32: RCL AX,1 ; ROTATE CARRY (DATA BIT)
	5750	; INTO CRC REG
FA54 A36900	5751	MOV CRC_REG,AX ; UPDATE CRC_REG
FA57 C3	5752	RET ; FINISHED
	5753	CRC_GEN ENDP
	5754	
FA58	5755	BEGIN_OP PROC NEAR
FA58 E82AFE	5756	CALL MOTOR_ON ; START TAPE AND DELAY
FA5B B342	5757	MOV BL,42H ;TURN ON MOTOR
	5758	;DELAY FOR TAPE DRIVE
FA5D	5759	W33: ;TO GET UP TO SPEED (1/2 SEC)
FA5D B90007	5760	MOV CX,700H ;INNER LOOP= APPROX. 10 MILLISEC
FA60 E2FE	5761	W34: LOOP W34
FA62 FECB	5762	DEC BL
FA64 75F7	5763	JNZ W33
FA66 C3	5764	RET
	5765	BEGIN_OP ENDP
	5766	
FA67 20323031	5767	E1 DB ' 201',13,10
FA6B 00		
FA6C 0A		
	5768	
	5769	;
	5770	; CHARACTER GENERATOR GRAPHICS FOR 320X200 AND 640X200 GRAPHICS
	5771	;
FA6E	5772	ORG 0FA6EH
FA6E	5773	CRT_CHAR_GEN LABEL BYTE
FA6E 0000000000000000	5774	DB 000H,000H,000H,000H,000H,000H,000H,000H ; D_00
FA76 7E81A581B099817E	5775	DB 07EH,081H,0A5H,081H,0BDH,099H,081H,07EH ; D_01
FA7E 7EFFDBFFC3E7FF7E	5776	DB 07EH,0FFH,0DBH,0FFH,0C3H,0E7H,0FFH,07EH ; D_02
FA86 6CFE7E7C381000	5777	DB 06CH,0FEH,0FEH,0FEH,07CH,038H,010H,000H ; D_03
FA8E 10387CFE7C381000	5778	DB 010H,038H,07CH,0FEH,07CH,038H,010H,000H ; D_04
FA96 387C38FE7E7C387C	5779	DB 038H,07CH,038H,0FEH,0FEH,07CH,038H,07CH ; D_05
FA9E 1010387CFE7C387C	5780	DB 010H,010H,038H,07CH,0FEH,07CH,038H,07CH ; D_06
FAA6 0000183C3C180000	5781	DB 000H,000H,018H,03CH,03CH,018H,000H,000H ; D_07
FAAE FFFFE7C3C3E7FFFF	5782	DB 0FFH,0FFH,0E7H,0C3H,0C3H,0E7H,0FFH,0FFH ; D_08
FAB6 003C664242663C00	5783	DB 000H,03CH,066H,042H,042H,066H,03CH,000H ; D_09
FABE FFC399BDB099C3FF	5784	DB 0FFH,0C3H,099H,0BDH,0BDH,099H,0C3H,0FFH ; D_0A
FAC6 0F070F7DCCCCC78	5785	DB 00FH,007H,00FH,07DH,0CCH,0CCH,0CCH,078H ; D_0B
FACE 3C6666663C187E18	5786	DB 03CH,066H,066H,066H,03CH,018H,07EH,018H ; D_0C

LOC OBJ	LINE	SOURCE
FAD6 3F333F303070F0E0	5787	DB 03FH, 033H, 03FH, 030H, 030H, 070H, 0F0H, 0E0H ; D_0D
FAD6 7F637F63636766C0	5788	DB 07FH, 063H, 07FH, 063H, 063H, 067H, 0E6H, 0C0H ; D_0E
FAE6 995A3CE773C5A99	5789	DB 099H, 05AH, 03CH, 0E7H, 0E7H, 03CH, 05AH, 099H ; D_0F
FAEE 80E0F8FEF8E08000	5790	DB 080H, 0E0H, 0F8H, 0FEH, 0F8H, 0E0H, 080H, 000H ; D_10
FAF6 020E3FE3E0E0200	5791	DB 002H, 00EH, 03EH, 0FEH, 03EH, 00EH, 002H, 000H ; D_11
FAFE 183C7E18187E3C18	5792	DB 018H, 03CH, 07EH, 018H, 018H, 07EH, 03CH, 018H ; D_12
FB06 6666666666660660	5793	DB 066H, 066H, 066H, 066H, 066H, 000H, 066H, 000H ; D_13
FB0E 7FD0BD7B1B1B1B00	5794	DB 07FH, 00BH, 00BH, 07BH, 01BH, 01BH, 01BH, 000H ; D_14
FB16 3663366C6C38CC78	5795	DB 03EH, 063H, 03BH, 06CH, 06CH, 03BH, 0CCH, 07BH ; D_15
FB1E 000000007E7E7E00	5796	DB 000H, 000H, 000H, 000H, 07EH, 07EH, 07EH, 000H ; D_16
FB26 183C7E187E3C18FF	5797	DB 018H, 03CH, 07EH, 018H, 07EH, 03CH, 018H, 0FFH ; D_17
FB2E 183C7E1818181800	5798	DB 018H, 03CH, 07EH, 018H, 018H, 018H, 018H, 000H ; D_18
FB36 181818187E3C1800	5799	DB 018H, 018H, 018H, 018H, 07EH, 03CH, 018H, 000H ; D_19
FB3E 00180CFE0C180000	5800	DB 000H, 018H, 00CH, 0FEH, 00CH, 018H, 00CH, 000H ; D_1A
FB46 003060FE60300000	5801	DB 000H, 030H, 060H, 0FEH, 060H, 030H, 000H, 000H ; D_1B
FB4E 0000C0C0C0C0FE00	5802	DB 000H, 000H, 0C0H, 0C0H, 0C0H, 0FEH, 000H, 000H ; D_1C
FB56 002466FF66240000	5803	DB 000H, 024H, 066H, 0FFH, 066H, 024H, 000H, 000H ; D_1D
FB5E 00183C7EFFFF0000	5804	DB 000H, 018H, 03CH, 07EH, 0FFH, 0FFH, 000H, 000H ; D_1E
FB66 00FFFF7E3C180000	5805	DB 000H, 0FFH, 0FFH, 07EH, 03CH, 018H, 000H, 000H ; D_1F
FB6E 0000000000000000	5806	DB 000H, 000H, 000H, 000H, 000H, 000H, 00CH, 000H ; SP_D_20
FB76 3078783030003000	5807	DB 030H, 078H, 078H, 030H, 030H, 000H, 030H, 000H ; ? D_21
FB7E 6C6C6C0000000000	5808	DB 06CH, 06CH, 06CH, 000H, 000H, 000H, 000H, 000H ; ? D_22
FB86 6C6CFE6CFE6C6C00	5809	DB 06CH, 06CH, 0FEH, 06CH, 0FEH, 06CH, 06CH, 000H ; ? D_23
FB8E 307CC0780CF83000	5810	DB 030H, 07CH, 0C0H, 078H, 00CH, 0F8H, 030H, 000H ; ? D_24
FB96 00C6CC183066C660	5811	DB 000H, 0C6H, 0CCH, 018H, 030H, 066H, 0C6H, 000H ; PERCENT_D_25
FB9E 386C38760CC76000	5812	DB 038H, 06CH, 038H, 076H, 0DCH, 0CCH, 076H, 000H ; & D_26
FBA6 6060C00000000000	5813	DB 060H, 060H, 0C0H, 000H, 000H, 000H, 000H, 000H ; ? D_27
FBAE 1830606060301800	5814	DB 018H, 030H, 060H, 060H, 060H, 030H, 018H, 000H ; ? D_28
FBBA 6030181818306000	5815	DB 060H, 030H, 018H, 018H, 018H, 030H, 060H, 000H ; ? D_29
FBBE 00663CFF3C660000	5816	DB 000H, 066H, 03CH, 0FFH, 03CH, 066H, 000H, 000H ; ? D_2A
FBCE 003030FC30300000	5817	DB 000H, 030H, 030H, 0FCH, 030H, 030H, 000H, 000H ; + D_2B
FBCE 0000000003030600	5818	DB 000H, 000H, 000H, 000H, 000H, 030H, 030H, 060H ; + D_2C
FBDE 0000000003030000	5819	DB 000H, 000H, 000H, 0FCH, 000H, 000H, 000H, 000H ; + D_2D
FBDE 0000000003030000	5820	DB 000H, 000H, 000H, 000H, 000H, 030H, 030H, 000H ; + D_2E
FBEE 060C183060C08000	5821	DB 060H, 00CH, 018H, 030H, 060H, 0C0H, 080H, 000H ; + D_2F
FBE6 7CC6CEDEF6E67C00	5822	DB 07CH, 0C6H, 0CEH, 0DEH, 0F6H, 0E6H, 07CH, 000H ; + D_30
FBF6 307030303070FC00	5823	DB 030H, 070H, 030H, 030H, 030H, 030H, 0FCH, 000H ; + D_31
FBFE 78CC0C3860CCFC00	5824	DB 078H, 0CCH, 0CCH, 038H, 060H, 0CCH, 0FCH, 000H ; + D_32
FC06 78CC0C380CC78000	5825	DB 078H, 0CCH, 0CCH, 038H, 00CH, 0CCH, 078H, 000H ; + D_33
FC0E 183C6CCCFE0C1E00	5826	DB 01CH, 03CH, 06CH, 0CCH, 0FEH, 0CCH, 01EH, 000H ; + D_34
FC16 FCC0F80CC0C78000	5827	DB 0FCH, 0C0H, 0F8H, 00CH, 0CCH, 0CCH, 078H, 000H ; + D_35
FC1E 3860CF8C0CC78000	5828	DB 038H, 060H, 0C0H, 0F8H, 0CCH, 0CCH, 078H, 000H ; + D_36
FC26 FCC0CC1830303000	5829	DB 0FCH, 0CCH, 0CCH, 018H, 030H, 030H, 030H, 000H ; + D_37
FC2E 78CCCC78CCCC7800	5830	DB 078H, 0CCH, 0CCH, 078H, 0CCH, 0CCH, 078H, 000H ; + D_38
FC36 78CCCC7C0C187000	5831	DB 078H, 0CCH, 0CCH, 07CH, 0CCH, 018H, 070H, 000H ; + D_39
FC3E 0030300000303000	5832	DB 000H, 030H, 030H, 000H, 000H, 030H, 030H, 000H ; + D_3A
FC46 0030300000303060	5833	DB 000H, 030H, 030H, 000H, 000H, 030H, 030H, 060H ; + D_3B
FC4E 183060C060301800	5834	DB 018H, 030H, 060H, 0C0H, 060H, 030H, 018H, 000H ; + D_3C
FC56 0000FC0000FC0000	5835	DB 000H, 000H, 0FCH, 000H, 000H, 0FCH, 000H, 000H ; + D_3D
FC5E 6030180C18306000	5836	DB 060H, 030H, 018H, 00CH, 018H, 030H, 060H, 000H ; + D_3E
FC66 78CC0C1830003000	5837	DB 078H, 0CCH, 0CCH, 018H, 030H, 000H, 030H, 000H ; + D_3F
FC6E 7CC6DEDE0EC78000	5838	DB 07CH, 0C6H, 0DEH, 0DEH, 0DEH, 0CCH, 078H, 000H ; + D_40
FC76 3078CCCCFC000000	5839	DB 030H, 078H, 0CCH, 0CCH, 0FCH, 0CCH, 0CCH, 000H ; + D_41
FC7E 7CC6667C6666FC00	5840	DB 0FCH, 066H, 066H, 07CH, 066H, 066H, 0FCH, 000H ; + D_42
FC86 3C66C0C0C0663C00	5841	DB 03CH, 066H, 0C0H, 0C0H, 0C0H, 066H, 03CH, 000H ; + D_43
FC8E F86C6666666CF800	5842	DB 0F8H, 06CH, 066H, 066H, 066H, 06CH, 0F8H, 000H ; + D_44
FC96 FE6268786862FE00	5843	DB 0FEH, 062H, 068H, 078H, 068H, 062H, 0FEH, 000H ; + D_45
FC9E FE6268786862FE00	5844	DB 0FEH, 062H, 068H, 078H, 068H, 060H, 0F0H, 000H ; + D_46
FCAB 3C66C0C0C663E000	5845	DB 03CH, 066H, 0C0H, 0C0H, 0CEH, 066H, 03EH, 000H ; + D_47
FCAC CCCCCCF000000000	5846	DB 0CCH, 0CCH, 0CCH, 0FCH, 0CCH, 0CCH, 0CCH, 000H ; + D_48
FCBB 7830303030307800	5847	DB 078H, 030H, 030H, 030H, 030H, 030H, 078H, 000H ; + D_49
FCBE 1E0C0C0CCCCC7800	5848	DB 01EH, 0CCH, 0CCH, 0CCH, 0CCH, 0CCH, 078H, 000H ; + D_4A
FCCE E6666C786C6E6000	5849	DB 0E6H, 066H, 06CH, 078H, 06CH, 066H, 0E6H, 000H ; + D_4B
FCDE F0606060626FE000	5850	DB 0F0H, 060H, 060H, 060H, 062H, 066H, 0FEH, 000H ; + D_4C
FCDE C6E6FEEDC6C6C000	5851	DB 0C6H, 0EEH, 0FEH, 0FEH, 0D6H, 0C6H, 0C6H, 000H ; + D_4D
FCDE C6E6F0E6C6C6C000	5852	DB 0C6H, 0E6H, 0F6H, 0DEH, 0CEH, 0C6H, 0C6H, 000H ; + D_4E
FCDE 386C6AC666C63800	5853	DB 038H, 06CH, 0C6H, 0C6H, 0C6H, 06CH, 038H, 000H ; + D_4F
FCDE FC66667C6060F000	5854	DB 0FCH, 066H, 066H, 07CH, 060H, 060H, 0F0H, 000H ; + D_50
FCF6 78CCCCC0C781C000	5855	DB 078H, 0CCH, 0CCH, 0CCH, 0CCH, 078H, 01CH, 000H ; + D_51
FCFE FC66667C6666E000	5856	DB 0FCH, 066H, 066H, 07CH, 06CH, 066H, 0E6H, 00CH ; + D_52
FD06 78CE0701C0C78000	5857	DB 078H, 0CCH, 0E0H, 070H, 01CH, 0CCH, 078H, 000H ; + D_53
FD0E FC84303030307800	5858	DB 0FCH, 0B4H, 030H, 030H, 030H, 030H, 078H, 000H ; + D_54
FD16 CCCCCCCCCCF0C000	5859	DB 0CCH, 0CCH, 0CCH, 0CCH, 0CCH, 0CCH, 0FCH, 000H ; + D_55
FD1E CCCCCCCCCC783000	5860	DB 0CCH, 0CCH, 0CCH, 0CCH, 0CCH, 078H, 030H, 000H ; + D_56
FD26 C6C6C6D6FE6EC600	5861	DB 0C6H, 0C6H, 0C6H, 0D6H, 0FEH, 0E6H, 0C6H, 000H ; + D_57
FD2E C6C6C38386C6C000	5862	DB 0C6H, 0C6H, 06CH, 038H, 038H, 06CH, 0C6H, 000H ; + D_58
FD36 7CCCCC7830307800	5863	DB 0CCH, 0CCH, 0CCH, 078H, 030H, 030H, 078H, 000H ; + D_59

LOC OBJ	LINE	SOURCE
FD3E FEC60C103266FE00	5864	DB 0FEH,0C6H,00CH,010H,032H,066H,0FEH,000H ; Z D_5A
FD46 7860606060607800	5865	DB 078H,060H,060H,060H,060H,060H,078H,000H ; I D_5B
FD4E C06030C060602000	5866	DB 0C0H,060H,030H,018H,0C0H,066H,002H,000H ; BACKSLASH D_5C
FD56 7818181818187800	5867	DB 078H,018H,018H,018H,018H,018H,078H,000H ; I D_5D
FD5E 10386CC600000000FF	5868	DB 010H,038H,06CH,0C6H,0C0H,000H,000H,000H ; CIRCUMFLEX D_5E
FD66 00000000000000FF	5869	DB 000H,000H,000H,000H,000H,000H,000H,0FFH ; _ D_5F
FD6E 3030180000000000	5870	DB 030H,030H,018H,000H,000H,000H,000H,000H ; ' D_60
FD76 0000780C7CC76000	5871	DB 000H,000H,078H,00CH,07CH,0CCH,076H,000H ; LOWER CASE A D_61
FD7E E060607C6666DC00	5872	DB 0E0H,060H,060H,07CH,066H,066H,0DCCH,000H ; L.C. B D_62
FD86 000078CC0CC07800	5873	DB 000H,000H,078H,0CCH,0CCH,0CCH,0CCH,000H ; L.C. C D_63
FD8E 1C0C0C7C0CC07600	5874	DB 01CH,0CCH,00CH,07CH,0CCH,0CCH,076H,000H ; L.C. D D_64
FD96 000078CC7CC07800	5875	DB 000H,000H,078H,0CCH,0FCH,0CCH,078H,000H ; L.C. E D_65
FD9E 386C6F06060F0000	5876	DB 038H,06CH,060H,0F0H,060H,060H,0F0H,000H ; L.C. F D_66
FDA6 000076CCCC7C0CF8	5877	DB 000H,000H,076H,0CCH,0CCH,07CH,0CCH,0F8H ; L.C. G D_67
FDAE E0606C766666E000	5878	DB 0E0H,060H,06CH,076H,066H,066H,0E6H,000H ; L.C. H D_68
FDB6 3000703030307800	5879	DB 030H,000H,070H,030H,030H,030H,078H,000H ; L.C. I D_69
FD8E 0C000C0C0CCCC7600	5880	DB 0CCH,000H,0CCH,0CCH,0CCH,0CCH,0CCH,078H ; L.C. J D_6A
FDCE E06066C786CE6000	5881	DB 0E0H,060H,066H,06CH,078H,06CH,06CH,000H ; L.C. K D_6B
FDCE 7030303030307800	5882	DB 070H,030H,030H,030H,030H,030H,078H,000H ; L.C. L D_6C
FD06 0000CCFFED6C6000	5883	DB 000H,000H,0CCH,0FEH,0FEH,0D6H,0C6H,000H ; L.C. M D_6D
FDDE 0000F8CCCCCCCC00	5884	DB 000H,000H,0F8H,0CCH,0CCH,0CCH,0CCH,000H ; L.C. N D_6E
FD06 000078CCCC780000	5885	DB 000H,000H,078H,0CCH,0CCH,0CCH,078H,000H ; L.C. O D_6F
FDDE 0000DC66667C60F0	5886	DB 000H,000H,0DCH,066H,066H,07CH,066H,0F0H ; L.C. P D_70
FD06 000076CCCC7C0C1E	5887	DB 000H,000H,076H,0CCH,0CCH,07CH,0CCH,01EH ; L.C. Q D_71
FD0E 0000DC766660F000	5888	DB 000H,000H,0DCH,076H,066H,060H,0F0H,000H ; L.C. R D_72
FE06 00007CC0780CF800	5889	DB 000H,000H,07CH,0C0H,078H,0CCH,0F8H,000H ; L.C. S D_73
FE0E 10307C3030341800	5890	DB 010H,030H,07CH,030H,030H,034H,018H,000H ; L.C. T D_74
FE16 0000CCCCCCCC7600	5891	DB 000H,000H,0CCH,0CCH,0CCH,0CCH,076H,000H ; L.C. U D_75
FE1E 0000CCCCCCCC783000	5892	DB 000H,000H,0CCH,0CCH,0CCH,078H,030H,000H ; L.C. V D_76
FE26 0000C6D6FEF6C600	5893	DB 000H,000H,0C6H,0D6H,0FEH,0FEH,06CH,000H ; L.C. W D_77
FE2E 0000C6C386CC6000	5894	DB 000H,000H,0C6H,06CH,038H,06CH,0C6H,000H ; L.C. X D_78
FE36 0000CCCCCCCC7C0CF8	5895	DB 000H,000H,0CCH,0CCH,0CCH,0CCH,07CH,0CCH,0F8H ; L.C. Y D_79
FE3E 0000F983064FC000	5896	DB 000H,000H,0FCH,098H,030H,064H,0FCH,000H ; L.C. Z D_7A
FE46 1C3030E030301C00	5897	DB 01CH,030H,030H,0E0H,030H,030H,01CH,000H ; (D_7B
FE4E 1818180018181800	5898	DB 018H,018H,018H,000H,018H,018H,018H,000H ; D_7C
FE56 E030301C3030E000	5899	DB 0E0H,030H,030H,01CH,030H,030H,0E0H,000H ; D_7D
FE5E 76DC000000000000	5900	DB 076H,0DCH,000H,000H,000H,000H,000H,000H ; TILDE D_7E
FE66 0010386CC6C6FE00	5901	DB 000H,010H,038H,06CH,0C6H,0C6H,0FEH,000H ; DELTA D_7F

```

5902
5903 ;--- INT 1A -----
5904 ; TIME_OF_DAY
5905 ; THIS ROUTINE ALLOWS THE CLOCK TO BE SET/READ
5906 ;
5907 ; INPUT
5908 ; (AH) = 0 READ THE CURRENT CLOCK SETTING
5909 ; RETURNS CX = HIGH PORTION OF COUNT
5910 ; DX = LOW PORTION OF COUNT
5911 ; AL = 0 IF TIMER HAS NOT PASSED
5912 ; 24 HOURS SINCE LAST READ
5913 ; <0 IF ON ANOTHER DAY
5914 ; (AH) = 1 SET THE CURRENT CLOCK
5915 ; CX = HIGH PORTION OF COUNT
5916 ; DX = LOW PORTION OF COUNT
5917 ; NOTE: COUNTS OCCUR AT THE RATE OF
5918 ; 1193180/65536 COUNTS/SEC
5919 ; (OR ABOUT 18.2 PER SECOND -- SEE EQUATES BELOW)
5920 ;-----
5921 ASSUME CS:CODE,DS:DATA
5922 ORG 0FE6EH
5923 TIME_OF_DAY PROC FAR
5924 STI
5925 PUSH DS
5926 CALL DDS
5927 OR AH,AH
5928 JZ T2
5929 DEC AH
5930 JZ T3
5931 T1:
5932 STI
5933 POP DS
5934 IRET
5935 T2:
5936 CLI
5937 MOV AL,TIMER_OFL
5938 MOV TIMER_OFL,0
5939 MOV CX,TIMER_HIGH
5940 MOV DX,TIMER_LOW
5941
5942 ; INTERRUPTS BACK ON
5943 ; SAVE SEGMENT
5944 ; AH=0
5945 ; READ_TIME
5946 ; AH=1
5947 ; SET_TIME
5948 ; TOD_RETURN
5949 ; INTERRUPTS BACK ON
5950 ; RECOVER SEGMENT
5951 ; RETURN TO CALLER
5952 ; READ_TIME
5953 ; NO TIMER INTERRUPTS WHILE READING
5954 ; GET OVERFLOW, AND RESET THE FLAG

```

```

LOC OBJ          LINE    SOURCE

FE8F EBEA        5941      JMP     T1                ; TOD_RETURN
FE91             5942      T3:          ; SET_TIME
FE91 FA          5943      CLI             ; NO INTERRUPTS WHILE WRITING
FE92 89166C00    5944      MOV     TIMER_LOW,DX
FE96 8906E00     5945      MOV     TIMER_HIGH,CX    ; SET THE TIME
FE9A C0670000D   5946      MOV     TIMER_OFL,0     ; RESET OVERFLOW
FE9F EBD A       5947      JMP     T1                ; TOD_RETURN
                    5948      TIME_OF_DAY      ENDP
                    5949
                    5950      ;-----
                    5951      ; THIS ROUTINE HANDLES THE TIMER INTERRUPT FROM      :
                    5952      ; CHANNEL 0 OF THE 8253 TIMER. INPUT FREQUENCY      :
                    5953      ; IS 1.19318 MHZ AND THE DIVISOR IS 65536, RESULTING :
                    5954      ; IN APPROX. 18.2 INTERRUPTS EVERY SECOND.      :
                    5955      ; :
                    5956      ; THE INTERRUPT HANDLER MAINTAINS A COUNT OF INTERRUPTS :
                    5957      ; SINCE POWER ON TIME, WHICH MAY BE USED TO ESTABLISH :
                    5958      ; TIME OF DAY. :
                    5959      ; THE INTERRUPT HANDLER ALSO DECREMENTS THE MOTOR :
                    5960      ; CONTROL COUNT OF THE DISKETTE, AND WHEN IT EXPIRES, :
                    5961      ; WILL TURN OFF THE DISKETTE MOTOR, AND RESET THE :
                    5962      ; MOTOR RUNNING FLAGS. :
                    5963      ; THE INTERRUPT HANDLER WILL ALSO INVOKE A USER ROUTINE :
                    5964      ; THROUGH INTERRUPT ICH AT EVERY TIME TICK. THE USER :
                    5965      ; MUST CODE A ROUTINE AND PLACE THE CORRECT ADDRESS IN :
                    5966      ; THE VECTOR TABLE. :
                    5967      ;-----
FEA5             5968      ORG     OFEASH
FEA5             5969      TIMER_INT  PROC     FAR
FEA5 FB          5970      STI             ; INTERRUPTS BACK ON
FEA6 1E          5971      PUSH    DS
FEA7 50          5972      PUSH    AX
FEA8 52          5973      PUSH    DX                ; SAVE MACHINE STATE
FEA9 E89200      5974      CALL    DDS
FEAC FF066C00    5975      INC     TIMER_LOW        ; INCREMENT TIME
FEB0 7504        5976      JNZ     T4                ; TEST_DAY
FEB2 FF066E00    5977      INC     TIMER_HIGH       ; INCREMENT HIGH WORD OF TIME
FEB6             5978      T4:          ; TEST_DAY
FEB6 833E60010   5979      CMP     TIMER_HIGH,010H   ; TEST FOR COUNT EQUALING 24 HOURS
FEBB 7515        5980      JNZ     T5                ; DISKETTE_CTL
FEBD 813E6C00B000 5981      CMP     TIMER_LOW,0B0H
FECC 750D        5982      JNZ     T5                ; DISKETTE_CTL
                    5983
                    5984      ;----- TIMER HAS GONE 24 HOURS
                    5985
FECE 2BC0        5986      SUB     AX,AX
FECE A36E00      5987      MOV     TIMER_HIGH,AX
FECA A36C00      5988      MOV     TIMER_LOW,AX
FECD C06700001   5989      MOV     TIMER_OFL,1
                    5990
                    5991      ;----- TEST FOR DISKETTE TIME OUT
                    5992
FED2             5993      T5:          ; DISKETTE_CTL
FED2 FE0E4000    5994      DEC     MOTOR_COUNT
FED6 750B        5995      JNZ     T6                ; RETURN IF COUNT NOT OUT
FED8 80263F00F0 5996      AND     MOTOR_STATUS,0F0H ; TURN OFF MOTOR RUNNING BITS
FEDD B00C        5997      MOV     AL,0CH
FEDF BA203       5998      MOV     DX,03F2H        ; FDC CTL PORT
FEE2 EE          5999      OUT     DX,AL           ; TURN OFF THE MOTOR
FEE3             6000      T6:          ; TIMER_RET:
FEE3 CD1C        6001      INT     ICH             ; TRANSFER CONTROL TO A USER ROUTINE
FEE5 B020        6002      MOV     AL,EDI
FEE7 E620        6003      OUT     020H,AL        ; END OF INTERRUPT TO 8259
FEE9 5A          6004      POP     DX
FEEA 58          6005      POP     AX
FEEB 1F          6006      POP     DS                ; RESET MACHINE STATE
FECC CF          6007      IRET             ; RETURN FROM INTERRUPT
                    6008      TIMER_INT      ENDP
                    6009
FEED 31303031    6010      F3B    DB     '1801',13,10
FEF1 0D          6011
FEF2 0A          6012
                    6013      ;-----
                    6014      ; THESE ARE THE VECTORS WHICH ARE MOVED INTO      :
                    6015      ; THE 8086 INTERRUPT AREA DURING POWER ON. :
                    6016      ; ONLY THE OFFSETS ARE DISPLAYED HERE, CODE SEGMENT :

```

LOC OBJ

LINE SOURCE

```

6016 ; WILL BE ADDED FOR ALL OF THEM, EXCEPT WHERE NOTED :
6017 ;-----
6018 ASSUME CS:CODE
FEF3 6019 ORG OFF47H
FEF3 6020 VECTOR_TABLE LABEL WORD ; VECTOR TABLE FOR MOVE TO INTERRUPTS
FEF3 A5FE 6021 DW OFFSET TIMER_INT ; INTERRUPT 8
FEF5 87E9 6022 DW OFFSET KB_INT ; INTERRUPT 9
FEF7 D0E6 6023 DW OFFSET D_EOI ; INTERRUPT A
FEF9 D0E6 6024 DW OFFSET D_EOI ; INTERRUPT B
FEFB D0E6 6025 DW OFFSET D_EOI ; INTERRUPT C
FEFD D0E6 6026 DW OFFSET D_EOI ; INTERRUPT D
FEFF 57EF 6027 DW OFFSET DISK_INT ; INTERRUPT E
FF01 D0E6 6028 DW OFFSET D_EOI ; INTERRUPT F
FF03 65F0 6029 DW OFFSET VIDEO_IO ; INTERRUPT 10H
FF05 40F8 6030 DW OFFSET EQUIPMENT ; INTERRUPT 11H
FF07 41F8 6031 DW OFFSET MEMORY_SIZE_DET ; INTERRUPT 12H
FF09 59EC 6032 DW OFFSET DISKETTE_IO ; INTERRUPT 13H
FF0B 39E7 6033 DW OFFSET RS232_IO ; INTERRUPT 14H
FF0D 59F8 6034 DW OFFSET CASSETTE_IO ; INTERRUPT 15H
FF0F 2EE8 6035 DW OFFSET KEYBOARD_IO ; INTERRUPT 16H
FF11 D2EF 6036 DW OFFSET PRINTER_IO ; INTERRUPT 17H
6037
FF13 0000 6038 DW 00000H ; INTERRUPT 18H
6039 ; DW 0F600H ; MUST BE INSERTED INTO TABLE LATER
6040
FF15 F2E6 6041 DW OFFSET BOOT_STRAP ; INTERRUPT 19H
FF17 6EFE 6042 DW TIME_OF_DAY ; INTERRUPT 1AH -- TIME OF DAY
FF19 53FF 6043 DW DUMMY_RETURN ; INTERRUPT 1BH -- KEYBOARD BREAK ADDR
FF1B 53FF 6044 DW DUMMY_RETURN ; INTERRUPT 1C -- TIMER BREAK ADDR
FF1D A4F0 6045 DW VIDEO_PARMS ; INTERRUPT 1D -- VIDEO PARAMETERS
FF1F C7EF 6046 DW OFFSET DISK_BASE ; INTERRUPT 1E -- DISK PARMS
FF21 0000 6047 DW 0 ; INTERRUPT 1F -- POINTER TO VIDEO EXT
6048
FF23 50415249545920 6049 D2 DB 'PARITY CHECK 1',13,10
43484543482031
FF31 00
FF32 0A
FF33 20333031 6050 F1 DB '301',13,10
FF37 00
FF38 0A
FF39 313331 6051 F2 DB '131',13,10
FF3C 00
FF3D 0A
6052
FF3E 6053 DDS PROC NEAR
FF3E 50 6054 PUSH AX ; SAVE AX
FF3F B84000 6055 MOV AX,DATA
FF42 8ED8 6056 MOV DS,AX ; SET DATA SEGMENT
FF44 58 6057 POP AX ; RESTORE AX
FF45 C3 6058 RET
6059 DDS ENDP
6060
6061 ;-----
6062 ; TEMPORARY INTERRUPT SERVICE ROUTINE :
6063 ;-----
FF47 6064 ORG OFF47H
FF47 6065 D11 PROC NEAR
FF47 B401 6066 MOV AH,1
FF49 50 6067 PUSH AX ; SAVE REG AX CONTENTS
FF4A B0FF 6068 MOV AL,OFFH ; MASK ALL INTERRUPTS OFF
FF4C E621 6069 OUT INTA01,AL
FF4E B020 6070 MOV AL,EOI
FF50 E620 6071 OUT INTA00,AL
FF52 58 6072 POP AX ; RESTORE REG AX CONTENTS
FF53 6073 DUMMY_RETURN: ; NEED IRET FOR VECTOR TABLE
FF53 CF 6074 IRET
6075 D11 ENDP
6076
6077 ;-- INT 5 -----
6078 ; THIS LOGIC WILL BE INVOKED BY INTERRUPT 05H TO PRINT THE :
6079 ; SCREEN. THE CURSOR POSITION AT THE TIME THIS ROUTINE IS INVOKED :
6080 ; WILL BE SAVED AND RESTORED UPON COMPLETION. THE ROUTINE IS :
6081 ; INTENDED TO RUN WITH INTERRUPTS ENABLED. IF A SUBSEQUENT :
6082 ; 'PRINT SCREEN' KEY IS DEPRESSED DURING THE TIME THIS ROUTINE :
6083 ; IS PRINTING IT WILL BE IGNORED. :
6084 ; ADDRESS 50:0 CONTAINS THE STATUS OF THE PRINT SCREEN: :
6085 ; :

```

```

6086 ; 50:0 =0 EITHER PRINT SCREEN HAS NOT BEEN CALLED ;
6087 ; OR UPON RETURN FROM A CALL THIS INDICATES ;
6088 ; A SUCCESSFUL OPERATION. ;
6089 ; =1 PRINT SCREEN IS IN PROGRESS ;
6090 ; =255 ERROR ENCOUNTERED DURING PRINTING ;
6091 ;-----
6092 ASSUME CS:CODE,DS:XXDATA
6093 ORG OFF54H
FF54 PRINT_SCREEN PROC FAR
6094 STI ; MUST RUN WITH INTERRUPTS ENABLED
FF54 FB 6095 ; MUST USE 50:0 FOR DATA AREA STORAGE
FF55 1E 6096
FF56 50 6097
FF57 53 6098
FF58 51 6099
FF59 52 6100
FF5A B05000 6101
FF5D 0ED8 6102
FF5F 803E000001 6103
FF64 745F 6104
FF66 C060000001 6105
FF6B B04F 6106
FF6D C010 6107
6108
6109
6110 ;-----
6111 ; AT THIS POINT WE KNOW THE COLUMNS/LINE ARE IN ;
6112 ; [AX] AND THE PAGE IF APPLICABLE IS IN [BH]. THE STACK ;
6113 ; HAS DS,AX,BX,CX,DX PUSHED. [AL] HAS VIDEO MODE ;
6114 ;-----
FF6F 8ACC 6115
FF71 B519 6116
FF73 E05500 6117
FF76 51 6118
FF77 B403 6119
FF79 CD10 6120
FF7B 59 6121
FF7C 52 6122
FF7D 33D2 6123
6124
6125 ;-----
6126 ; THE LOOP FROM PRI10 TO THE INSTRUCTION PRIOR TO PRI20 ;
6127 ; IS THE LOOP TO READ EACH CURSOR POSITION FROM THE ;
6128 ; SCREEN AND PRINT. ;
6129 ;-----
FF7F PRI10:
FF7F B402 6130
FF81 CD10 6131
FF83 B408 6132
FF85 CD10 6133
FF87 0AC0 6134
FF89 7502 6135
FF8B B020 6136
FF8D 6137
FF8D 52 6138
FF8E 33D2 6139
FF90 32E4 6140
FF92 CD17 6141
FF94 5A 6142
FF95 F6C425 6143
FF98 7521 6144
FF9A FEC2 6145
FF9C 3ACA 6146
FF9E 75D0 6147
FFA0 32D2 6148
FFA2 0AE2 6149
FFA4 52 6150
FFA5 E02300 6151
FFA8 5A 6152
FFA9 FEC6 6153
FFAB 3AEE 6154
FFAD 75D0 6155
FFAF 6156
FFAF 5A 6157
FFB0 B402 6158
FFB2 CD10 6159
FFB4 C060000000 6160
FFB9 EBOA 6161
FFBB 6162
ERR10:

```

LOC OBJ	LINE	SOURCE
FFBB 5A	6163	POP DX ; GET CURSOR POSITION
FFBC B402	6164	MOV AH,2 ; TO REQUEST CURSOR SET
FFBE CD10	6165	INT 10H ; CURSOR POSITION RESTORED
FFC0	6166	ERR20:
FFC0 C6060000FF	6167	MOV STATUS_BYTE,0FFH ; INDICATE ERROR
FFC5	6168	EXIT:
FFC5 5A	6169	POP DX ; RESTORE ALL THE REGISTERS USED
FFC6 59	6170	POP CX
FFC7 5B	6171	POP BX
FFC8 58	6172	POP AX
FFC9 1F	6173	POP DS
FFCA CF	6174	IRET
	6175	PRINT_SCREEN ENDP
	6176	
	6177	;------ CARRIAGE RETURN, LINE FEED SUBROUTINE
	6178	
FFCB	6179	CRLF PROC NEAR
FFCB 3302	6180	XOR DX,DX ; PRINTER 0
FFCD 32E4	6181	XOR AH,AH ; WILL NOW SEND INITIAL LF,CR
	6182	; TO PRINTER
FFCF B00A	6183	MOV AL,12Q ; LF
FFD1 CD17	6184	INT 17H ; SEND THE LINE FEED
FFD3 32E4	6185	XOR AH,AH ; NOW FOR THE CR
FFD5 B00D	6186	MOV AL,15Q ; CR
FFD7 CD17	6187	INT 17H ; SEND THE CARRIAGE RETURN
FFD9 C3	6188	RET
	6189	CRLF ENDP
	6190	
FFDA 50415249545920 434845434B2032	6191	D1 DB 'PARITY CHECK 2',13,10
FFE8 0D		
FFE9 0A		
FFEA 363031	6192	F3 DB '601',13,10
FFED 0D		
FFEE 0A		
----	6193	
	6194	CODE ENDS
	6195	
	6196	;------
	6197	; POWER ON RESET VECTOR :
	6198	;------
----	6199	VECTOR SEGMENT AT 0FFFFH
	6200	
	6201	;------ POWER ON RESET
	6202	
0000 EASBE000F0	6203	JMP RESET
	6204	
0005 31302F32372F38 32	6205	DB '10/27/82' ; RELEASE MARKER
----	6206	VECTOR ENDS
	6207	END

```

1  *TITLE(FIXED DISK BIOS FOR IBM DISK CONTROLLER)
2
3  |-- INT 13 -----
4  ;
5  ; FIXED DISK I/O INTERFACE
6  ;
7  ;     THIS INTERFACE PROVIDES ACCESS TO 5 1/4" FIXED DISKS
8  ;     THROUGH THE IBM FIXED DISK CONTROLLER.
9  ;
10 -----
11
12 |-----
13 ;     THE BIOS ROUTINES ARE MEANT TO BE ACCESSED THROUGH
14 ;     SOFTWARE INTERRUPTS ONLY. ANY ADDRESSES PRESENT IN
15 ;     THE LISTINGS ARE INCLUDED ONLY FOR COMPLETENESS,
16 ;     NOT FOR REFERENCE. APPLICATIONS WHICH REFERENCE
17 ;     ABSOLUTE ADDRESSES WITHIN THE CODE SEGMENT
18 ;     VIOLATE THE STRUCTURE AND DESIGN OF BIOS.
19 ;
20 |-----
21 ; INPUT    (AH = HEX VALUE)
22 ;
23 ;     (AH)=00 RESET DISK (DL = 80H,81H) / DISKETTE
24 ;     (AH)=01 READ THE STATUS OF THE LAST DISK OPERATION INTO (AL)
25 ;     NOTE: DL < 80H - DISKETTE
26 ;           DL > 80H - DISK
27 ;     (AH)=02 READ THE DESIRED SECTORS INTO MEMORY
28 ;     (AH)=03 WRITE THE DESIRED SECTORS FROM MEMORY
29 ;     (AH)=04 VERIFY THE DESIRED SECTORS
30 ;     (AH)=05 FORMAT THE DESIRED TRACK
31 ;     (AH)=06 FORMAT THE DESIRED TRACK AND SET BAD SECTOR FLAGS
32 ;     (AH)=07 FORMAT THE DRIVE STARTING AT THE DESIRED TRACK
33 ;     (AH)=08 RETURN THE CURRENT DRIVE PARAMETERS
34 ;
35 ;     (AH)=09 INITIALIZE DRIVE PAIR CHARACTERISTICS
36 ;           INTERRUPT 41 POINTS TO DATA BLOCK
37 ;     (AH)=0A READ LONG
38 ;     (AH)=0B WRITE LONG
39 ;     NOTE: READ AND WRITE LONG ENCOMPASS 512 + 4 BYTES ECC
40 ;     (AH)=0C SEEK
41 ;     (AH)=0D ALTERNATE DISK RESET (SEE DL)
42 ;     (AH)=0E READ SECTOR BUFFER
43 ;     (AH)=0F WRITE SECTOR BUFFER,
44 ;           (RECOMMENDED PRACTICE BEFORE FORMATTING)
45 ;     (AH)=10 TEST DRIVE READY
46 ;     (AH)=11 RECALIBRATE
47 ;     (AH)=12 CONTROLLER RAM DIAGNOSTIC
48 ;     (AH)=13 DRIVE DIAGNOSTIC
49 ;     (AH)=14 CONTROLLER INTERNAL DIAGNOSTIC
50 ;
51 ;     REGISTERS USED FOR FIXED DISK OPERATIONS
52 ;
53 ;     (DL)  - DRIVE NUMBER    (80H-87H FOR DISK, VALUE CHECKED)
54 ;     (DH)  - HEAD NUMBER    (0-7 ALLOWED, NOT VALUE CHECKED)
55 ;     (CH)  - CYLINDER NUMBER (0-1023, NOT VALUE CHECKED)(SEE CL)
56 ;     (CL)  - SECTOR NUMBER  (1-17, NOT VALUE CHECKED)
57 ;
58 ;     NOTE: HIGH 2 BITS OF CYLINDER NUMBER ARE PLACED
59 ;           IN THE HIGH 2 BITS OF THE CL REGISTER
60 ;           (10 BITS TOTAL)
61 ;     (AL)  - NUMBER OF SECTORS (MAXIMUM POSSIBLE RANGE 1-80H,
62 ;           FOR READ/WRITE LONG 1-79H)
63 ;           (INTERLEAVE VALUE FOR FORMAT 1-160)
64 ;     (ES:BX) - ADDRESS OF BUFFER FOR READS AND WRITES,
65 ;           (NOT REQUIRED FOR VERIFY)
66 ;
67 ; OUTPUT
68 ;     AH = STATUS OF CURRENT OPERATION
69 ;     STATUS BITS ARE DEFINED IN THE EQUATES BELOW
70 ;     CY = 0 SUCCESSFUL OPERATION (AH=0 ON RETURN)
71 ;     CY = 1 FAILED OPERATION (AH HAS ERROR REASON)
72 ;
73 ;     NOTE: ERROR 11H INDICATES THAT THE DATA READ HAD A RECOVERABLE
74 ;           ERROR WHICH WAS CORRECTED BY THE ECC ALGORITHM. THE DATA
75 ;           IS PROBABLY GOOD, HOWEVER THE BIOS ROUTINE INDICATES AN
76 ;           ERROR TO ALLOW THE CONTROLLING PROGRAM A CHANCE TO DECIDE
77 ;           FOR ITSELF. THE ERROR MAY NOT RECUR IF THE DATA IS

```

```

78      ; REWRITTEN. (AL) CONTAINS THE BURST LENGTH.
79      ;
80      ; IF DRIVE PARAMETERS WERE REQUESTED,
81      ;
82      ; DL = NUMBER OF CONSECUTIVE ACKNOWLEDGING DRIVES ATTACHED (0-2)
83      ; (CONTROLLER CARD ZERO TALLY ONLY)
84      ; DH = MAXIMUM USEABLE VALUE FOR HEAD NUMBER
85      ; CH = MAXIMUM USEABLE VALUE FOR CYLINDER NUMBER
86      ; CL = MAXIMUM USEABLE VALUE FOR SECTOR NUMBER
87      ; AND CYLINDER NUMBER HIGH BITS
88      ;
89      ; REGISTERS WILL BE PRESERVED EXCEPT WHEN THEY ARE USED TO RETURN
90      ; INFORMATION.
91      ;
92      ; NOTE: IF AN ERROR IS REPORTED BY THE DISK CODE, THE APPROPRIATE
93      ; ACTION IS TO RESET THE DISK, THEN RETRY THE OPERATION.
94      ;
95      ; -----
96
00FF    97 SENSE_FAIL EQU 0FFH ; SENSE OPERATION FAILED
00BB    98 UNDEF_ERR EQU 0BBH ; UNDEFINED ERROR OCCURRED
0060    99 TIME_OUT EQU 60H ; ATTACHMENT FAILED TO RESPOND
0040   100 BAD_SEEK EQU 40H ; SEEK OPERATION FAILED
0020   101 BAD_CNTLR EQU 20H ; CONTROLLER HAS FAILED
0011   102 DATA_CORRECTED EQU 11H ; ECC CORRECTED DATA ERROR
0010   103 BAD_ECC EQU 10H ; BAD ECC ON DISK READ
000B   104 BAD_TRACK EQU 0BH ; BAD TRACK FLAG DETECTED
0009   105 DMA_BOUNDARY EQU 09H ; ATTEMPT TO DMA ACROSS 64K BOUNDARY
0007   106 INIT_FAIL EQU 07H ; DRIVE PARAMETER ACTIVITY FAILED
0005   107 BAD_RESET EQU 05H ; RESET FAILED
0004   108 RECORD_NOT_FND EQU 04H ; REQUESTED SECTOR NOT FOUND
0002   109 BAD_ADDR_MARK EQU 02H ; ADDRESS MARK NOT FOUND
0001   110 BAD_CMD EQU 01H ; BAD COMMAND PASSED TO DISK I/O
111
112      ; -----
113      ; INTERRUPT AND STATUS AREAS :
114      ; -----
115
----    116 DUMMY SEGMENT AT 0
0034   117 ORG 00H*4 ; FIXED DISK INTERRUPT VECTOR
0034   118 HDISK_INT LABEL DWORD
004C   119 ORG 13H*4 ; DISK INTERRUPT VECTOR
004C   120 ORG_VECTOR LABEL DWORD
0064   121 ORG 19H*4 ; BOOTSTRAP INTERRUPT VECTOR
0064   122 BOOT_VEC LABEL DWORD
007B   123 ORG 1EH*4 ; DISKETTE PARAMETERS
007B   124 DISKETTE_PARM LABEL DWORD
0100   125 ORG 040H*4 ; NEW DISKETTE INTERRUPT VECTOR
0100   126 DISK_VECTOR LABEL DWORD
0104   127 ORG 041H*4 ; FIXED DISK PARAMETER VECTOR
0104   128 HF_TBL_VEC LABEL DWORD
7C00   129 ORG 7C00H ; BOOTSTRAP LOADER VECTOR
7C00   130 BOOT_LOCN LABEL FAR
----    131 DUMMY ENDS
132
----    133 DATA SEGMENT AT 40H
0042   134 ORG 42H
0042   135 CMD_BLOCK LABEL BYTE
0042 (7 ??) 136 HD_ERROR DB 7 DUP(?) ; OVERLAYS DISKETTE STATUS
006C   137 ORG 06CH
006C 006C ???? 138 TIMER_LOW DW ? ; TIMER LOW WORD
0072   139 ORG 72H
0072 0072 ???? 140 RESET_FLAG DW ? ; 1234H IF KEYBOARD RESET UNDERWAY
0074   141 ORG 74H
0074 ?? 142 DISK_STATUS DB ? ; FIXED DISK STATUS BYTE
0075 ?? 143 HF_NUM DB ? ; COUNT OF FIXED DISK DRIVES
0076 ?? 144 CONTROL_BYTE DB ? ; CONTROL BYTE DRIVE OPTIONS
0077 ?? 145 PORT_OFF DB ? ; PORT OFFSET
----    146 DATA ENDS
147
----    148 CODE SEGMENT
149
150      ; -----
151      ; HARDWARE SPECIFIC VALUES :
152      ; :
153      ; - CONTROLLER I/O PORT :
154      ; > WHEN READ FROM: :
```

```

155 ; HF_PORT+0 - READ DATA (FROM CONTROLLER TO CPU) ;
156 ; HF_PORT+1 - READ CONTROLLER HARDWARE STATUS ;
157 ; (CONTROLLER TO CPU) ;
158 ; HF_PORT+2 - READ CONFIGURATION SWITCHES ;
159 ; HF_PORT+3 - NOT USED ;
160 ; > WHEN WRITTEN TO: ;
161 ; HF_PORT+0 - WRITE DATA (FROM CPU TO CONTROLLER) ;
162 ; HF_PORT+1 - CONTROLLER RESET ;
163 ; HF_PORT+2 - GENERATE CONTROLLER SELECT PULSE ;
164 ; HF_PORT+3 - WRITE PATTERN TO DMA AND INTERRUPT ;
165 ; MASK REGISTER ;
166 ; ;
167 ;-----
168
0320 169 HF_PORT EQU 0320H ; DISK PORT
0008 170 RI_BUSY EQU 00001000B ; DISK PORT 1 BUSY BIT
0004 171 RI_BUS EQU 00000100B ; COMMAND/DATA BIT
0002 172 RI_IOMODE EQU 00000010B ; MODE BIT
0001 173 RI_REQ EQU 00000001B ; REQUEST BIT
174
0047 175 DMA_READ EQU 01000111B ; CHANNEL 3 (047H)
0048 176 DMA_WRITE EQU 01001011B ; CHANNEL 3 (048H)
0000 177 DMA EQU 0 ; DMA ADDRESS
0002 178 DMA_HIGH EQU 002H ; PORT FOR HIGH 4 BITS OF DMA
179
0000 180 TST_PDY_CMD EQU 00000000B ; CNTLR READY (00H)
0001 181 RECAL_CMD EQU 00000001B ; RECAL (01H)
0003 182 SENSE_CMD EQU 00000011B ; SENSE (03H)
0004 183 FMTDRV_CMD EQU 00000100B ; DRIVE (04H)
0005 184 CHK_TRK_CMD EQU 00000101B ; T CHK (05H)
0006 185 FMTTRK_CMD EQU 00000110B ; TRACK (06H)
0007 186 FMTBAD_CMD EQU 00000111B ; BAD (07H)
0008 187 READ_CMD EQU 00001000B ; READ (08H)
000A 188 WRITE_CMD EQU 00001010B ; WRITE (0AH)
000B 189 SEEK_CMD EQU 00001011B ; SEEK (0BH)
000C 190 INIT_DRV_CMD EQU 00001100B ; INIT (0CH)
000D 191 RD_ECC_CMD EQU 00001101B ; BURST (0DH)
000E 192 RD_BUFF_CMD EQU 00001110B ; BUFFER (0EH)
000F 193 WR_BUFF_CMD EQU 00001111B ; BUFFER (0FH)
00E0 194 RAM_DIAG_CMD EQU 11100000B ; RAM (E0H)
00E3 195 CHK_DRV_CMD EQU 11100011B ; DRV (E3H)
00E4 196 CNTLR_DIAG_CMD EQU 11100100B ; CNTLR (E4H)
00E5 197 RD_LONG_CMD EQU 11100101B ; RLONG (E5H)
00E6 198 WR_LONG_CMD EQU 11100110B ; WLONG (E6H)
199
0020 200 INT_CTL_PORT EQU 20H ; 8259 CONTROL PORT
0020 201 EOI EQU 20H ; END OF INTERRUPT COMMAND
202
0008 203 MAX_FILE EQU 8
0002 204 S_MAX_FILE EQU 2
205
206 ASSUME CS:CODE
207 ORG 0H
0000 55 208 DB 055H ; GENERIC BIOS HEADER
0001 AA 209 DB 0AAH
0002 10 210 DB 16D
211
212 ;-----
213 ; FIXED DISK I/O SETUP ;
214 ; ;
215 ; - ESTABLISH TRANSFER VECTORS FOR THE FIXED DISK ;
216 ; - PERFORM POWER ON DIAGNOSTICS ;
217 ; SHOULD AN ERROR OCCUR A "1701" MESSAGE IS DISPLAYED ;
218 ; ;
219 ;-----
220
0003 221 DISK_SETUP PROC FAR
0003 EB1E 222 JMP SHORT L3
0005 35303030303539 223 DB '5000059 (C)COPYRIGHT IBM 1982' ; COPYRIGHT NOTICE
20284329434F50
59524947485420
20494240203139
3832
0023 224 L3:
225 ASSUME DS:DUMMY
0023 2BC0 226 SUB AX,AX ; ZERO
0025 8ED8 227 MOV DS,AX

```


LOC	OBJ	LINE	SOURCE
0027	FA	228	CLI
0028	A14C00	229	MOV AX,WORD PTR ORG_VECTOR ; GET DISKETTE VECTOR
002B	A30001	230	MOV WORD PTR DISK_VECTOR,AX ; INTO INT 40H
002E	A14E00	231	MOV AX,WORD PTR ORG_VECTOR+2
0031	A30201	232	MOV WORD PTR DISK_VECTOR+2,AX
0034	C7064C005602	233	MOV WORD PTR ORG_VECTOR,OFFSET DISK_ID ; HDISK HANDLER
003A	8C0E4E00	234	MOV WORD PTR ORG_VECTOR+2,CS
003E	B86007	235	MOV AX,OFFSET HD_INT ; HDISK INTERRUPT
0041	A33400	236	MOV WORD PTR HDISK_INT,AX
0044	8C0E3600	237	MOV WORD PTR HDISK_INT+2,CS
0048	C70664C008601	238	MOV WORD PTR BOOT_VEC,OFFSET BOOT_STRAP ; BOOTSTRAP
004E	8C0E6600	239	MOV WORD PTR BOOT_VEC+2,CS
0052	C7060401E703	240	MOV WORD PTR HF_TBL_VEC,OFFSET FD_TBL ; PARAMETER TBL
0058	8C0E0601	241	MOV WORD PTR HF_TBL_VEC+2,CS
005C	FB	242	STI
		243	
		244	ASSUME DS:DATA
005D	B84000	245	MOV AX,DATA ; ESTABLISH SEGMENT
0060	8ED8	246	MOV DS,AX
0062	C606740000	247	MOV DISK_STATUS,0 ; RESET THE STATUS INDICATOR
0067	C606750000	248	MOV HF_NUM,0 ; ZERO COUNT OF DRIVES
006C	C606430000	249	MOV CHD_BLOCK+1,0 ; DRIVE ZERO, SET VALUE IN BLOCK
0071	C606770000	250	MOV PORT_OFF,0 ; ZERO CARD OFFSET
		251	
0076	B92500	252	MOV CX,25H ; RETRY COUNT
0079		253	
0079	E8F200	254	L4: CALL HD_RESET_1 ; RESET CONTROLLER
007C	7305	255	JNC L7
007E	E2F9	256	LOOP L4 ; TRY RESET AGAIN
0080	E9BF00	257	JMP ERROR_EX
0083		258	
0083	B90100	259	L7: MOV CX,1
0086	BA8000	260	MOV DX,80H
		261	
0089	B80012	262	MOV AX,1200H ; CONTROLLER DIAGNOSTICS
008C	CD13	263	INT 13H
008E	7303	264	JNC P7
0090	E9AF00	265	JMP ERROR_EX
0093		266	
0093	B80014	267	P7: MOV AX,1400H ; CONTROLLER DIAGNOSTICS
0096	CD13	268	INT 13H
0098	7303	269	JNC P9
009A	E9A500	270	JMP ERROR_EX
009D		271	
009D	C7066C000000	272	P9: MOV TIMER_LOW,0 ; ZERO TIMER
00A3	A17200	273	MOV AX,RESET_FLAG
00A6	303412	274	CMR AX,1234H ; KEYBOARD RESET
00A9	7506	275	JLIE P8
00AB	C7066C009A01	276	MOV TIMER_LOW,410D ; SNIP WAIT ON RESET
00B1		277	
00B1	E621	278	P8: IN AL,021H ; TIMER
00B3	24FE	279	AND AL,0FEH ; ENABLE TIMER
00B5	E621	280	OUT 021H,AL ; START TIMER
00B7		281	
00B7	E8B400	282	P4: CALL HD_RESET_1 ; RESET CONTROLLER
00BA	7207	283	JC P10
00BC	B80010	284	MOV AX,1000H ; READY
00BF	CD13	285	INT 13H
00C1	730B	286	JNC P2
00C3		287	
00C3	A16C00	288	P10: MOV AX,TIMER_LOW
00C6	30BE01	289	CMR AX,446D ; 25 SECONDS
00C9	72EC	290	JB P4
00CB	EB7590	291	JMP ERROR_EX
00CE		292	
00CE	B90100	293	P2: MOV CX,1
00D1	BA8000	294	MOV DX,80H
		295	
00D4	B80011	296	MOV AX,1100H ; RECALIBRATE
00D7	CD13	297	INT 13H
00D9	7267	298	JC ERROR_EX
		299	
00DB	B80009	300	MOV AX,0900H ; SET DRIVE PARAMETERS
00DE	CD13	301	INT 13H
00E0	7260	302	JC ERROR_EX
		303	
00E2	B800C8	304	MOV AX,0C800H ; DMA TO BUFFER

A-90 Fixed Disk BIOS

LOC OBJ	LINE	SOURCE	
00E5 8EC0	305	MOV	ES,AX ; SET SEGMENT
00E7 2B0B	306	SUB	BX,BX
00E9 B8000F	307	MOV	AX,0F00H ; WRITE SECTOR BUFFER
00EC CD13	308	INT	13H
00EE 7252	309	JC	ERROR_EX
	310		
00F0 FE067500	311	INC	HF_NUM ; DRIVE ZERO RESPONDED
	312		
00F4 BA1302	313	MOV	DX,213H ; EXPANSION BOX
00F7 B000	314	MOV	AL,0
00F9 EE	315	OUT	DX,AL ; TURN BOX OFF
00FA BA2103	316	MOV	DX,321H ; TEST IF CONTROLLER
00FD EC	317	IN	AL,DX ; ... IS IN THE SYSTEM UNIT
00FE 240F	318	AND	AL,0FH
0100 3C0F	319	CMP	AL,0FH
0102 7406	320	JE	BOX_ON
0104 C7066C00A401	321	MOV	TIMER_LOW,4200 ; CONTROLLER IS IN SYSTEM UNIT
010A	322		
010A BA1302	323	MOV	DX,213H ; EXPANSION BOX
010D B0FF	324	MOV	AL,0FFH
010F EE	325	OUT	DX,AL ; TURN BOX ON
	326		
0110 B90100	327	MOV	CX,1 ; ATTEMPT NEXT DRIVES
0113 BA8100	328	MOV	DX,081H
0116	329		
0116 2BC0	330	SUB	AX,AX ; RESET
0118 CD13	331	INT	13H
011A 7240	332	JC	POD_DONE
011C B90011	333	MOV	AX,01100H ; RECAL
011F CD13	334	INT	13H
0121 730B	335	JNC	P5
0123 A16C00	336	MOV	AX,TIMER_LOW
0126 30BE01	337	CMF	AX,446D ; 25 SECONDS
0129 72EB	338	JB	P3
012B ED2F90	339	JMP	POD_DONE
012E	340		
012E B80009	341	MOV	AX,0900H ; INITIALIZE CHARACTERISTICS
0131 CD13	342	INT	13H
0133 7227	343	JC	POD_DONE
0135 FE067500	344	INC	HF_NUM ; TALLY ANOTHER DRIVE
0139 81FA8100	345	CMF	DX,(60H + S_MAX_FILE - 1)
013D 731D	346	JAE	POD_DONE
013F 42	347	INC	DX
0140 EDD4	348	JMP	P3
	349		
	350		;----- POD ERROR
	351		
0142	352		ERROR_EX:
0142 BD0F00	353	MOV	BP,0FH ; POD ERROR FLAG
0145 2BC0	354	SUB	AX,AX
0147 8BF0	355	MOV	SI,AX
0149 B9060090	356	MOV	CX,F17L ; MESSAGE CHARACTER COUNT
014D B700	357	MOV	BH,0 ; PAGE ZERO
014F	358		OUT_CH:
014F 2E8A046801	359	MOV	AL,CS:F17[SI] ; GET BYTE
0154 B40E	360	MOV	AH,14D ; VIDEO OUT
0156 CD10	361	INT	10H ; DISPLAY CHARACTER
0158 46	362	INC	SI ; NEXT CHAR
0159 E2F4	363	LOOP	OUT_CH ; DO MORE
015B F9	364	STC	
015C	365		POD_DONE:
015C FA	366	CLI	
015D E421	367	IN	AL,021H ; BE SURE TIMER IS DISABLED
015F 0C01	368	OR	AL,01H
0161 E621	369	OUT	021H,AL
0163 FB	370	STI	
0164 E8A500	371	CALL	DSBL
0167 CB	372	RET	
	373		
0168 31373031	374	F17 DB	'1701',0DH,0AH

LOC OBJ	LINE	SOURCE
016C 0D		
016D 0A		
0006	375	F17L EQU 4-F17
	376	
016E	377	HD_RESET_1 PROC NEAR
016E 51	378	PUSH CX ; SAVE REGISTER
016F 52	379	PUSH DX
0170 F8	380	CLC ; CLEAR CARRY
0171 B90001	381	MOV CX,0100H ; RETRY COUNT
0174	382	L6:
0174 E80706	383	CALL PORT_1
0177 EE	384	OUT DX,AL ; RESET CARD
0178 E80306	385	CALL PORT_1
017B EC	386	IN AL,DX ; CHECK STATUS
017C 2402	387	AND AL,2 ; ERROR BIT
017E 7403	388	JZ R3
0180 E2F2	389	LOOP L6
0182 F9	390	STC
0183	391	R3:
0183 5A	392	POP DX ; RESTORE REGISTER
0184 59	393	POP CX
0185 C3	394	RET
	395	HD_RESET_1 ENDP
	396	
	397	DISK_SETUP ENDP
	398	
	399	----- INT 19 -----
	400	;
	401	; INTERRUPT 19 BOOT STRAP LOADER ;
	402	;
	403	; - THE FIXED DISK BIOS REPLACES THE INTERRUPT 19 ;
	404	; BOOT STRAP VECTOR WITH A POINTER TO THIS BOOT ROUTINE ;
	405	; - RESET THE DEFAULT DISK AND DISKETTE PARAMETER VECTORS ;
	406	; - THE BOOT BLOCK TO BE READ IN WILL BE ATTEMPTED FROM ;
	407	; CYLINDER 0 SECTOR 1 OF THE DEVICE. ;
	408	; - THE BOOTSTRAP SEQUENCE IS: ;
	409	; > ATTEMPT TO LOAD FROM THE DISKETTE INTO THE BOOT ;
	410	; LOCATION (0000:7C00) AND TRANSFER CONTROL THERE ;
	411	; > IF THE DISKETTE FAILS THE FIXED DISK IS TRIED FOR A ;
	412	; VALID BOOTSTRAP BLOCK. A VALID BOOT BLOCK ON THE ;
	413	; FIXED DISK CONSISTS OF THE BYTES 055H 0AAH AS THE ;
	414	; LAST TWO BYTES OF THE BLOCK ;
	415	; > IF THE ABOVE FAILS CONTROL IS PASSED TO RESIDENT BASIC ;
	416	;
	417	-----
	418	
0186	419	BOOT_STRAP:
	420	ASSUME DS:DUMMY,ES:DUMMY
0186 2BC0	421	SUB AX,AX
0188 8ED8	422	MOV DS,AX ; ESTABLISH SEGMENT
	423	
	424	----- RESET PARAMETER VECTORS
	425	
018A FA	426	CLI
018B C7060401E703	427	MOV WORD PTR HF_TBL_VEC, OFFSET FD_TBL
0191 8C0E0601	428	MOV WORD PTR HF_TBL_VEC+2, CS
0195 C70678000102	429	MOV WORD PTR DISKETTE_PARM, OFFSET DISKETTE_TBL
019B 8C0E7A00	430	MOV WORD PTR DISKETTE_PARM+2, CS
019F FB	431	STI
	432	
	433	----- ATTEMPT BOOTSTRAP FROM DISKETTE
	434	
01A0 B90300	435	MOV CX,3 ; SET RETRY COUNT
01A3	436	H1: ; IPL_SYSTEM
01A3 51	437	PUSH CX ; SAVE RETRY COUNT
01A4 2BD2	438	SUB DX,DX ; DRIVE ZERO
01A6 2BC0	439	SUB AX,AX ; RESET THE DISKETTE
01A8 CD13	440	INT 13H ; FILE IO CALL
01AA 720F	441	JC H2 ; IF ERROR, TRY AGAIN
01AC B80102	442	MOV AX,0201H ; READ IN THE SINGLE SECTOR
	443	
01AF 2BD2	444	SUB DX,DX
01B1 8EC2	445	MOV ES,DX ; ESTABLISH SEGMENT
01B3 BB007C	446	MOV BX,OFFSET BOOT_LOCN
	447	
01B6 B90100	448	MOV CX,1 ; SECTOR 1, TRACK 0
01B9 CD13	449	INT 13H ; FILE IO CALL

LOC OBJ	LINE	SOURCE	
01B0 59	450	H2: POP CX	; RECOVER RETRY COUNT
01BC 730A	451	JNC H4	; CF SET BY UNSUCCESSFUL READ
01BE 80FC80	452	CMP AH,80H	; IF TIME OUT, NO RETRY
01C1 740A	453	JZ H5	; TRY FIXED DISK
01C3 E2DE	454	LOOP H1	; DO IT FOR RETRY TIMES
01C5 E0690	455	JMP H5	; UNABLE TO IPL FROM THE DISKETTE
01C8	456		; IPL WAS SUCCESSFUL
01C8 EA007C0000	457	H4: JMP BOOT_LOCN	
	458		
	459	;----- ATTEMPT BOOTSTRAP FROM FIXED DISK	
	460		
01CD	461	H5:	
01CD 2BC0	462	SUB AX,AX	; RESET DISKETTE
01CF 2B02	463	SUB DX,DX	
01D1 CD13	464	INT 13H	
01D3 B90300	465	MOV CX,3	; SET RETRY COUNT
01D6	466	H6:	; IPL_SYSTEM
01D6 51	467	PUSH CX	; SAVE RETRY COUNT
01D7 BA8000	468	MOV DX,0080H	; FIXED DISK ZERO
01DA 2BC0	469	SUB AX,AX	; RESET THE FIXED DISK
01DC CD13	470	INT 13H	; FILE IO CALL
01DE 7212	471	JC H7	; IF ERROR, TRY AGAIN
01E0 B80102	472	MOV AX,0201H	; READ IN THE SINGLE SECTOR
01E3 2B0B	473	SUB BX,BX	
01E5 8EC3	474	MOV ES,BX	
01E7 B0007C	475	MOV BX,OFFSET BOOT_LOCN	; TO THE BOOT LOCATION
01EA BA8000	476	MOV DX,80H	; DRIVE NUMBER
01ED B90100	477	MOV CX,1	; SECTOR 1, TRACK 0
01F0 CD13	478	INT 13H	; FILE IO CALL
01F2 59	479	H7: POP CX	; RECOVER RETRY COUNT
01F3 720B	480	JC H8	
01F5 A1FE7D	481	MOV AX,WORD PTR BOOT_LOCN+510D	
01F6 3055AA	482	CMP AX,0AA55H	; TEST FOR GENERIC BOOT BLOCK
01FB 74CB	483	JZ H4	
01FD	484	H8:	
01FD E2D7	485	LOOP H6	; DO IT FOR RETRY TIMES
	486		
	487	;----- UNABLE TO IPL FROM THE DISKETTE OR FIXED DISK	
	488		
01FF CD18	489	INT 18H	; RESIDENT BASIC
	490		
0201	491	DISKETTE_TBL:	
	492		
0201 CF	493	DB 11001111B	; SRT=C, HD UNLOAD=0F - 1ST SPEC BYTE
0202 02	494	DB 2	; HD LOAD=1, MODE=DMA - 2ND SPEC BYTE
0203 25	495	DB 25H	; WAIT AFTER OPN TIL MOTOR OFF
0204 02	496	DB 2	; 512 BYTES PER SECTOR
0205 08	497	DB 8	; EOT (LAST SECTOR ON TRACK)
0206 2A	498	DB 02AH	; GAP LENGTH
0207 FF	499	DB 0FFH	; DTL
0208 50	500	DB 050H	; GAP LENGTH FOR FORMAT
0209 F6	501	DB 0F6H	; FILL BYTE FOR FORMAT
020A 19	502	DB 25	; HEAD SETTLE TIME (MILLISECONDS)
020B 04	503	DB 4	; MOTOR START TIME (1/8 SECOND)
	504		
	505	;----- MAKE SURE THAT ALL HOUSEKEEPING IS DONE BEFORE EXIT	
	506		
020C	507	DSBL PROC NEAR	
	508	ASSUME DS:DATA	
020C 1E	509	PUSH DS	; SAVE SEGMENT
020D B84000	510	MOV AX,DATA	
0210 8ED8	511	MOV DS,AX	
	512		
0212 8A267700	513	MOV AH,PORT_OFF	
0216 50	514	PUSH AX	; SAVE OFFSET
	515		
0217 C606770000	516	MOV PORT_OFF,0H	
021C E06905	517	CALL PORT_3	
021F 2AC0	518	SUB AL,AL	
0221 EE	519	OUT DX,AL	; RESET INT/DMA MASK
0222 C606770004	520	MOV PORT_OFF,AH	
0227 E05E05	521	CALL PORT_3	
022A 2AC0	522	SUB AL,AL	
022C EE	523	OUT DX,AL	; RESET INT/DMA MASK
022D C606770008	524	MOV PORT_OFF,8H	
0232 E05305	525	CALL PORT_3	
0235 2AC0	526	SUB AL,AL	

LOC	OBJ	LINE	SOURCE	
0237	EE	527	OUT	DX,AL ; RESET INT/DMA MASK
0238	C60677000C	528	MOV	PORT_OFF,0CH
023D	E84805	529	CALL	PORT_3
0240	2AC0	530	SUB	AL,AL
0242	EE	531	OUT	DX,AL ; RESET INT/DMA MASK
0243	B007	532	MOV	AL,07H
0245	E60A	533	OUT	DMA+10,AL ; SET DMA MODE TO DISABLE
0247	FA	534	CLI	; DISABLE INTERRUPTS
0248	E421	535	IN	AL,021H
024A	0C20	536	OR	AL,020H
024C	E621	537	OUT	021H,AL ; DISABLE INTERRUPT 5
024E	FB	538	STI	; ENABLE INTERRUPTS
024F	58	539	POP	AX ; RESTORE OFFSET
0250	88267700	540	MOV	PORT_OFF,AH
0254	1F	541	POP	DS ; RESTORE SEGMENT
0255	C3	542	RET	
		543	DSBL	ENDP
		544		
		545		-----
		546		; FIXED DISK BIOS ENTRY POINT ;
		547		-----
		548		
0256		549	DISK_IO PROC	FAR
		550	ASSUME	DS:NOTHING,ES:NOTHING
0256	80FA80	551	CHP	DL,80H ; TEST FOR FIXED DISK DRIVE
0259	7305	552	JAE	HARD_DISK ; YES, HANDLE HERE
025B	CD40	553	INT	40H ; DISKETTE HANDLER
025D		554	RET_2:	
025D	CA0200	555	RET	2 ; BACK TO CALLER
0260		556	HARD_DISK:	
		557	ASSUME	DS:DATA
0260	FB	558	STI	; ENABLE INTERRUPTS
0261	0AE4	559	OR	AH,AH
0263	7509	560	JNZ	A3
0265	CD40	561	INT	40H ; RESET NEC WHEN AH=0
0267	2AE4	562	SUB	AH,AH
0269	80FA81	563	CHP	DL,(80H + S_MAX_FILE - 1)
026C	77EF	564	JA	RET_2
026E		565	A3:	
026E	80FC08	566	CHP	AH,08 ; GET PARAMETERS IS A SPECIAL CASE
0271	7503	567	JNZ	A2
0273	E91A01	568	JMP	GET_PARM_N
0276		569	A2:	
0276	53	570	PUSH	BX ; SAVE REGISTERS DURING OPERATION
0277	51	571	PUSH	CX
0278	52	572	PUSH	DX
0279	1E	573	PUSH	DS
027A	06	574	PUSH	ES
027B	56	575	PUSH	SI
027C	57	576	PUSH	DI
		577		
027D	E86A00	578	CALL	DISK_IO_CONT ; PERFORM THE OPERATION
		579		
0280	50	580	PUSH	AX
0281	E880FF	581	CALL	DSBL ; BE SURE DISABLES OCCURRED
0284	B84000	582	MOV	AX,DATA
0287	8ED8	583	MOV	DS,AX ; ESTABLISH SEGMENT
0289	58	584	POP	AX
028A	BA267400	585	MOV	AH,DISK_STATUS ; GET STATUS FROM OPERATION
028E	80FC01	586	CHP	AH,1 ; SET THE CARRY FLAG TO INDICATE
0291	F5	587	CMC	; SUCCESS OR FAILURE
0292	5F	588	POP	DI ; RESTORE REGISTERS
0293	5E	589	POP	SI
0294	07	590	POP	ES
0295	1F	591	POP	DS
0296	5A	592	POP	DX
0297	59	593	POP	CX
0298	5B	594	POP	BX
0299	CA0200	595	RET	2 ; THROW AWAY SAVED FLAGS
		596	DISK_IO ENDP	
		597		
029C		598	M1	LABEL WORD ; FUNCTION TRANSFER TABLE
029C	3003	599	DW	DISK_RESET ; 000H
029E	4D03	600	DW	RETURN_STATUS ; 001H
02A0	5603	601	DW	DISK_READ ; 002H
02A2	6003	602	DW	DISK_WRITE ; 003H
02A4	6A03	603	DW	DISK_VRF ; 004H

LOC OBJ	LINE	SOURCE	
02A6 7203	604	DW FMT_TRK	; 005H
02A8 7903	605	DW FMT_BAD	; 006H
02AA 8003	606	DW FMT_DRV	; 007H
02AC 3003	607	DW BAD_COMMAND	; 008H
02AE 2704	608	DW INIT_DRV	; 009H
02B0 CF04	609	DW RD_LONG	; 00AH
02B2 D004	610	DW WR_LONG	; 00BH
02B4 F204	611	DW DISK_SEEK	; 00CH
02B6 3803	612	DW DISK_RESET	; 00DH
02B8 F904	613	DW RD_BUFF	; 00EH
02BA 0705	614	DW WR_BUFF	; 00FH
02BC 1505	615	DW TST_RDY	; 010H
02BE 1C05	616	DW HDISK_RECAL	; 011H
02C0 2305	617	DW RAM_DIAG	; 012H
02C2 2A05	618	DW CHK_DRV	; 013H
02C4 3105	619	DW CHTLR_DIAG	; 014H
002A	620	MIL EQU 6-M1	
	621		
02C6	622	SETUP_A PROC NEAR	
	623		
02C6 C606740000	624	MOV DISK_STATUS,0	; RESET THE STATUS INDICATOR
02CB 51	625	PUSH CX	; SAVE CX
	626		
	627	;----- CALCULATE THE PORT OFFSET	
	628		
02CC 8AEA	629	MOV CH,DL	; SAVE DL
02CE 80CA01	630	OR DL,1	
02D1 FECA	631	DEC DL	
02D3 D0E2	632	SHL DL,1	; GENERATE OFFSET
02D5 8B167700	633	MOV PORT_OFF,DL	; STORE OFFSET
02D9 8AD5	634	MOV DL,CH	; RESTORE DL
02DB 80E201	635	AND DL,1	
	636		
02DE B105	637	MOV CL,5	; SHIFT COUNT
02E0 D2E2	638	SHL DL,CL	; DRIVE NUMBER (0,1)
02E2 0AD6	639	OR DL,DH	; HEAD NUMBER
02E4 8B164300	640	MOV CHD_BLOCK+1,DL	
02E8 59	641	POP CX	
02E9 C3	642	RET	
	643	SETUP_A ENDP	
	644		
02EA	645	DISK_IO_COUNT PROC NEAR	
02EA 50	646	PUSH AX	
02EB B04000	647	MOV AX,DATA	
02EE 8ED8	648	MOV DS,AX	; ESTABLISH SEGMENT
02F0 58	649	POP AX	
02F1 80FC01	650	CMF AH,01H	; RETURN STATUS
02F4 7503	651	JNZ A4	
02F6 EB5590	652	JMP RETURN_STATUS	
02F9	653	A4:	
02F9 80EA80	654	SUB DL,80H	; CONVERT DRIVE NUMBER TO 0 BASED RANGE
02FC 80FA08	655	CMF DL,MAX_FILE	; LEGAL DRIVE TEST
02FF 732F	656	JAE BAD_COMMAND	
	657		
0301 E8C2FF	658	CALL SETUP_A	
	659		
	660	;----- SET UP COMMAND BLOCK	
	661		
0304 FEC9	662	DEC CL	; SECTORS 0-16 FOR CONTROLLER
0306 C606420000	663	MOV CHD_BLOCK+0,0	
0308 880E4400	664	MOV CHD_BLOCK+2,CL	; SECTOR AND HIGH 2 BITS CYLINDER
030F 882E4500	665	MOV CHD_BLOCK+3,CH	; CYLINDER
0313 A24600	666	MOV CHD_BLOCK+4,AL	; INTERLEAVE / BLOCK COUNT
0316 A07600	667	MOV AL,CONTROL_BYTE	; CONTROL BYTE (STEP OPTION)
0319 A24700	668	MOV CHD_BLOCK+5,AL	
031C 50	669	PUSH AX	; SAVE AX
031D 8AC4	670	MOV AL,AH	; GET INTO LOW BYTE
031F 32E4	671	XOR AH,AH	; ZERO HIGH BYTE
0321 D1E0	672	SAL AX,1	; #2 FOR TABLE LOOKUP
0323 8BF0	673	MOV SI,AX	; PUT INTO SI FOR BRANCH
0325 3D2A00	674	CMF AX,M1L	; TEST WITHIN RANGE
0328 58	675	POP AX	; RESTORE AX
0329 7305	676	JNB BAD_COMMAND	
032B 2EFFA49C02	677	JMP WORD PTR CS:[SI + OFFSET M1]	
0330	678	BAD_COMMAND:	
0330 C606740001	679	MOV DISK_STATUS,BAD_CHD	; COMMAND ERROR
0335 B000	680	MOV AL,0	

LOC	OBJ	LINE	SOURCE
0337	C3	681	RET
		682	DISK_IO_CONT ENDP
		683	
		684	;
		685	; RESET THE DISK SYSTEM (AH = 000H) :
		686	;
		687	
0338		688	DISK_RESET PROC NEAR
0338 E84304		689	CALL PORT_1 ; RESET PORT
0338 EE		690	OUT DX,AL ; ISSUE RESET
033C E83F04		691	CALL PORT_1 ; CONTROLLER HARDWARE STATUS
033F EC		692	IN AL,DX ; GET STATUS
0340 2402		693	AND AL,2 ; ERROR BIT
0342 7406		694	JZ DR1
0344 C606740005		695	MOV DISK_STATUS,BAD_RESET
0349 C3		696	RET
034A		697	DR1:
034A E90A00		698	JMP INIT_DRV ; SET THE DRIVE PARAMETERS
		699	DISK_RESET ENDP
		700	
		701	;
		702	; DISK STATUS ROUTINE (AH = 001H) :
		703	;
		704	
034D		705	RETURN_STATUS PROC NEAR
034D A07400		706	MOV AL,DISK_STATUS ; OBTAIN PREVIOUS STATUS
0350 C606740000		707	MOV DISK_STATUS,0 ; RESET STATUS
0355 C3		708	RET
		709	RETURN_STATUS ENDP
		710	
		711	;
		712	; DISK READ ROUTINE (AH = 002H) :
		713	;
		714	
0356		715	DISK_READ PROC NEAR
0356 B047		716	MOV AL,DMA_READ ; MODE BYTE FOR DMA READ
0358 C606420008		717	MOV CHD_BLOCK+0,READ_CHD
035D E9E501		718	JMP DMA_OPN
		719	DISK_READ ENDP
		720	
		721	;
		722	; DISK WRITE ROUTINE (AH = 003H) :
		723	;
		724	
0360		725	DISK_WRITE PROC NEAR
0360 B04B		726	MOV AL,DMA_WRITE ; MODE BYTE FOR DMA WRITE
0362 C60642000A		727	MOV CHD_BLOCK+0,WRITE_CHD
0367 E90B01		728	JMP DMA_OPN
		729	DISK_WRITE ENDP
		730	
		731	;
		732	; DISK VERIFY (AH = 004H) :
		733	;
		734	
036A		735	DISK_VERF PROC NEAR
036A C606420005		736	MOV CHD_BLOCK+0,CHK_TRK_CHD
036F E9C401		737	JMP NDHIA_OPN
		738	DISK_VERF ENDP
		739	
		740	;
		741	; FORMATTING (AH = 005H 006H 007H) :
		742	;
		743	
0372		744	FMT_TRK PROC NEAR ; FORMAT TRACK (AH = 005H)
0372 C606420006		745	MOV CHD_BLOCK,FMTTRK_CHD
0377 EB0C		746	JMP SHORT FMT_CONT
		747	FMT_TRK ENDP
		748	
0379		749	FMT_BAD PROC NEAR ; FORMAT BAD TRACK (AH = 006H)
0379 C606420007		750	MOV CHD_BLOCK,FMTBAD_CHD
037E EB05		751	JMP SHORT FMT_CONT
		752	FMT_BAD ENDP
		753	
0380		754	FMT_DRV PROC NEAR ; FORMAT DRIVE (AH = 007H)
0380 C606420004		755	MOV CHD_BLOCK,FMTDRV_CHD
		756	FMT_DRV ENDP
		757	

```

LOC OBJ          LINE  SOURCE

0385             758  FHT_CONT:
0385 A04400       759      MOV     AL,CHD_BLOCK+2      ; ZERO OUT SECTOR FIELD
0388 24C0         760      AND     AL,11000000B
038A A24400       761      MOV     CHD_BLOCK+2,AL
038D E9A601       762      JMP     NDMA_OPM
038D E9A601       763
038D E9A601       764
038D E9A601       765  ;-----
038D E9A601       765  ;      GET PARAMETERS   (AH = 8)      ;
038D E9A601       766  ;-----
038D E9A601       767

0390             768  GET_PARM_H   LABEL  NEAR
0390             769  GET_PARM   PROC    FAR      ; GET DRIVE PARAMETERS
0390 1E           770      PUSH   DS          ; SAVE REGISTERS
0391 06           771      PUSH   ES
0392 53           772      PUSH   BX
0392 53           773
0392 53           774      ASSUME  DS:DUPM1
0393 2BC0         775      SUB     AX,AX      ; ESTABLISH ADDRESSING
0395 8ED8         776      MOV     DS,AX
0397 C41E0401     777      LES     BX,HF_TBL_VEC
0397 C41E0401     778      ASSUME  DS:DATA
039B B84000       779      MOV     AX,DATA
039E 8ED8         780      MOV     DS,AX      ; ESTABLISH SEGMENT
039E 8ED8         781
03A0 80EA00       782      SUB     DI,80H
03A3 80FA08       783      CMP     DI,MAX_FILE      ; TEST WITHIN RANGE
03A6 732F         784      JAE     G4
03A6 732F         785
03A8 E81BFF       786      CALL    SETUP_A
03A8 E81BFF       787
03AB E8DF03       788      CALL    SW2_OFFS
03AE 7227         789      JC      G4
03B0 03D8         790      ADD     BX,AX
03B0 03D8         791
03B2 268B07       792      MOV     AX,ES:[BX]      ; MAX NUMBER OF CYLINDERS
03B5 2D0200       793      SUB     AX,2      ; ADJUST FOR 0-N
03B5 2D0200       794      ; AND RESERVE LAST TRACK
03B8 8AE8         795      MOV     CH,AL
03BA 250003       796      AND     AX,0300H      ; HIGH TWO BITS OF CYL
03BD D1E8         797      SHR     AX,1
03BF D1E8         798      SHR     AX,1
03C1 0C11         799      OR      AL,011H      ; SECTORS
03C3 8AC8         800      MOV     CL,AL
03C3 8AC8         801
03C5 268A7702     802      MOV     DH,ES:[BX+2]      ; HEADS
03C9 FECE         803      DEC     DH      ; 0-N RANGE
03CB 8A167500     804      MOV     DL,HF_NUM      ; DRIVE COUNT
03CF 2BC0         805      SUB     AX,AX
03D1             806  GS:
03D1 5B           807      POP     BX      ; RESTORE REGISTERS
03D2 07           808      POP     ES
03D3 1F           809      POP     DS
03D4 CA0200       810      RET     2
03D7             811  G4:
03D7 C06740007    812      MOV     DISK_STATUS,INIT_FAIL ; OPERATION FAILED
03DC B407         813      MOV     AH,INIT_FAIL
03DE 2AC0         814      SUB     AL,AL
03E0 2B02         815      SUB     DX,DX
03E2 2BC9         816      SUB     CX,CX
03E4 F9           817      STC      ; SET ERROR FLAG
03E5 EBEA         818      JMP     G5
03E5 EBEA         819  GET_PARM   ENDP
03E5 EBEA         820
03E5 EBEA         821  ;-----
03E5 EBEA         822  ; INITIALIZE DRIVE CHARACTERISTICS ;
03E5 EBEA         823  ; ;
03E5 EBEA         824  ; FIXED DISK PARAMETER TABLE ;
03E5 EBEA         825  ; ;
03E5 EBEA         826  ; - THE TABLE IS COMPOSED OF A BLOCK DEFINED AS: ;
03E5 EBEA         827  ; ;
03E5 EBEA         828  ; (1 WORD) - MAXIMUM NUMBER OF CYLINDERS ;
03E5 EBEA         829  ; (1 BYTE) - MAXIMUM NUMBER OF HEADS ;
03E5 EBEA         830  ; (1 WORD) - STARTING REDUCED WRITE CURRENT CYL ;
03E5 EBEA         831  ; (1 WORD) - STARTING WRITE PRECOMPENSATION CYL ;
03E5 EBEA         832  ; (1 BYTE) - MAXIMUM ECC DATA BURST LENGTH ;
03E5 EBEA         833  ; (1 BYTE) - CONTROL BYTE (DRIVE STEP OPTION) ;
03E5 EBEA         834  ; BIT 7 DISABLE DISK-ACCESS RETRIES ;
03E5 EBEA         835  ; BIT 6 DISABLE ECC RETRIES ;

```



```

836 ;          BITS 5-3 ZERO          ;
837 ;          BITS 2-0 DRIVE OPTION  ;
838 ;          (1 BYTE) - STANDARD TIME OUT VALUE (SEE BELOW) ;
839 ;          (1 BYTE) - TIME OUT VALUE FOR FORMAT DRIVE    ;
840 ;          (1 BYTE) - TIME OUT VALUE FOR CHECK DRIVE     ;
841 ;          (4 BYTES)                                     ;
842 ;          - RESERVED FOR FUTURE USE                     ;
843 ;                                                         ;
844 ;          - TO DYNAMICALLY DEFINE A SET OF PARAMETERS   ;
845 ;          BUILD A TABLE OF VALUES AND PLACE THE       ;
846 ;          CORRESPONDING VECTOR INTO INTERRUPT 41.       ;
847 ;                                                         ;
848 ;          NOTE:                                           ;
849 ;          THE DEFAULT TABLE IS VECTORED IN FOR         ;
850 ;          AN INTERRUPT 19H (BOOTSTRAP)                   ;
851 ;                                                         ;
852 ;                                                         ;
853 ; ON THE CAPD SWITCH SETTINGS                             ;
854 ;                                                         ;
855 ;          DRIVE 0   DRIVE 1                               ;
856 ;          -----   -----                               ;
857 ;          ON       : / :                                   ;
858 ;          : -1-   -2- / -3- -4- :                         ;
859 ;          OFF      : / :                                   ;
860 ;          -----   -----                               ;
861 ;                                                         ;
862 ;                                                         ;
863 ;          TRANSLATION TABLE                             ;
864 ;                                                         ;
865 ;          1/3 : 2/4 : TABLE ENTRY                       ;
866 ;          -----   -----                               ;
867 ;          ON : ON : 0                                     ;
868 ;          ON : OFF : 1                                    ;
869 ;          OFF : ON : 2                                    ;
870 ;          OFF : OFF : 3                                   ;
871 ;                                                         ;
872 ; -----
873
03E7 874 FD_TBL:
875
876 ;----- DRIVE TYPE 00
877
03E7 878 DW 0306D
03E9 879 DB 02D
03EA 880 DW 0306D
03EC 881 DW 0000D
03EE 882 DB 0BH
03EF 883 DB 00H
03F0 884 DB 0CH ; STANDARD
03F1 885 DB 0B4H ; FORMAT DRIVE
03F2 886 DB 028H ; CHECK DRIVE
03F3 887 DB 0,0,0,0
888
889 ;----- DRIVE TYPE 01
890
03F7 891 DW 0375D
03F9 892 DB 05D
03FA 893 DW 0375D
03FC 894 DW 0000D
03FE 895 DB 0BH
03FF 896 DB 05H
0400 897 DB 0CH ; STANDARD
0401 898 DB 0B4H ; FORMAT DRIVE
0402 899 DB 028H ; CHECK DRIVE
0403 900 DB 0,0,0,0
901
902 ;----- DRIVE TYPE 02
903
0407 904 DW 0306D
0409 905 DB 06D
040A 906 DW 0128D
040C 907 DW 0256D
040E 908 DB 0BH
040F 909 DB 05H
0410 910 DB 0CH ; STANDARD
0411 911 DB 0B4H ; FORMAT DRIVE

```

LOC OBJ	LINE	SOURCE
0412 20	912	DB 020H ; CHECK DRIVE
0413 00000000	913	DB 0,0,0,0
	914	
	915	;----- DRIVE TYPE 03
	916	
0417 3201	917	DW 0306D
0419 04	918	DB 04D
041A 3201	919	DW 0306D
041C 0000	920	DW 0000D
041E 0B	921	DB 0BH
041F 05	922	DB 05H
0420 0C	923	DB 0CH ; STANDARD
0421 B4	924	DB 0B4H ; FORMAT DRIVE
0422 28	925	DB 028H ; CHECK DRIVE
0423 00000000	926	DB 0,0,0,0
	927	
0427	928	INIT_DRV PROC NEAR
	929	
	930	;----- DO DRIVE ZERO
	931	
0427 C0642000C	932	MOV CHD_BLOCK+0,INIT_DRV_CHD
042C C06430000	933	MOV CHD_BLOCK+1,0
0431 E81000	934	CALL INIT_DRV_R
0434 720D	935	JC INIT_DRV_OUT
	936	
	937	;----- DO DRIVE ONE
	938	
0436 C0642000C	939	MOV CHD_BLOCK+0,INIT_DRV_CHD
043B C06430020	940	MOV CHD_BLOCK+1,00100000B
0440 E80100	941	CALL INIT_DRV_R
0443	942	INIT_DRV_OUT:
0443 C3	943	RET
	944	INIT_DRV ENDP
	945	
0444	946	INIT_DRV_R PROC NEAR
	947	ASSUME ES:CODE
0444 2AC0	948	SUB AL,AL
0446 E81901	949	CALL COMMAND ; ISSUE THE COMMAND
0449 7301	950	JNC B1
044B C3	951	RET
044C	952	B1:
044C 1E	953	PUSH DS ; SAVE SEGMENT
	954	ASSUME DS:DUP1
044D 2BC0	955	SUB AX,AX
044F 8ED8	956	MOV DS,AX ; ESTABLISH SEGMENT
0451 C41E0401	957	LES BX,HF_TBL_VEC
0455 1F	958	POP DS ; RESTORE SEGMENT
	959	ASSUME DS:DATA
0456 E83403	960	CALL SW2_OFFS
0459 7257	961	JC B3
045B 03D8	962	ADD BX,AX
	963	
	964	;----- SEND DRIVE PARAMETERS MOST SIGNIFICANT BYTE FIRST
	965	
045D BF0100	966	MOV DI,1
0460 E85F00	967	CALL INIT_DRV_S
0463 724D	968	JC B3
	969	
0465 BF0000	970	MOV DI,0
0468 E85700	971	CALL INIT_DRV_S
046B 7245	972	JC B3
	973	
046D BF0200	974	MOV DI,2
0470 E84F00	975	CALL INIT_DRV_S
0473 7230	976	JC B3
	977	
0475 BF0400	978	MOV DI,4
0478 E84700	979	CALL INIT_DRV_S
047B 7235	980	JC B3
	981	
047D BF0300	982	MOV DI,3
0480 E83F00	983	CALL INIT_DRV_S
0483 722D	984	JC B3
	985	
0485 BF0600	986	MOV DI,6
0488 E83700	987	CALL INIT_DRV_S
048B 7225	988	JC B3

LOC OBJ	LINE	SOURCE
	989	
0480 BF0500	990	MOV DI,5
0490 E82F00	991	CALL INIT_DRV_S
0493 721D	992	JC B3
	993	
0495 BF0700	994	MOV DI,7
0498 E82700	995	CALL INIT_DRV_S
049B 7215	996	JC B3
	997	
049D BF0800	998	MOV DI,8
04A0 268A01	999	MOV AL,ES:[BX + DI] ; DRIVE STEP OPTION
04A3 A27600	1000	MOV CONTROL_BYTE,AL
	1001	
04A6 2BC9	1002	SUB CX,CX
04A8	1003	B5:
04AB E8D302	1004	CALL PORT_1
04AB EC	1005	IN AL,DX
04AC A802	1006	TEST AL,R1_TOMODE ; STATUS INPUT MODE
04AE 7509	1007	JNZ B6
04B0 E2F6	1008	LOOP B5
04B2	1009	B3:
04B2 C606740007	1010	MOV DISK_STATUS,INIT_FAIL ; OPERATION FAILED
04B7 F9	1011	STC
04B8 C3	1012	RET
	1013	
04B9	1014	B6:
04B9 E8B502	1015	CALL PORT_0
04BC EC	1016	IN AL,DX
04BD 2402	1017	AND AL,2 ; MASK ERROR BIT
04BF 75F1	1018	JNZ B3
04C1 C3	1019	RET
	1020	ASSUME ES:NOTHING
	1021	INIT_DRV_R ENDP
	1022	
	1023	;----- SEND THE BYTE OUT TO THE CONTROLLER
	1024	
04C2	1025	INIT_DRV_S PROC NEAR
04C2 E8C501	1026	CALL HD_WAIT_REQ
04C5 7207	1027	JC D1
04C7 E8A702	1028	CALL PORT_0
04CA 268A01	1029	MOV AL,ES:[BX + DI]
04CD EE	1030	OUT DX,AL
04CE	1031	D1:
04CE C3	1032	RET
	1033	INIT_DRV_S ENDP
	1034	
	1035	;------
	1036	; READ LONG (AH = 0AH) ;
	1037	;------
	1038	
04CF	1039	RD_LONG PROC NEAR
04CF E81900	1040	CALL CHK_LONG
04D2 726B	1041	JC G8
04D4 C6064200E5	1042	MOV CMD_BLOCK+0,RD_LONG_CMD
04D9 B047	1043	MOV AL,DMA_READ
04DB EB68	1044	JMP SHORT DMA_OPN
	1045	RD_LONG ENDP
	1046	
	1047	;------
	1048	; WRITE LONG (AH = 0BH) ;
	1049	;------
	1050	
04DD	1051	WR_LONG PROC NEAR
04DD E80B00	1052	CALL CHK_LONG
04E0 725D	1053	JC G8
04E2 C6064200E6	1054	MOV CMD_BLOCK+0,WR_LONG_CMD
04E7 B04B	1055	MOV AL,DMA_WRITE
04E9 EB5A	1056	JMP SHORT DMA_OPN
	1057	WR_LONG ENDP
	1058	
04EB	1059	CHK_LONG PROC NEAR
04EB A04600	1060	MOV AL,CMD_BLOCK+4
04EE 3C80	1061	CMPI AL,080H
04F0 F5	1062	CMC
04F1 C3	1063	RET
	1064	CHK_LONG ENDP
	1065	

LOC	OBJ	LINE	SOURCE
		1066	-----
		1067	; SEEK (AH = 0CH) ;
		1068	-----
		1069	
04F2		1070	DISK_SEEK PROC NEAR
04F2 C064200B		1071	MOV CHD_BLOCK, SEEK_CHD
04F7 EB3D		1072	JMP SHORT NDMA_OPN
		1073	DISK_SEEK ENDP
		1074	
		1075	-----
		1076	; READ SECTOR BUFFER (AH = 0EH) ;
		1077	-----
		1078	
04F9		1079	RD_BUFF PROC NEAR
04F9 C064200E		1080	MOV CHD_BLOCK+0, RD_BUFF_CHD
04FE C06460001		1081	MOV CHD_BLOCK+4, 1 ; ONLY ONE BLOCK
0503 B047		1082	MOV AL, DMA_READ
0505 EB3E		1083	JMP SHORT DMA_OPN
		1084	RD_BUFF ENDP
		1085	
		1086	-----
		1087	; WRITE SECTOR BUFFER (AH = 0FH) ;
		1088	-----
		1089	
0507		1090	WR_BUFF PROC NEAR
0507 C064200F		1091	MOV CHD_BLOCK+0, WR_BUFF_CHD
050C C06460001		1092	MOV CHD_BLOCK+4, 1 ; ONLY ONE BLOCK
0511 B04B		1093	MOV AL, DMA_WRITE
0513 EB3D		1094	JMP SHORT DMA_OPN
		1095	WR_BUFF ENDP
		1096	
		1097	-----
		1098	; TEST DISK READY (AH = 010H) ;
		1099	-----
		1100	
0515		1101	TST_RDY PROC NEAR
0515 C0642000		1102	MOV CHD_BLOCK+0, TST_RDY_CHD
051A EB1A		1103	JMP SHORT NDMA_OPN
		1104	TST_RDY ENDP
		1105	
		1106	-----
		1107	; RECALIBRATE (AH = 011H) ;
		1108	-----
		1109	
051C		1110	HDISK_RECAL PROC NEAR
051C C06420001		1111	MOV CHD_BLOCK, RECAL_CHD
0521 EB13		1112	JMP SHORT NDMA_OPN
		1113	HDISK_RECAL ENDP
		1114	
		1115	-----
		1116	; CONTROLLER RAM DIAGNOSTICS (AH = 012H) ;
		1117	-----
		1118	
0523		1119	RAM_DIAG PROC NEAR
0523 C064200E6		1120	MOV CHD_BLOCK+0, RAM_DIAG_CHD
0528 EB0C		1121	JMP SHORT NDMA_OPN
		1122	RAM_DIAG ENDP
		1123	
		1124	-----
		1125	; DRIVE DIAGNOSTICS (AH = 013H) ;
		1126	-----
		1127	
052A		1128	CHK_DRV PROC NEAR
052A C064200E3		1129	MOV CHD_BLOCK+0, CHK_DRV_CHD
052F EB05		1130	JMP SHORT NDMA_OPN
		1131	CHK_DRV ENDP
		1132	
		1133	-----
		1134	; CONTROLLER INTERNAL DIAGNOSTICS (AH = 014H) ;
		1135	-----
		1136	
0531		1137	CNTRLR_DIAG PROC NEAR
0531 C064200E4		1138	MOV CHD_BLOCK+0, CNTRLR_DIAG_CHD
		1139	CNTRLR_DIAG ENDP
		1140	

LOC OBJ	LINE	SOURCE
	1141	-----
	1142	SUPPORT ROUTINES
	1143	-----
	1144	
0536	1145	NDMA_OPN:
0536 B002	1146	MOV AL,02H
0538 E82700	1147	CALL COMMAND ; ISSUE THE COMMAND
0538 7221	1148	JC G11
0530 EB16	1149	JMP SHORT G3
053F	1150	G3:
053F C606740009	1151	MOV DISK_STATUS,DMA_BOUNDARY
0544 C3	1152	RET
0545	1153	DMA_OPN:
0545 E85701	1154	CALL DMA_SETUP ; SET UP FOR DMA OPERATION
0548 72F5	1155	JC G8
054A B003	1156	MOV AL,03H
054C E81300	1157	CALL COMMAND ; ISSUE THE COMMAND
054F 7200	1158	JC G11
0551 B003	1159	MOV AL,03H
0553 E60A	1160	OUT DMA+10,AL ; INITIALIZE THE DISK CHANNEL
0555	1161	G3:
0555 E421	1162	IN AL,021H
0557 24DF	1163	AND AL,0DFH
0559 E621	1164	OUT 021H,AL
055B E8AA01	1165	CALL WAIT_INT
055E	1166	G11:
055E E83B00	1167	CALL ERROR_CHK
0561 C3	1168	RET
	1169	
	1170	-----
	1171	COMMAND
	1172	THIS ROUTINE OUTPUTS THE COMMAND BLOCK
	1173	INPUT
	1174	AL = CONTROLLER DMA/INTERRUPT REGISTER MASK
	1175	
	1176	-----
	1177	
0562	1178	COMMAND PROC NEAR
0562 BE4200	1179	MOV SI,OFFSET CMD_BLOCK
0565 E81B02	1180	CALL PORT_2
0568 EE	1181	OUT DX,AL ; CONTROLLER SELECT PULSE
0569 E81C02	1182	CALL PORT_3
056C EE	1183	OUT DX,AL
056D 2BC9	1184	SUB CX,CX ; WAIT COUNT
056F E80C02	1185	CALL PORT_1
0572	1186	WAIT_BUSY:
0572 EC	1187	IN AL,DX ; GET STATUS
0573 240F	1188	AND AL,0FH
0575 3C00	1189	CMPL AL,R1_BUSY OR R1_BUS OR R1_REQ
0577 7409	1190	JE C1
0579 E2F7	1191	LOOP WAIT_BUSY
057B C606740080	1192	MOV DISK_STATUS,TIME_OUT
0580 F9	1193	STC
0581 C3	1194	RET ; ERROR RETURN
0582	1195	C1:
0582 FC	1196	CLD
0583 B90600	1197	MOV CX,6 ; BYTE COUNT
0586	1198	CH3:
0586 E8E801	1199	CALL PORT_0
0589 AC	1200	LODSB ; GET THE NEXT COMMAND BYTE
058A EE	1201	OUT DX,AL ; OUT IT GOES
058B E2F9	1202	LOOP CH3 ; DO MORE
	1203	
058D E8EE01	1204	CALL PORT_1 ; STATUS
0590 EC	1205	IN AL,DX
0591 A801	1206	TEST AL,R1_REQ
0593 7406	1207	JZ CH7
0595 C606740020	1208	MOV DISK_STATUS,BAD_CHTLR
059A F9	1209	STC
059B	1210	CH7:
059B C3	1211	RET
	1212	COMMAND ENDP
	1213	
	1214	-----
	1215	SENSE STATUS BYTES
	1216	
	1217	BYTE 0

LOC OBJ

LINE SOURCE

```

1218 | BIT 7 ADDRESS VALID, WHEN SET |
1219 | BIT 6 SPARE, SET TO ZERO |
1220 | BITS 5-4 ERROR TYPE |
1221 | BITS 3-0 ERROR CODE |
1222 | |
1223 | BYTE 1 |
1224 | BITS 7-6 ZERO |
1225 | BIT 5 DRIVE (0-1) |
1226 | BITS 4-0 HEAD NUMBER |
1227 | |
1228 | BYTE 2 |
1229 | BITS 7-5 CYLINDER HIGH |
1230 | BITS 4-0 SECTOR NUMBER |
1231 | |
1232 | BYTE 3 |
1233 | BITS 7-0 CYLINDER LOW |
1234 | |
1235 |-----|
1236 |
059C 1237 ERROR_CHK PROC NEAR
1238 ASSUME ES:DATA
059C A07400 1239 MOV AL,DISK_STATUS ; CHECK IF THERE WAS AN ERROR
059F 0AC0 1240 OR AL,AL
05A1 7501 1241 JNZ G21
05A3 C3 1242 RET
1243 |
1244 |----- PERFORM SENSE STATUS
1245 |
05A4 1246 G21:
05A4 B04000 1247 MOV AX,DATA
05A7 8EC0 1248 MOV ES,AX ; ESTABLISH SEGMENT
05A9 2BC0 1249 SUB AX,AX
05AB 6BF0 1250 MOV DI,AX
05AD C06420003 1251 MOV CHD_BLOCK+0,SENSE_CHD
05B2 2AC0 1252 SUB AL,AL
05B4 E0ABFF 1253 CALL COMMAND ; ISSUE SENSE STATUS COMMAND
05B7 7223 1254 JC SENSE_ABORT ; CANNOT RECOVER
05B9 B90400 1255 MOV CX,4
05BC 1256 G22:
05BC E0CB00 1257 CALL HD_WAIT_REQ
05BF 7220 1258 JC G24
05C1 E0AD01 1259 CALL PORT_0
05C4 EC 1260 IN AL,DX
05C5 26884542 1261 MOV ES:HD_ERROR(DI),AL ; STORE AWAY SENSE BYTES
05C9 47 1262 INC DI
05CA E0B101 1263 CALL PORT_1
05CD E2ED 1264 LOOP G22
05CF E0B000 1265 CALL HD_WAIT_REQ
05D2 72D0 1266 JC G24
05D4 E09A01 1267 CALL PORT_0
05D7 EC 1268 IN AL,DX
05D8 A802 1269 TEST AL,2
05DA 740F 1270 JZ STAT_ERR
05DC 1271 SENSE_ABORT:
05DC C067400FF 1272 MOV DISK_STATUS,SENSE_FAIL
05E1 1273 G24:
05E1 F9 1274 STC
05E2 C3 1275 RET
1276 ERROR_CHK ENDP
1277 |
05E3 1A06 1278 T_0 DW TYPE_0
05E5 2706 1279 T_1 DW TYPE_1
05E7 6A06 1280 T_2 DW TYPE_2
05E9 7706 1281 T_3 DW TYPE_3
1282 |
05EB 1283 STAT_ERR:
05EB 268A1E4200 1284 MOV BL,ES:HD_ERROR ; GET ERROR BYTE
05F0 8AC3 1285 MOV AL,BL
05F2 240F 1286 AND AL,0FH
05F4 80E330 1287 AND BL,00110000B ; ISOLATE TYPE
05F7 2AFF 1288 SUB BH,BH
05F9 B103 1289 MOV CL,3
05FB D3EB 1290 SHR BX,CL ; ADJUST
05FD 2EFA7E305 1291 JMP WORD PTR CS:[BX + OFFSET T_0]
1292 ASSUME ES:NOTHING
1293 |
0602 1294 TYPE0_TABLE LABEL BYTE

```

LOC OBJ	LINE	SOURCE
0602 00204020800020	1295	DB 0,BAD_CNTLRL,BAD_SEEK,BAD_CNTLRL,TIME_OUT,0,BAD_CNTLRL
0609 0040	1296	DB 0,BAD_SEEK
0009	1297	TYPE0_LEN EQU 4-TYPE0_TABLE
060B	1298	TYPE1_TABLE LABEL BYTE
060B 1010020004	1299	DB BAD_ECC,BAD_ECC,BAD_ADDR_MARK,0,RECORD_NOT_FND
0610 400000110B	1300	DB BAD_SEEK,0,0,DATA_CORRECTED,BAD_TRACK
000A	1301	TYPE1_LEN EQU 4-TYPE1_TABLE
0615	1302	TYPE2_TABLE LABEL BYTE
0615 0102	1303	DB BAD_CMD,BAD_ADDR_MARK
0002	1304	TYPE2_LEN EQU 4-TYPE2_TABLE
0617	1305	TYPE3_TABLE LABEL BYTE
0617 202010	1306	DB BAD_CNTLRL,BAD_CNTLRL,BAD_ECC
0003	1307	TYPE3_LEN EQU 4-TYPE3_TABLE
	1308	
	1309	1----- TYPE 0 ERROR
	1310	
061A	1311	TYPE_0:
061A BB0206	1312	MOV BX,OFFSET TYPE0_TABLE
061D 3C09	1313	CMPL AL,TYPE0_LEN ; CHECK IF ERROR IS DEFINED
061F 7363	1314	JAE UNDEF_ERR_L
0621 2ED7	1315	XLAT CS:TYPE0_TABLE ; TABLE LOOKUP
0623 A27400	1316	MOV DISK_STATUS,AL ; SET ERROR CODE
0626 C3	1317	RET
	1318	
	1319	1----- TYPE 1 ERROR
	1320	
0627	1321	TYPE_1:
0627 BB0B06	1322	MOV BX,OFFSET TYPE1_TABLE
062A 6BC8	1323	MOV CX,AX
062C 3C0A	1324	CMPL AL,TYPE1_LEN ; CHECK IF ERROR IS DEFINED
062E 7354	1325	JAE UNDEF_ERR_L
0630 2ED7	1326	XLAT CS:TYPE1_TABLE ; TABLE LOOKUP
0632 A27400	1327	MOV DISK_STATUS,AL ; SET ERROR CODE
0635 80E108	1328	AND CL,0BH ; CORRECTED ECC
0638 80F908	1329	CMPL CL,0BH
063B 752A	1330	JNZ G30
	1331	
	1332	1----- OBTAIN ECC ERROR BURST LENGTH
	1333	
063D C06642000D	1334	MOV CMD_BLOCK+0,RO_ECC_CMD
0642 2AC0	1335	SUB AL,AL
0644 E81BFF	1336	CALL COMMAND
0647 721E	1337	JC G30
0649 E83E00	1338	CALL HD_WAIT_REQ
064C 7219	1339	JC G30
064E E82001	1340	CALL PORT_0
0651 EC	1341	IN AL,DX
0652 8AC8	1342	MOV CL,AL
0654 E83300	1343	CALL HD_WAIT_REQ
0657 720E	1344	JC G30
0659 E81501	1345	CALL PORT_0
065C EC	1346	IN AL,DX
065D A801	1347	TEST AL,01H
065F 7406	1348	JZ G30
0661 C066740020	1349	MOV DISK_STATUS,BAD_CNTLRL
0666 F9	1350	STC
0667	1351	G30:
0667 8AC1	1352	MOV AL,CL
0669 C3	1353	RET
	1354	
	1355	1----- TYPE 2 ERROR
	1356	
066A	1357	TYPE_2:
066A BB1506	1358	MOV BX,OFFSET TYPE2_TABLE
066D 3C02	1359	CMPL AL,TYPE2_LEN ; CHECK IF ERROR IS DEFINED
066F 7313	1360	JAE UNDEF_ERR_L
0671 2ED7	1361	XLAT CS:TYPE1_TABLE ; TABLE LOOKUP
0673 A27400	1362	MOV DISK_STATUS,AL ; SET ERROR CODE
0676 C3	1363	RET
	1364	
	1365	1----- TYPE 3 ERROR
	1366	
0677	1367	TYPE_3:
0677 BB1706	1368	MOV BX,OFFSET TYPE3_TABLE
067A 3C03	1369	CMPL AL,TYPE3_LEN
067C 7306	1370	JAE UNDEF_ERR_L
067E 2ED7	1371	XLAT CS:TYPE3_TABLE

LOC OBJ	LINE	SOURCE	
0680 A27400	1372	MOV	DISK_STATUS,AL
0683 C3	1373	RET	
	1374		
0684	1375	UNDEF_ERR_L:	
0684 C6067400BB	1376	MOV	DISK_STATUS,UNDEF_ERR
0689 C3	1377	RET	
	1378		
068A	1379	HD_WAIT_REQ	PROC NEAR
068A 51	1380	PUSH	CX
068B 2BC9	1381	SUB	CX,CX
068D E8EE00	1382	CALL	PORT_1
0690	1383	L1:	
0690 EC	1384	IN	AL,DX
0691 A801	1385	TEST	AL,RI_REQ
0693 7508	1386	JNZ	L2
0695 E2F9	1387	LOOP	L1
0697 C6067400B0	1388	MOV	DISK_STATUS,TIME_OUT
069C F9	1389	STC	
069D	1390	L2:	
069D 59	1391	POP	CX
069E C3	1392	RET	
	1393	HD_WAIT_REQ	ENDP
	1394		
	1395	;-----	
	1396	; DMA_SETUP	:
	1397	; THIS ROUTINE SETS UP FOR DMA OPERATIONS.	:
	1398	; INPUT	:
	1399	; (AL) = MODE BYTE FOR THE DMA	:
	1400	; (ES:BX) = ADDRESS TO READ/WRITE THE DATA	:
	1401	; OUTPUT	:
	1402	; (AX) DESTROYED	:
	1403	;-----	
069F	1404	DMA_SETUP	PROC NEAR
069F 50	1405	PUSH	AX
06A0 A04600	1406	MOV	AL,CHD_BLOCK+4
06A3 3C81	1407	CMPL	AL,81H ; BLOCK COUNT OUT OF RANGE
06A5 58	1408	POP	AX
06A6 7202	1409	JB	J1
06A8 F9	1410	STC	
06A9 C3	1411	RET	
06AA	1412	J1:	
06AA 51	1413	PUSH	CX ; SAVE THE REGISTER
06AB FA	1414	CLI	; NO MORE INTERRUPTS
06AC E60C	1415	OUT	DMA+12,AL ; SET THE FIRST/LAST F/F
06AE 50	1416	PUSH	AX
06AF 58	1417	POP	AX
06B0 E60B	1418	OUT	DMA+11,AL ; OUTPUT THE MODE BYTE
06B2 8CC0	1419	MOV	AX,ES ; GET THE ES VALUE
06B4 B104	1420	MOV	CL,4 ; SHIFT COUNT
06B6 D3C0	1421	ROL	AX,CL ; ROTATE LEFT
06B8 8AE8	1422	MOV	CH,AL ; GET HIGHEST NYBBLE OF ES TO CH
06BA 24F0	1423	AND	AL,0F0H ; ZERO THE LOW NYBBLE FROM SEGMENT
06BC 03C3	1424	ADD	AX,BX ; TEST FOR CARRY FROM ADDITION
06BE 7302	1425	JNC	J33
06C0 FEC5	1426	INC	CH ; CARRY MEANS HIGH 4 BITS MUST BE INC
06C2	1427	J33:	
06C2 50	1428	PUSH	AX ; SAVE START ADDRESS
06C3 E606	1429	OUT	DMA+6,AL ; OUTPUT LOW ADDRESS
06C5 8AC4	1430	MOV	AL,AH
06C7 E606	1431	OUT	DMA+6,AL ; OUTPUT HIGH ADDRESS
06C9 8AC5	1432	MOV	AL,CH ; GET HIGH 4 BITS
06CB 240F	1433	AND	AL,0FH
06CD E682	1434	OUT	DMA_HIGH,AL ; OUTPUT THE HIGH 4 BITS TO PAGE REG
	1435		
	1436	;----- DETERMINE COUNT	
	1437		
06CF A04600	1438	MOV	AL,CHD_BLOCK+4 ; RECOVER BLOCK COUNT
06D2 D0E0	1439	SHL	AL,1 ; MULTIPLY BY 512 BYTES PER SECTOR
06D4 FEC8	1440	DEC	AL ; AND DECREMENT VALUE BY ONE
06D6 8AE0	1441	MOV	AH,AL
06D8 B0FF	1442	MOV	AL,OFFH
	1443		
	1444	;----- HANDLE READ AND WRITE LONG (5160 BYTE BLOCKS)	
	1445		
06DA 50	1446	PUSH	AX ; SAVE REGISTER
06DB A04200	1447	MOV	AL,CHD_BLOCK+0 ; GET COMMAND
06DE 3CE5	1448	CMPL	AL,RD_LONG_CHD

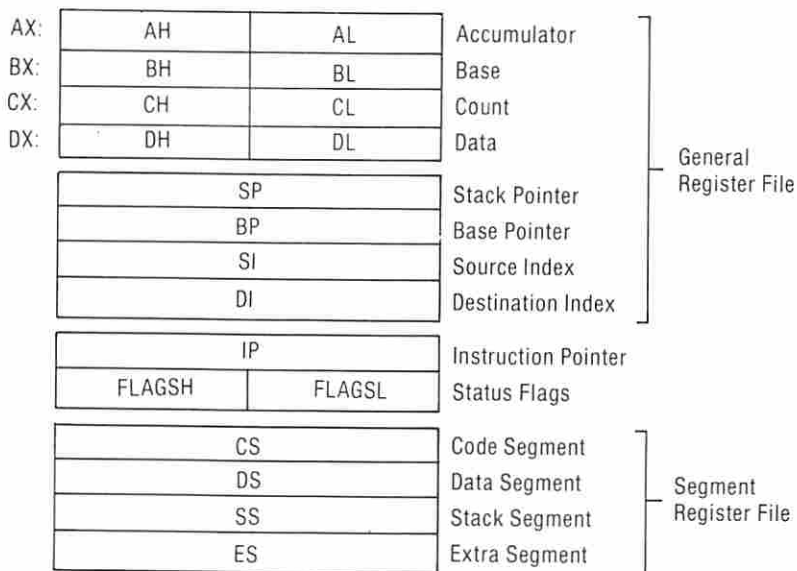
LOC OBJ	LINE	SOURCE
06E0 7407	1449	JE ADD4
06E2 3CE6	1450	CMF AL,MR_LONG_CMD
06E4 7403	1451	JE ADD4
06E6 58	1452	POP AX ; RESTORE REGISTER
06E7 EB11	1453	JMP SHORT J20
06E9	1454	ADD4:
06E9 58	1455	POP AX ; RESTORE REGISTER
06EA B80402	1456	MOV AX,5160 ; ONE BLOCK (512) PLUS 4 BYTES ECC
06ED 53	1457	PUSH BX
06EE 2AFF	1458	SUB BH,BH
06F0 8A1E4600	1459	MOV BL,CMD_BLOCK+4
06F4 52	1460	PUSH DX
06F5 F7E3	1461	MUL BX ; BLOCK COUNT TIMES 516
06F7 5A	1462	POP DX
06F8 5B	1463	POP BX
06F9 48	1464	DEC AX ; ADJUST
06FA	1465	J20:
	1466	
06FA 50	1467	PUSH AX ; SAVE COUNT VALUE
06FB E607	1468	OUT DMA+7,AL ; LOW BYTE OF COUNT
06FD 8AC4	1469	MOV AL,AH
06FF E607	1470	OUT DMA+7,AL ; HIGH BYTE OF COUNT
0701 FB	1471	STI ; INTERRUPTS BACK ON
0702 59	1472	POP CX ; RECOVER COUNT VALUE
0703 58	1473	POP AX ; RECOVER ADDRESS VALUE
0704 03C1	1474	ADD AX,CX ; ADD, TEST FOR 64K OVERFLOW
0706 59	1475	POP CX ; RECOVER REGISTER
0707 C3	1476	RET ; RETURN TO CALLER, CFL SET BY ABOVE IF ERROR
	1477	DMA_SETUP ENDP
	1478	
	1479	;-----
	1480	WAIT_INT ;
	1481	THIS ROUTINE WAITS FOR THE FIXED DISK ;
	1482	CONTROLLER TO SIGNAL THAT AN INTERRUPT ;
	1483	HAS OCCURRED. ;
	1484	;-----
0708	1485	WAIT_INT PROC NEAR
0708 FB	1486	STI ; TURN ON INTERRUPTS
0709 53	1487	PUSH BX ; PRESERVE REGISTERS
070A 51	1488	PUSH CX
070B 06	1489	PUSH ES
070C 56	1490	PUSH SI
070D 1E	1491	PUSH DS
	1492	ASSUME DS:DUMMY
070E 2BC0	1493	SUB AX,AX
0710 8ED8	1494	MOV DS,AX ; ESTABLISH SEGMENT
0712 C4360401	1495	LES SI,HF_TBL_VEC
	1496	ASSUME DS:DATA
0716 1F	1497	POP DS
	1498	
	1499	;----- SET TIMEOUT VALUES
	1500	
0717 2AFF	1501	SUB BH,BH
0719 268A5CD9	1502	MOV BL,BYTE PTR ES:[SI][0]
071D 8A264200	1503	MOV AL,020H ; STANDARD TIME OUT
0721 80FC04	1504	CMF AH,FMTDRV_CMD
0724 7506	1505	JNZ M5
0726 268A5CD0A	1506	MOV BL,BYTE PTR ES:[SI][0AH]
072A EB09	1507	JMP SHORT M4 ; FORMAT DRIVE
072C 80FCE3	1508	M5: CMF AH,CHK_DRV_CMD
072F 7504	1509	JNZ M4
0731 268A5C0B	1510	MOV BL,BYTE PTR ES:[SI][0BH]
0735	1511	M4: ; CHECK DRIVE
0735 2BC9	1512	SUB CX,CX
	1513	
	1514	;----- WAIT FOR INTERRUPT
	1515	
0737	1516	M1:
0737 E84400	1517	CALL PORT_1
073A EC	1518	IN AL,DX
073B 2420	1519	AND AL,020H
073D 3C20	1520	CMF AL,020H ; DID INTERRUPT OCCUR
073F 740A	1521	JZ M2
0741 E2F4	1522	LOOP M1 ; INNER LOOP
0743 4B	1523	DEC BX
0744 75F1	1524	JNZ M1 ; OUTER LOOP
0746 C606740080	1525	MOV DISK_STATUS,TIME_OUT
074B	1526	M2:

LOC	OBJ	LINE	SOURCE
074B	E02300	1527	CALL PORT_0
074E	EC	1528	IN AL,DX
074F	2402	1529	AND AL,2 ; ERROR BIT
0751	08067400	1530	OR DISK_STATUS,AL ; SAVE
0755	E63000	1531	CALL PORT_3 ; INTERRUPT MASK REGISTER
0758	32C0	1532	XOR AL,AL ; ZERO
075A	EE	1533	OUT DX,AL ; RESET MASK
075B	5E	1534	POP SI ; RESTORE REGISTERS
075C	07	1535	POP ES
075D	59	1536	POP CX
075E	5B	1537	POP BX
075F	C3	1538	RET
		1539	WAIT_INT ENDP
		1540	
0760		1541	HD_INT PROC NEAR
0760	50	1542	PUSH AX
0761	B020	1543	MOV AL,E0I ; END OF INTERRUPT
0763	E620	1544	OUT INT_CTL_PORT,AL
0765	B007	1545	MOV AL,07H ; SET DMA MODE TO DISABLE
0767	E60A	1546	OUT DMA+10,AL
0769	E421	1547	IN AL,021H
076B	0C20	1548	OR AL,020H
076D	E621	1549	OUT 021H,AL
076F	5B	1550	POP AX
0770	CF	1551	IRET
		1552	HD_INT ENDP
		1553	
		1554	;------
		1555	; PORTS :
		1556	; GENERATE PROPER PORT VALUE :
		1557	; BASED ON THE PORT OFFSET :
		1558	;------
		1559	
0771		1560	PORT_0 PROC NEAR
0771	BA2003	1561	MOV DX,HF_PORT ; BASE VALUE
0774	50	1562	PUSH AX
0775	2AE4	1563	SUB AH,AH
0777	A07700	1564	MOV AL,PORT_OFF ; ADD IN THE OFFSET
077A	03D0	1565	ADD DX,AX
077C	5B	1566	POP AX
077D	C3	1567	RET
		1568	PORT_0 ENDP
		1569	
077E		1570	PORT_1 PROC NEAR
077E	E8F0FF	1571	CALL PORT_0
0781	42	1572	INC DX ; INCREMENT TO PORT ONE
0782	C3	1573	RET
		1574	PORT_1 ENDP
		1575	
0783		1576	PORT_2 PROC NEAR
0783	E8F8FF	1577	CALL PORT_1
0786	42	1578	INC DX ; INCREMENT TO PORT TWO
0787	C3	1579	RET
		1580	PORT_2 ENDP
		1581	
0788		1582	PORT_3 PROC NEAR
0788	E8F8FF	1583	CALL PORT_2
078B	42	1584	INC DX ; INCREMENT TO PORT THREE
078C	C3	1585	RET
		1586	PORT_3 ENDP
		1587	
		1588	;------
		1589	; SH2_OFFS :
		1590	; DETERMINE PARAMETER TABLE OFFSET :
		1591	; USING CONTROLLER PORT TWO AND :
		1592	; DRIVE NUMBER SPECIFIER (0-1) :
		1593	;------
		1594	
078D		1595	SH2_OFFS PROC NEAR
078D	E8F3FF	1596	CALL PORT_2
0790	EC	1597	IN AL,DX ; READ PORT 2
0791	50	1598	PUSH AX
0792	E8E9FF	1599	CALL PORT_1
0795	EC	1600	IN AL,DX
0796	2402	1601	AND AL,2 ; CHECK FOR ERROR
0798	5B	1602	POP AX
0799	7516	1603	JNZ SH2_OFFS_ERR
079B	8A264300	1604	MOV AH,CHD_BLOCK+1

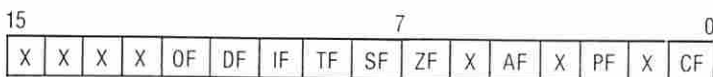
LOC OBJ	LINE	SOURCE	
079F 80E420	1605	AND AH,00100000B	; DRIVE 0 OR 1
07A2 7504	1606	JNZ SW2_AND	
07A4 D0E8	1607	SHR AL,1	; ADJUST
07A6 D0E8	1608	SHR AL,1	
07A8	1609	SW2_AND:	
07A8 2403	1610	AND AL,011B	; ISOLATE
07AA B104	1611	MOV CL,4	
07AC D2E0	1612	SHL AL,CL	; ADJUST
07AE 2AE4	1613	SUB AH,AH	
07B0 C3	1614	RET	
07B1	1615	SW2_OFFS_ERR:	
07B1 F9	1616	STC	
07B2 C3	1617	RET	
	1618	SW2_OFFS	ENDP
	1619		
07B3 30382F31362F36	1620	DB '08/16/82'	; RELEASE MARKER
32			
	1621		
07BB	1622	END_ADDRESS	LABEL BYTE
----	1623	CODE ENDS	
	1624	END	

APPENDIX B: 8088 ASSEMBLY INSTRUCTION SET REFERENCE

8088 Register Model



Instructions which reference the flag register file as a 16-bit object use the symbol **FLAGS** to represent the file:



x = Don't Care



Operand Summary

"reg" field Bit Assignments:

16-Bit (w=1)	8-Bit (w=0)	Segment
000 AX	000 AL	00 ES
001 CX	001 CL	01 CS
010 DX	010 DL	10 SS
011 BX	011 BL	11 DS
100 SP	100 AH	
101 BP	101 CH	
110 SI	110 DH	
111 DI	111 BH	

Second Instruction Byte Summary

mod	xxx	r/m
-----	-----	-----

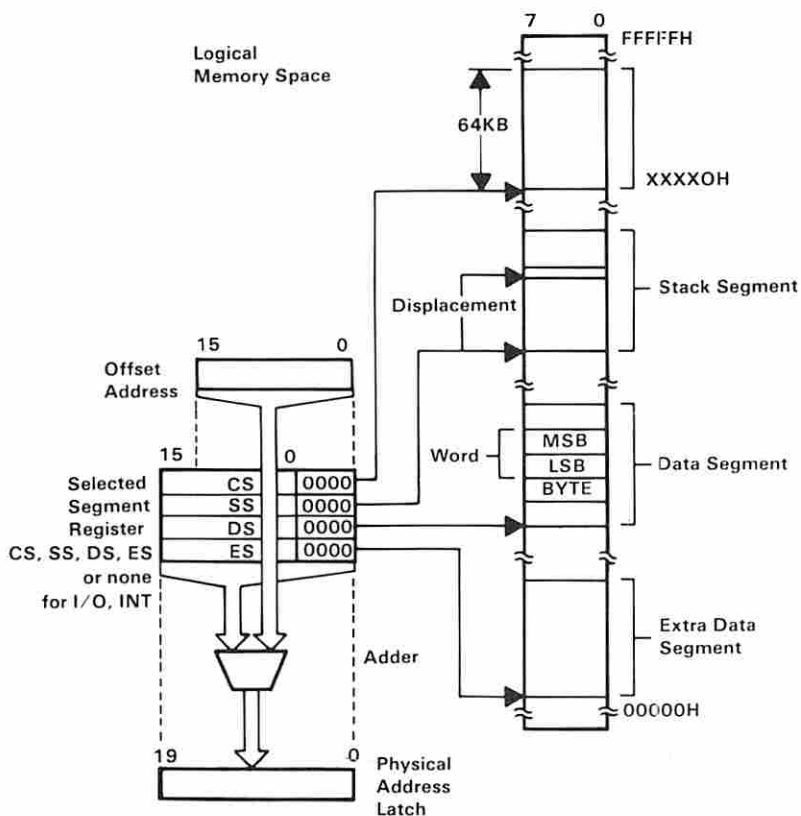
mod	Displacement
00	DISP=0*. disp-low and disp-high are absent
01	DISP=disp-low sign-extended to 16-bits. disp-high is absent
10	DISP=disp-high; disp-low
11	r/m is treated as a "reg" field

r/m	Operand Address
000	(BX) + (SI) + DISP
001	(BX) + (DI) + DISP
010	(BP) + (SI) + DISP
011	(BP) + (DI) + DISP
100	(SI) + DISP
101	(DI) + DISP
110	(BP) + DISP*
111	(BX) + DISP

DISP follows 2nd byte of instruction (before data if required).

*except if mod = 00 and r/m = 110 then EA = disp-high: disp-low.

Memory Segmentation Model



Segment Override Prefix

0 0 1 reg 1 1 0

Use of Segment Override

Operand Register	Default	With Override Prefix
IP (Code Address)	CS	Never
SP (Stack Address)	SS	Never
BP (Stack Address or Stack Marker)	SS	BP + DS or ES, or CS
SI or DI (not including strings)	DS	ES, SS, or CS
SI (Implicit Source Address for Strings)	DS	ES, SS, or CS
DI (Implicit Destination Address for Strings)	ES	Never

Data Transfer

MOV = Move

Register/memory to/from register

1	0	0	0	1	0	d	w	mod	reg	r/m
---	---	---	---	---	---	---	---	-----	-----	-----

Immediate to register/memory

1	1	0	0	0	1	1	w	mod	0	0	0	r/m	data	data if w=1
---	---	---	---	---	---	---	---	-----	---	---	---	-----	------	-------------

Immediate to register

1	0	1	1	w	reg	data	data if w=1
---	---	---	---	---	-----	------	-------------

Memory to accumulator

1	0	1	0	0	0	0	w	addr-low	addr-high
---	---	---	---	---	---	---	---	----------	-----------

Accumulator to memory

1	0	1	0	0	0	1	w	addr-low	addr-high
---	---	---	---	---	---	---	---	----------	-----------

Register/memory to segment register

1	0	0	0	1	1	1	0	mod	0	reg	r/m
---	---	---	---	---	---	---	---	-----	---	-----	-----

Segment register to register/memory

1	0	0	0	1	1	0	0	mod	0	reg	r/m
---	---	---	---	---	---	---	---	-----	---	-----	-----

PUSH = Push

Register/memory

1	1	1	1	1	1	1	1	mod	1	1	0	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

Register

0	1	0	1	0	reg
---	---	---	---	---	-----

Segment register

0	0	0	reg	1	1	0
---	---	---	-----	---	---	---

POP = Pop

Register/memory

1	0	0	0	1	1	1	1	mod	0	0	0	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

Register

0	1	0	1	1	reg
---	---	---	---	---	-----

Segment register

0	0	0	reg	1	1	1
---	---	---	-----	---	---	---

XCHG = Exchange

Register/memory with register

1	0	0	0	0	1	1	w	mod	reg	r/m
---	---	---	---	---	---	---	---	-----	-----	-----

Register with accumulator

1	0	0	1	0	reg
---	---	---	---	---	-----

IN = Input to AL/AX from

Fixed port

1	1	1	0	0	1	0	w	port
---	---	---	---	---	---	---	---	------

Variable port (DX)

1	1	1	0	1	1	0	w
---	---	---	---	---	---	---	---

OUT = Output from AL/AX to

Fixed port

1	1	1	0	0	1	1	w	port
---	---	---	---	---	---	---	---	------

Variable port (DX)

1	1	1	0	1	1	0	w
---	---	---	---	---	---	---	---

XLAT = Translate byte to AL

1	1	0	1	0	1	1	1
---	---	---	---	---	---	---	---

LEA = Load EA to register

1	0	0	0	1	1	0	1	mod	reg	r/m
---	---	---	---	---	---	---	---	-----	-----	-----

LDS = Load pointer to DS

1	1	0	0	0	1	0	1	mod	reg	r/m
---	---	---	---	---	---	---	---	-----	-----	-----

LES = Load pointer to ES

1	1	0	0	0	1	0	0	mod	reg	r/m
---	---	---	---	---	---	---	---	-----	-----	-----

LAHF = Load AH with flags

1	0	0	1	1	1	1	1
---	---	---	---	---	---	---	---

SAHF = Store AH into flags

1	0	0	1	1	1	1	0
---	---	---	---	---	---	---	---

PUSHF = Push flags

1	0	0	1	1	1	0	0
---	---	---	---	---	---	---	---

POPF = Pop flags

1	0	0	1	1	1	0	1
---	---	---	---	---	---	---	---

Arithmetic

ADD = Add

Register/memory with register to either

0	0	0	0	0	0	d	w	mod	reg	r/m
---	---	---	---	---	---	---	---	-----	-----	-----

Immediate to register/memory

1	0	0	0	0	0	s	w	mod	0	0	0	r/m	data	data if s:w=01
---	---	---	---	---	---	---	---	-----	---	---	---	-----	------	----------------

Immediate to accumulator

0	0	0	0	0	1	0	w	data	data if w=1
---	---	---	---	---	---	---	---	------	-------------

ADC = Add with carry

Register/memory with register to either

0	0	0	1	0	0	d	w	mod	reg	r/m
---	---	---	---	---	---	---	---	-----	-----	-----

Immediate to register/memory

1	0	0	0	0	0	s	w	mod	0	1	0	r/m	data	data if s:w=01
---	---	---	---	---	---	---	---	-----	---	---	---	-----	------	----------------

Immediate to accumulator

0	0	0	1	0	1	0	w	data	data if w=1
---	---	---	---	---	---	---	---	------	-------------

INC = Increment

Register/memory

1	1	1	1	1	1	1	w	mod	0	0	0	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

Register

0	1	0	0	0	reg
---	---	---	---	---	-----

AAA = ASCII adjust for add

0	0	1	1	0	1	1	1
---	---	---	---	---	---	---	---

DAA = Decimal adjust for add

0	0	1	0	0	1	1	1
---	---	---	---	---	---	---	---

SUB = Subtract

Register/memory and register to either

0	0	1	0	1	0	d	w	mod	reg	r/m
---	---	---	---	---	---	---	---	-----	-----	-----

Immediate from register/memory

1	0	0	0	0	0	s	w	mod	1	0	1	r/m	data	data if s:w=01
---	---	---	---	---	---	---	---	-----	---	---	---	-----	------	----------------

Immediate from accumulator

0	0	1	0	1	1	0	w	data	data if w=1
---	---	---	---	---	---	---	---	------	-------------

SBB = Subtract with borrow

Register/memory and register to either

0	0	0	1	1	0	d	w	mod	reg	r/m
---	---	---	---	---	---	---	---	-----	-----	-----

Immediate from register/memory

1	0	0	0	0	0	s	w	mod	0	1	1	r/m	data	data if s:w=01
---	---	---	---	---	---	---	---	-----	---	---	---	-----	------	----------------

Immediate from accumulator

0	0	0	1	1	1	0	w	data	data if w=1
---	---	---	---	---	---	---	---	------	-------------

DEC = Decrement

Register/memory

1	1	1	1	1	1	1	w	mod	0	0	1	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

Register

0	1	0	0	1	reg
---	---	---	---	---	-----

NEG = Change sign

1	1	1	1	0	1	1	w	mod	0	1	1	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

CMP = Compare

Register/memory and register

0	0	1	1	1	0	d	w	mod	reg	r/m
---	---	---	---	---	---	---	---	-----	-----	-----

Immediate with register/memory

1	0	0	0	0	0	s	w	mod	1	1	1	r/m	data	data if s:w=01
---	---	---	---	---	---	---	---	-----	---	---	---	-----	------	----------------

Immediate with accumulator

0	0	1	1	1	1	0	w	data	data if w=1
---	---	---	---	---	---	---	---	------	-------------

AAS = ASCII adjust for subtract

0	0	1	1	1	1	1	1
---	---	---	---	---	---	---	---

DAS = Decimal adjust for subtract

0	0	1	0	1	1	1	1
---	---	---	---	---	---	---	---

MUL = Multiply (unsigned)

1	1	1	1	0	1	1	w	mod	1	0	0	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

IMUL = Integer multiply (signed)

1	1	1	1	0	1	1	w	mod	1	0	1	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

AAM = ASCII adjust for multiply

1	1	0	1	0	1	0	0	0	0	0	1	0	1	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

DIV = Divide (unsigned)

1	1	1	1	0	1	1	w	mod	1	1	0	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

IDIV = Integer divide (signed)

1	1	1	1	0	1	1	w	mod	1	1	1	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

AAD = ASCII adjust for divide

1	1	0	1	0	1	0	1	0	0	0	1	0	1	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

CBW = Convert byte to word

1	0	0	1	1	0	0	0
---	---	---	---	---	---	---	---

CWD = Convert word to double word

1	0	0	1	1	0	0	1
---	---	---	---	---	---	---	---

Logic

NOT = Invert

1	1	1	1	0	1	1	w	mod	0	1	0	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

SHL/SAL = Shift logical/arithmetic left

1	1	0	1	0	0	v	w	mod	1	0	0	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

SHR = Shift logical right

1	1	0	1	0	0	v	w	mod	1	0	1	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

SAR = Shift arithmetic right

1	1	0	1	0	0	v	w	mod	1	1	1	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

ROL = Rotate left

1	1	0	1	0	0	v	w	mod	0	0	0	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

ROR = Rotate right

1	1	0	1	0	0	v	w	mod	0	0	1	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

RCL = Rotate through carry left

1	1	0	1	0	0	v	w	mod	0	1	0	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

RCR = Rotate through carry right

1	1	0	1	0	0	v	w	mod	0	1	1	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

AND = And

Register/memory and register to either

0	0	1	0	0	0	d	w	mod	reg	r/m
---	---	---	---	---	---	---	---	-----	-----	-----

Immediate to register/memory

1	0	0	0	0	0	0	w	mod	1	0	0	r/m	data	data if w=1
---	---	---	---	---	---	---	---	-----	---	---	---	-----	------	-------------

Immediate to accumulator

0	0	1	0	0	1	0	w	data	data if w=1
---	---	---	---	---	---	---	---	------	-------------

TEST = And function to flags, no result
Register/memory and register

1	0	0	0	0	1	0	w	mod	reg	r/m
---	---	---	---	---	---	---	---	-----	-----	-----

Immediate data and register/memory

1	1	1	1	0	1	1	w	mod	0	0	0	r/m	data	data if w=1
---	---	---	---	---	---	---	---	-----	---	---	---	-----	------	-------------

Immediate data and accumulator

1	0	1	0	1	0	0	w	data	data if w=1
---	---	---	---	---	---	---	---	------	-------------

OR = OR

Register/memory and register to either

0	0	0	0	1	0	d	w	mod	reg	r/m
---	---	---	---	---	---	---	---	-----	-----	-----

Immediate to register/memory

1	0	0	0	0	0	0	w	mod	0	0	1	r/m	data	data if w=1
---	---	---	---	---	---	---	---	-----	---	---	---	-----	------	-------------

Immediate to accumulator

0	0	0	0	1	1	0	w	data	data if w=1
---	---	---	---	---	---	---	---	------	-------------

XOR = Exclusive or

Register/memory and register to either

0	0	1	1	0	0	d	w	mod	reg	r/m
---	---	---	---	---	---	---	---	-----	-----	-----

Immediate to register/memory

1	0	0	0	0	0	0	w	mod	1	1	0	r/m	data	data if w=1
---	---	---	---	---	---	---	---	-----	---	---	---	-----	------	-------------

Immediate to accumulator

0	0	1	1	0	1	0	w	data	data if w=1
---	---	---	---	---	---	---	---	------	-------------

String Manipulation

REP = Repeat

1	1	1	1	0	0	1	z
---	---	---	---	---	---	---	---

MOVS = Move String

1	0	1	0	0	1	0	w
---	---	---	---	---	---	---	---

CMPS = Compare String

1	0	1	0	0	1	1	w
---	---	---	---	---	---	---	---

SCAS = Scan String

1	0	1	0	1	1	1	w
---	---	---	---	---	---	---	---

LODS = Load String

1	0	1	0	1	1	0	w
---	---	---	---	---	---	---	---

STOS = Store String

1	0	1	0	1	0	1	w
---	---	---	---	---	---	---	---

Control Transfer

CALL = Call

Direct within segment

1	1	1	0	1	0	0	0	disp-low	disp-high
---	---	---	---	---	---	---	---	----------	-----------

Indirect within segment

1	1	1	1	1	1	1	1	mod	0	1	0	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

Direct intersegment

1	0	0	1	1	0	1	0	offset-low	offset-high
---	---	---	---	---	---	---	---	------------	-------------

seg-low	seg-high
---------	----------

Indirect intersegment

1	1	1	1	1	1	1	1	mod	0	1	1	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

JMP = Unconditional Jump

Direct within segment

1	1	1	0	1	0	0	1	disp-low	disp-high
---	---	---	---	---	---	---	---	----------	-----------

Direct within segment-short

1	1	1	0	1	0	1	1	disp
---	---	---	---	---	---	---	---	------

Indirect within segment

1	1	1	1	1	1	1	1	mod	1	0	0	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

Direct intersegment

1	1	1	0	1	0	1	0	offset-low	offset-high
---	---	---	---	---	---	---	---	------------	-------------

seg-low	seg-high
---------	----------

Indirect intersegment

1	1	1	1	1	1	1	1	mod	1	0	1	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

RET = Return from CALL

Within segment

1	1	0	0	0	0	1	1
---	---	---	---	---	---	---	---

Within segment adding immediate to SP

1	1	0	0	0	0	1	0
							data-low
							data-high

Intersegment

1	1	0	0	1	0	1	1
---	---	---	---	---	---	---	---

Intersegment, adding immediate to SP

1	1	0	0	0	0	1	0
							data-low
							data-high

JE/JZ = Jump on equal/zero

0	1	1	1	0	1	0	0
							disp

JL/JNGE = Jump on less/not greater or equal

0	1	1	1	1	1	0	0
							disp

JLE/JNG = Jump on less or equal/not greater

0	1	1	1	1	1	1	0
							disp

JB/JNAE = Jump on below/not above or equal

0	1	1	1	0	0	1	0
							disp

JBE/JNA = Jump on below or equal/not above

0	1	1	1	0	1	1	0
							disp

JP/JPE = Jump on parity/parity even

0	1	1	1	1	0	1	0
							disp

JO = Jump on overflow

0	1	1	1	0	0	0	0
							disp

JS = Jump on sign

0	1	1	1	1	0	0	0
							disp

JNE/JNZ = Jump on not equal/not zero

0	1	1	1	0	1	0	1
							disp

JNL/JGE = Jump on not less/greater or equal

0	1	1	1	1	1	0	1
							disp

JNLE/JG = Jump on not less or equal/greater

0 1 1 1 1 1 1 1	disp
-----------------	------

JNB/JAE = Jump on not below/above or equal

0 1 1 1 0 0 1 1	disp
-----------------	------

JNBE/JA = Jump on not below or equal/above

0 1 1 1 0 1 1 1	disp
-----------------	------

JNP/JPO = Jump on not parity/parity odd

0 1 1 1 1 0 1 1	disp
-----------------	------

JNO = Jump on not overflow

0 1 1 1 0 0 0 1	disp
-----------------	------

JNS = Jump on not sign

0 1 1 1 1 0 0 1	disp
-----------------	------

LOOP = Loop CX times

1 1 1 0 0 0 1 0	disp
-----------------	------

LOOPZ/LOOPE = Loop while zero/equal

1 1 1 0 0 0 0 1	disp
-----------------	------

LOOPNZ/LOOPNE = Loop while not zero/not equal

1 1 1 0 0 0 0 0	disp
-----------------	------

JCXZ = Jump on CX zero

1 1 1 0 0 0 1 1	disp
-----------------	------

8088 Conditional Transfer Operations

Instruction	Condition	Interpretation
JE or JZ	ZF = 1	"equal" or "zero"
JL or JNGE	(SF xor OF) = 1	"less" or "not greater or equal"
JLE or JNG	((SF xor OF) or ZF) = 1	"less or equal" or "not greater"
JB or JNAE or JC	CF = 1	"below" or "not above or equal"
JBE or JNA	(CF or ZF) = 1	"below or equal" or "not above"
JP or JPE	PF = 1	"parity" or "parity even"
JO	OF = 1	"overflow"
JS	SF = 1	"sign"
JNE or JNZ	ZF = 0	"not equal" or "not zero"
JNL or JGE	(SF xor OF) = 0	"not less" or "greater or equal"
JNLE or JG	((SF xor OF) or ZF) = 0	"not less or equal" or "greater"
JNB or JAE or JNC	CF = 0	"not below" or "above or equal"
JNBE or JA	(CF or ZF) = 0	"not below or equal" or "above"
JNP or JPO	PF = 0	"not parity" or "parity odd"
JNO	OF = 0	"not overflow"
JNS	SF = 0	"not sign"

*"Above" and "below" refer to the relation between two unsigned values, while "greater" and "less" refer to the relation between two signed values.

INT = Interrupt
Type specified

1 1 0 0 1 1 0 1	type
-----------------	------

Type 3

1 1 0 0 1 1 0 1

INT0 = Interrupt on overflow

1 1 0 0 1 1 1 0

IRET = Interrupt return

1 1 0 0 1 1 1 1

Processor Control

CLC = Clear carry

1	1	1	1	1	0	0	0
---	---	---	---	---	---	---	---

STC = Set carry

1	1	1	1	1	0	0	1
---	---	---	---	---	---	---	---

CMC = Complement carry

1	1	1	1	0	1	0	1
---	---	---	---	---	---	---	---

NOP = No operation

1	0	0	1	0	0	0	0
---	---	---	---	---	---	---	---

CLD = Clear direction

1	1	1	1	1	1	0	0
---	---	---	---	---	---	---	---

STD = Set direction

1	1	1	1	1	1	0	1
---	---	---	---	---	---	---	---

CLI = Clear interrupt

1	1	1	1	1	0	1	0
---	---	---	---	---	---	---	---

STI = Set interrupt

1	1	1	1	1	0	1	1
---	---	---	---	---	---	---	---

HLT = Halt

1	1	1	1	0	1	0	0
---	---	---	---	---	---	---	---

WAIT = Wait

1	0	0	1	1	0	1	1
---	---	---	---	---	---	---	---

LOCK = Bus lock prefix

1	1	1	1	0	0	0	0
---	---	---	---	---	---	---	---

ESC = Escape (to external device)

1	1	0	1	1	x	x	x	mod	x	x	x	r/m
---	---	---	---	---	---	---	---	-----	---	---	---	-----

Footnotes:

if d = 1 then "to"; if d = 0 then "from"

if w = 1 then word instruction; if w = 0 then byte instruction

if s:w = 01 then 16 bits of immediate data from the operand

if s:w = 11 then an immediate data byte is sign extended to form the 16-bit operand

if v = 0 then "count" = 1; if v = 1 then "count" in (CL)

x = don't care

z is used for some string primitives to compare with ZF FLAG

AL = 8-bit accumulator

AX = 16-bit accumulator

CX = Count register

DS = Data segment

DX = Variable port register

ES = Extra segment

Above/below refers to unsigned value

Greater = more positive;

Less = less positive (more negative) signed values

8088 Instruction Set Matrix

HI \ LO								
	0	1	2	3	4	5	6	7
0	ADD b,f,r/m	ADD w,f,r/m	ADD b,t,r/m	ADD w,t,r/m	ADD b,ia	ADD w,ia	PUSH ES	POP ES
1	ADC b,f,r/m	ADC w,f,r/m	ADC b,t,r/m	ADC w,t,r/m	ADC b,i	ADC w,i	PUSH SS	POP SS
2	AND b,f,r/m	AND w,f,r/m	AND b,t,r/m	AND w,t,r/m	AND b,i	AND w,i	SEG =ES	DAA
3	XOR b,f,r/m	XOR w,f,r/m	XOR b,t,r/m	XOR w,t,r/m	XOR b,i	XOR w,i	SEG =SS	AAA
4	INC AX	INC CX	INC DX	INC BX	INC SP	INC BP	INC SI	INC DI
5	PUSH AX	PUSH CX	PUSH DX	PUSH BX	PUSH SP	PUSH BP	PUSH SI	PUSH DI
6								
7	J0	JNO	JB/ JNAE	JNB/ JAE	JE/ JZ	JNE/ JNZ	JBE/ JNA	JNBE/ JA
8	Immed b,r/m	Immed w,r/m	Immed b,r/m	Immed is,r/m	TEST b,r/m	TEST w,r/m	XCHG b,r/m	XCHG w,r/m
9	NOP	XCHG CX	XCHG DX	XCHG BX	XCHG SP	XCHG BP	XCHG SI	XCHG DI
A	MOV m AL	MOV m AL	MOV AL m	MOV AL m	MOVS b	MOVS w	CMPS b	CMPS w
B	MOV i AL	MOV i CL	MOV i DL	MOV i BL	MOV i AH	MOV i CH	MOV i DH	MOV i BH
C			RET (i+SP)	RET	LES	LDS	MCV b,i,r/m	MOV w,i,r/m
D	Shift b	Shift w	Shift b,v	Shift w,v	AAM	AAD		XLAT
E	LOOPNZ/ LOOPNE	LOOPZ/ LOOPE	LOOP	JCXZ	IN b	IN w	OUT b	OUT w
F	LOCK		REP	REP z	HLT	CMC	Grp 1 b,r/m	Grp 1 w,r/m

b = byte operation

d = direct

f = from CPU reg

i = immediate

ia = immed. to accum.

id = indirect

is = immed. byte, sign ext.

l = long ie. intersegment

m = memory

r/m = EA is second byte

si = short intrasegment

sr = segment register

t = to CPU reg

v = variable

w = word operation

z = zero

8088 Instruction Set Matrix

LO HI	8	9	A	B	C	D	E	F
0	OR b,f,r/m	w,f,r/m	OR b,t,r/m	OR w,t,r/m	OR b,i	OR w,i	PUSH CS	
1	SBB b,f,r/m	SBB w,f,r/m	SBB b,t,r/m	SBB w,t,r/m	SBB b,i	SBB w,i	PUSH DS	POP DS
2	SUB b,f,r/m	SUB w,f,r/m	SUB b,t,r/m	SUB w,t,r/m	SUB b,i	SUB w,i	SEG= CS	DAS
3	CMP b,f,r/m	CMP w,f,r/m	CMP b,t,r/m	CMP w,t,r/m	CMP b,i	CMP w,i	SEG= CS	AAS
4	DEC AX	DEC CX	DEC DX	DEC BX	DEC SP	DEC BP	DEC SI	DEC DI
5	POP AX	POP CX	POP DX	POP BX	POP SP	POP BP	POP SI	POP DI
6								
7	JS	JNS	JP/ JPE	JNP/ JPO	JL/ JNGE	JNL/ JGE	JLE/ JNG	JNLE/ JG
8	MOV b,f,r/m	MOV w,f,r/m	MOV b,t,r/m	MOV w,t,r/m	MOV sr,t,r/m	LEA	MOV sr,f,r/m	POP r/m
9	CBW	CWD	CALL l,d	WAIT	PUSHF	POPF	SAHF	LAHF
A	TEST b,i	TEST w,i	STOS b	STOS w	LDS b	LDS w	SCAS b	SCAS w
B	MOV i AX	MOV i CX	MOV i DX	MOV i BX	MOV i SP	MOV i BP	MOV i SI	MOV i DI
C			RET l,(i+SP)	RET l	INT Type 3	INT (Any)	INTO	IRET
D	ESC 0	ESC 1	ESC 2	ESC 3	ESC 4	ESC 5	ESC 6	ESC 7
E	CALL d	JMP d	JMP l,d	JMP si,d	IN v,b	IN v,w	OUT v,b	OUT v,w
F	CLC	STC	CLI	STI	CLD	STD	Grp 2 b,r/m	Grp 2 w,r/m

where:

mod $\square$ r/m	000	001	010	011	100	101	110	111
Immed	ADD	OR	ADC	SBB	AND	SUB	XOR	CMP
Shift	ROL	ROR	RCL	RCR	SHL/SAL	SHR	—	SAR
Grp 1	TEST	—	NOT	NEG	MUL	IMUL	DIV	IDIV
Grp 2	INC	DEC	CALL id	CALL l,id	JMP id	JMP l,id	PUSH	—

Instruction Set Index

Mnemonic	Page	Mnemonic	Page	Mnemonic	Page
AAA	B-7	JG	B-13	MOV	B-5
AAD	B-9	JGE	B-12	MOVS	B-10
AAM	B-8	JL	B-12	MUL	B-8
AAS	B-8	JLE	B-12	NEG	B-8
ADC	B-7	JMP	B-11	NOP	B-15
ADD	B-7	JNA	B-12	NOT	B-9
AND	B-9	JNAE	B-12	OR	B-10
CALL	B-11	JNB	B-13	OUT	B-6
CBW	B-9	JNBE	B-13	POP	B-5
CLC	B-15	JNE	B-12	POPF	B-6
CLD	B-15	JNG	B-12	PUSH	B-5
CLI	B-15	JNGE	B-12	PUSHF	B-6
CMC	B-15	JNL	B-12	RCL	B-9
CMP	B-8	JNLE	B-13	RCR	B-9
CMPS	B-10	JNO	B-13	REP	B-10
CWD	B-9	JNP	B-13	RET	B-12
DAA	B-7	JNS	B-13	ROL	B-9
DAS	B-8	JNZ	B-12	ROR	B-9
DEC	B-8	JO	B-12	SAHF	B-6
DIV	B-8	JP	B-12	SAL	B-9
ESC	B-15	JPE	B-12	SAR	B-9
HLT	B-15	JPO	B-13	SBB	B-8
IDIV	B-9	JS	B-12	SCAS	B-10
IMUL	B-8	JZ	B-12	SHL	B-9
IN	B-6	LAHF	B-6	SHR	B-9
INC	B-7	LDS	B-6	STC	B-15
INT	B-14	LEA	B-6	STD	B-15
INTO	B-14	LES	B-6	STI	B-15
IRET	B-14	LOCK	B-15	STOS	B-11
JA	B-13	LODS	B-11	SUB	B-7
JAE	B-13	LOOP	B-13	TEST	B-10
JB	B-12	LOOPE	B-13	WAIT	B-15
JBE	B-12	LOOPNE	B-13	XCHG	B-6
JCXZ	B-13	LOOPNZ	B-13	XLAT	B-6
JE	B-12	LOOPZ	B-13	XOR	B-10

APPENDIX C: OF CHARACTERS, KEYSTROKES, AND COLORS

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
00	0	Blank (Null)	Ctrl 2		Black	Black	Non-Display
01	1	☺	Ctrl A		Black	Blue	Underline
02	2	☹	Ctrl B		Black	Green	Normal
03	3	♥	Ctrl C		Black	Cyan	Normal
04	4	♦	Ctrl D		Black	Red	Normal
05	5	♣	Ctrl E		Black	Magenta	Normal
06	6	♠	Ctrl F		Black	Brown	Normal
07	7	•	Ctrl G		Black	Light Grey	Normal
08	8	•	Ctrl H, Backspace, Shift Backspace		Black	Dark Grey	Non-Display
09	9	○	Ctrl I		Black	Light Blue	High Intensity Underline
0A	10	◉	Ctrl J, ⏏		Black	Light Green	High Intensity
0B	11	♂	Ctrl K		Black	Light Green	High Intensity
0C	12	♀	Ctrl L		Black	Light Red	High Intensity
0D	13	♪	Ctrl M, ⏏, Shift ⏏		Black	Light Magenta	High Intensity
0E	14	🎵	Ctrl N		Black	Yellow	High Intensity
0F	15	☀	Ctrl O		Black	White	High Intensity
10	16	▶	Ctrl P		Blue	Black	Normal
11	17	◀	Ctrl Q		Blue	Blue	Underline
12	18	↕	Ctrl R		Blue	Green	Normal
13	19	!!	Ctrl S		Blue	Cyan	Normal
14	20	⏏	Ctrl T		Blue	Red	Normal
15	21	§	Ctrl U			Magenta	Normal
16	22	■	Ctrl V		Blue	Brown	Normal
17	23	⏏	Ctrl W		Blue	Light Grey	Normal

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
18	24	!	Ctrl X		Blue	Dark Grey	High Intensity
19	25		Ctrl Y		Blue	Light Blue	High Intensity Underline
1A	26	—	Ctrl Z		Blue	Light Green	High Intensity
1B	27	— Esc, Shift Esc, Ctrl Esc	Ctrl [, Esc, Shift Esc, Ctrl Esc		Blue	Light Cyan	High Intensity
1C	28	└	Ctrl \		Blue	Light Red	High Intensity
1D	29	↔	Ctrl]		Blue	Light Magenta	High Intensity
1E	30	▲	Ctrl 6		Blue	Yellow	High Intensity
1F	31	▼	Ctrl —		Blue	White	High Intensity
20	32	Blank Space	Space Bar, Shift, Space, Ctrl Space, Alt Space		Green	Black	Normal
21	33	!	!	Shift	Green	Blue	Underline
22	34	“	“	Shift	Green	Green	Normal
23	35	#	#	Shift	Green	Cyan	Normal
24	36	\$	\$	Shift	Green	Red	Normal
25	37	%	%	Shift	Green	Magenta	Normal
26	38	&	&	Shift	Green	Brown	Normal
27	39	·	·		Green	Light Grey	Normal
28	40	(	(	Shift	Green	Dark Grey	High Intensity
29	41	)	)	Shift	Green	Light Blue	High Intensity Underline
2A	42	*	*	Note 1	Green	Light Green	High Intensity
2B	43	+	+	Shift	Green	Light Cyan	High Intensity
2C	44	·	·		Green	Light Red	High Intensity
2D	45	—	—		Green	Light Magenta	High Intensity
2E	46	.	.	Note 2	Green	Yellow	High Intensity

C-2 Of Characters, Keystrokes, and Colors

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
2F	47	/	/		Green	White	High Intensity
30	48	0	0	Note 3	Cyan	Black	Normal
31	49	1	1	Note 3	Cyan	Blue	Underline
32	50	2	2	Note 3	Cyan	Green	Normal
33	51	3	3	Note 3	Cyan	Cyan	Normal
34	52	4	4	Note 3	Cyan	Red	Normal
35	53	5	5	Note 3	Cyan	Magenta	Normal
36	54	6	6	Note 3	Cyan	Brown	Normal
37	55	7	7	Note 3	Cyan	Light Grey	Normal
38	56	8	8	Note 3	Cyan	Dark Grey	High Intensity
39	57	9	9	Note 3	Cyan	Light Blue	High Intensity Underline
3A	58	:	:	Shift	Cyan	Light Green	High Intensity
3B	59	;	;		Cyan	Light Cyan	High Intensity
3C	60	<	<	Shift	Cyan	Light Red	High Intensity
3D	61	=	=		Cyan	Light Magenta	High Intensity
3E	62	>	>	Shift	Cyan	Yellow	High Intensity
3F	63	?	?	Shift	Cyan	White	High Intensity
40	64	@	@	Shift	Red	Black	Normal
41	65	A	A	Note 4	Red	Blue	Underline
42	66	B	B	Note 4	Red	Green	Normal
43	67	C	C	Note 4	Red	Cyan	Normal
44	68	D	D	Note 4	Red	Red	Normal
45	69	E	E	Note 4	Red	Magenta	Normal
46	70	F	F	Note 4	Red	Brown	Normal
47	71	G	G	Note 4	Red	Light Grey	Normal
48	72	H	H	Note 4	Red	Dark Grey	High Intensity
49	73	I	I	Note 4	Red	Light Blue	High Intensity Underline
4A	74	J	J	Note 4	Red	Light Green	High Intensity

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
4B	75	K	K	Note 4	Red	Light Cyan	High Intensity
4C	76	L	L	Note 4	Red	Light Red	High Intensity
4D	77	M	M	Note 4	Red	Light Magenta	High Intensity
4E	78	N	N	Note 4	Red	Yellow	High Intensity
4F	79	O	O	Note 4	Red	White	High Intensity
50	80	P	P	Note 4	Magenta	Black	Normal
51	81	Q	Q	Note 4	Magenta	Blue	Underline
52	82	R	R	Note 4	Magenta	Green	Normal
53	83	S	S	Note 4	Magenta	Cyan	Normal
54	84	T	T	Note 4	Magenta	Red	Normal
55	85	U	U	Note 4	Magenta	Magenta	Normal
56	86	V	V	Note 4	Magenta	Brown	Normal
57	87	W	W	Note 4	Magenta	Light Grey	Normal
58	88	X	X	Note 4	Magenta	Dark Grey	High Intensity
59	89	Y	Y	Note 4	Magenta	Light Blue	High Intensity Underline
5A	90	Z	Z	Note 4	Magenta	Light Green	High Intensity
5B	91	[	[		Magenta	Light Cyan	High Intensity
5C	92	\	\		Magenta	Light Red	High Intensity
5D	93	]	]		Magenta	Light Magenta	High Intensity
5E	94	^	^	Shift	Magenta	Yellow	High Intensity
5F	95	_	_	Shift	Magenta	White	High Intensity
60	96	.	.		Yellow	Black	Normal
61	97	a	a	Note 5	Yellow	Blue	Underline
62	98	b	b	Note 5	Yellow	Green	Normal
63	99	c	c	Note 5	Yellow	Cyan	Normal
64	100	d	d	Note 5	Yellow	Red	Normal
65	101	e	e	Note 5	Yellow	Magenta	Normal
66	102	f	f	Note 5	Yellow	Brown	Normal

C-4 Of Characters, Keystrokes, and Colors

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
67	103	g	g	Note 5	Yellow	Light Grey	Normal
68	104	h	h	Note 5	Yellow	Dark Grey	High Intensity
69	105	i	i	Note 5	Yellow	Light Blue	High Intensity Underline
6A	106	j	j	Note 5	Yellow	Light Green	High Intensity
6B	107	k	k	Note 5	Yellow	Light Cyan	High Intensity
6C	108	l	l	Note 5	Yellow	Light Red	High Intensity
6D	109	m	m	Note 5	Yellow	Light Magenta	High Intensity
6E	110	n	n	Note 5	Yellow	Yellow	High Intensity
6F	111	o	o	Note 5	Yellow	White	High Intensity
70	112	p	p	Note 5	White	Black	Reverse Video
71	113	q	q	Note 5	White	Blue	Underline
72	114	r	r	Note 5	White	Green	Normal
73	115	s	s	Note 5	White	Cyan	Normal
74	116	f	f	Note 5	White	Red	Normal
75	117	u	u	Note 5	White	Magenta	Normal
76	118	v	v	Note 5	White	Brown	Normal
77	119	w	w	Note 5	White	Light Grey	Normal
78	120	x	x	Note 5	White	Dark Grey	Reverse Video
79	121	y	y	Note 5	White	Light Blue	High Intensity Underline
7A	122	z	z	Note 5	White	Light Green	High Intensity
7B	123			Shift	White	Light Cyan	High Intensity
7C	124			Shift	White	Light Red	High Intensity
7D	125			Shift	White	Light Magenta	High Intensity
7E	126	~	~	Shift	White	Yellow	High Intensity
7F	127	Δ	Ctrl ~		White	White	High Intensity

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
* * * * 80 to FF Hex are Flashing in both Color & IBM Monochrome * * * *							
80	128	Ç	Alt 128	Note 6	Black	Black	Non-Display
81	129	ū	Alt 129	Note 6	Black	Blue	Underline
82	130	é	Alt 130	Note 6	Black	Green	Normal
83	131	â	Alt 131	Note 6	Black	Cyan	Normal
84	132	ä	Alt 132	Note 6	Black	Red	Normal
85	133	à	Alt 133	Note 6	Black	Magenta	Normal
86	134	á	Alt 134	Note 6	Black	Brown	Normal
87	135	ç	Alt 135	Note 6	Black	Light Grey	Normal
88	136	ê	Alt 136	Note 6	Black	Dark Grey	Non-Display
89	137	ë	Alt 137	Note 6	Black	Light Blue	High Intensity Underline
8A	138	è	Alt 138	Note 6	Black	Light Green	High Intensity
8B	139	ï	Alt 139	Note 6	Black	Light Cyan	High Intensity
8C	140	î	Alt 140	Note 6	Black	Light Red	High Intensity
8D	141	í	Alt 141	Note 6	Black	Light Magenta	High Intensity
8E	142	Ā	Alt 142	Note 6	Black	Yellow	High Intensity
8F	143	Ă	Alt 143	Note 6	Black	White	High Intensity
90	144	É	Alt 144	Note 6	Blue	Black	Normal
91	145	æ	Alt 145	Note 6	Blue	Blue	Underline
92	146	Æ	Alt 146	Note 6	Blue	Green	Normal
93	147	ô	Alt 147	Note 6	Blue	Cyan	Normal
94	148	ö	Alt 148	Note 6	Blue	Red	Normal
95	149	õ	Alt 149	Note 6	Blue	Magenta	Normal
96	150	û	Alt 150	Note 6	Blue	Brown	Normal
97	151	ù	Alt 151	Note 6	Blue	Light Grey	Normal
98	152	ÿ	Alt 152	Note 6	Blue	Dark Grey	High Intensity
99	153	ö	Alt 153	Note 6	Blue	Light Blue	High Intensity Underline
9A	154	ü	Alt 154	Note 6	Blue	Light Green	High Intensity

C-6 Of Characters, Keystrokes, and Colors

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
9B	155	¢	Alt 155	Note 6	Blue	Light Cyan	High Intensity
9C	156	£	Alt 156	Note 6	Blue	Light Red	High Intensity
9D	157	¥	Alt 157	Note 6	Blue	Light Magenta	High Intensity
9E	158	Pt	Alt 158	Note 6	Blue	Yellow	High Intensity
9F	159	∫	Alt 159	Note 6	Blue	White	High Intensity
A0	160	á	Alt 160	Note 6	Green	Black	Normal
A1	161	í	Alt 161	Note 6	Green	Blue	Underline
A2	162	ó	Alt 162	Note 6	Green	Green	Normal
A3	163	ú	Alt 163	Note 6	Green	Cyan	Normal
A4	164	ñ	Alt 164	Note 6	Green	Red	Normal
A5	165	Ñ	Alt 165	Note 6	Green	Magenta	Normal
A6	166	<u>a</u>	Alt 166	Note 6	Green	Brown	Normal
A7	167	<u>o</u>	Alt 167	Note 6	Green	Light Grey	Normal
A8	168	¿	Alt 168	Note 6	Green	Dark Grey	High Intensity
A9	169	┐	Alt 169	Note 6	Green	Light Blue	High Intensity Underline
AA	170	└	Alt 170	Note 6	Green	Light Green	High Intensity
AB	171	½	Alt 171	Note 6	Green	Light Cyan	High Intensity
AC	172	¼	Alt 172	Note 6	Green	Light Red	High Intensity
AD	173	ı	Alt 173	Note 6	Green	Light Magenta	High Intensity
AE	174	<<	Alt 174	Note 6	Green	Yellow	High Intensity
AF	175	>>	Alt 175	Note 6	Green	White	High Intensity
B0	176	▒	Alt 176	Note 6	Cyan	Black	Normal
B1	177	▒	Alt 177	Note 6	Cyan	Blue	Underline
B2	178	▒	Alt 178	Note 6	Cyan	Green	Normal
B3	179	▒	Alt 179	Note 6	Cyan	Cyan	Normal
B4	180	▒	Alt 180	Note 6	Cyan	Red	Normal
B5	181	▒	Alt 181	Note 6	Cyan	Magenta	Normal
B6	182	▒	Alt 182	Note 6	Cyan	Brown	Normal

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
B7	183		Alt 183	Note 6	Cyan	Light Grey	Normal
B8	184		Alt 184	Note 6	Cyan	Dark Grey	High Intensity
B9	185		Alt 185	Note 6	Cyan	Light Blue	High Intensity Underline
BA	186		Alt 186	Note 6	Cyan	Light Green	High Intensity
BB	187		Alt 187	Note 6	Cyan	Light Cyan	High Intensity
BC	188		Alt 188	Note 6	Cyan	Light Red	High Intensity
BD	189		Alt 189	Note 6	Cyan	Light Magenta	High Intensity
BE	190		Alt 190	Note 6	Cyan	Yellow	High Intensity
BF	191		Alt 191	Note 6	Cyan	White	High Intensity
C0	192		Alt 192	Note 6	Red	Black	Normal
C1	193		Alt 193	Note 6	Red	Blue	Underline
C2	194		Alt 194	Note 6	Red	Green	Normal
C3	195		Alt 195	Note 6	Red	Cyan	Normal
C4	196		Alt 196	Note 6	Red	Red	Normal
C5	197		Alt 197	Note 6	Red	Magenta	Normal
C6	198		Alt 198	Note 6	Red	Brown	Normal
C7	199		Alt 199	Note 6	Red	Light Grey	Normal
C8	200		Alt 200	Note 6	Red	Dark Grey	High Intensity
C9	201		Alt 201	Note 6	Red	Light Blue	High Intensity Underline
CA	202		Alt 202	Note 6	Red	Light Green	High Intensity
CB	203		Alt 203	Note 6	Red	Light Cyan	High Intensity
CC	204		Alt 204	Note 6	Red	Light Red	High Intensity
CD	205		Alt 205	Note 6	Red	Light Magenta	High Intensity
CE	206		Alt 206	Note 6	Red	Yellow	High Intensity
CF	207		Alt 207	Note 6	Red	White	High Intensity
D0	208		Alt 208	Note 6	Magenta	Black	Normal

C-8 Of Characters, Keystrokes, and Colors

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
D1	209		Alt 209	Note 6	Magenta	Blue	Underline
D2	210		Alt 210	Note 6	Magenta	Green	Normal
D3	211		Alt 211	Note 6	Magenta	Cyan	Normal
D4	212		Alt 212	Note 6	Magenta	Red	Normal
D5	213		Alt 213	Note 6	Magenta	Magenta	Normal
D6	214		Alt 214	Note 6	Magenta	Brown	Normal
D7	215		Alt 215	Note 6	Magenta	Light Grey	Normal
D8	216		Alt 216	Note 6	Magenta	Dark Grey	High Intensity
D9	217		Alt 217	Note 6	Magenta	Light Blue	High Intensity Underline
DA	218		Alt 218	Note 6	Magenta	Light Green	High Intensity
DB	219		Alt 219	Note 6	Magenta	Light Cyan	High Intensity
DC	220		Alt 220	Note 6	Magenta	Light Red	High Intensity
DD	221		Alt 221	Note 6	Magenta	Light Magenta	High Intensity
DE	222		Alt 222	Note 6	Magenta	Yellow	High Intensity
DF	223		Alt 223	Note 6	Magenta	White	High Intensity
E0	224	α	Alt 224	Note 6	Yellow	Black	Normal
E1	225	β	Alt 225	Note 6	Yellow	Blue	Underline
E2	226	Γ	Alt 226	Note 6	Yellow	Green	Normal
E3	227	π	Alt 227	Note 6	Yellow	Cyan	Normal
E4	228	Σ	Alt 228	Note 6	Yellow	Red	Normal
E5	229	σ	Alt 229	Note 6	Yellow	Magenta	Normal
E6	230	μ	Alt 230	Note 6	Yellow	Brown	Normal
E7	231	τ	Alt 231	Note 6	Yellow	Light Grey	Normal
E8	232	Φ	Alt 232	Note 6	Yellow	Dark Grey	High Intensity
E9	233	θ	Alt 233	Note 6	Yellow	Light Blue	High Intensity Underline
EA	234	Ω	Alt 234	Note 6	Yellow	Light Green	High Intensity
EB	235	δ	Alt 235	Note 6	Yellow	Light Cyan	High Intensity

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
EC	236	∞	Alt 236	Note 6	Yellow	Light Red	High Intensity
ED	237	ϕ	Alt 237	Note 6	Yellow	Light Magenta	High Intensity
EE	238	€	Alt 238	Note 6	Yellow	Yellow	High Intensity
EF	239	∩	Alt 239	Note 6	Yellow	White	High Intensity
F0	240	≡	Alt 240	Note 6	White	Black	Reverse Video
F1	241	±	Alt 241	Note 6	White	Blue	Underline
F2	242	≥	Alt 242	Note 6	White	Green	Normal
F3	243	≤	Alt 243	Note 6	White	Cyan	Normal
F4	244	∫	Alt 244	Note 6	White	Red	Normal
F5	245	∫	Alt 245	Note 6	White	Magenta	Normal
F6	246	÷	Alt 246	Note 6	White	Brown	Normal
F7	247	≈	Alt 247	Note 6	White	Light Grey	Normal
F8	248	○	Alt 248	Note 6	White	Dark Grey	Reverse Video
F9	249	●	Alt 249	Note 6	White	Light Blue	High Intensity Underline
FA	250	•	Alt 250	Note 6	White	Light Green	High Intensity
FB	251	√	Alt 251	Note 6	White	Light Cyan	High Intensity
FC	252	η	Alt 252	Note 6	White	Light Red	High Intensity
FD	253	2	Alt 253	Note 6	White	Light Magenta	High Intensity
FE	254	■	Alt 254	Note 6	White	Yellow	High Intensity
FF	255	BLANK	Alt 255	Note 6	White	White	High Intensity

NOTE 1 Asterisk (*) can easily be keyed using two methods:

1) hit the  key or 2) in shift mode hit the

 key.

NOTE 2 Period (.) can easily be keyed using two methods:

1) hit the  key or 2) in shift or Num Lock

mode hit the  key.

NOTE 3 Numeric characters (0—9) can easily be keyed using two methods: 1) hit the numeric keys on the top row of the typewriter portion of the keyboard or 2) in shift or Num Lock mode hit the numeric keys in the 10—key pad portion of the keyboard.

NOTE 4 Upper case alphabetic characters (A—Z) can easily be keyed in two modes: 1) in shift mode the appropriate alphabetic key or 2) in Caps Lock mode hit the appropriate alphabetic key.

NOTE 5 Lower case alphabetic characters (a—z) can easily be keyed in two modes: 1) in "normal" mode hit the appropriate key or 2) in Caps Lock combined with shift mode hit the appropriate alphabetic key.

NOTE 6 The 3 digits after the Alt key must be typed from the numeric key pad (keys 71—73, 75—77, 79—82). Character codes 000 through 255 can be entered in this fashion. (With Caps Lock activated, Character codes 97 through 122 will display upper case rather than lower case alphabetic characters.)

Character Set (00-7F) Quick Reference

DECIMAL VALUE		0	16	32	48	64	80	96	112
 HEXA DECIMAL VALUE		0	1	2	3	4	5	6	7
0	0	BLANK (NULL)		BLANK (SPACE)	0	@	P	'	p
1	1			!	1	A	Q	a	q
2	2			"	2	B	R	b	r
3	3		!!	#	3	C	S	c	s
4	4		¶	\$	4	D	T	d	t
5	5		§	%	5	E	U	e	u
6	6		▬	&	6	F	V	f	v
7	7	•		'	7	G	W	g	w
8	8	•	↑	(	8	H	X	h	x
9	9	○	↓	)	9	I	Y	i	y
10	A	○	→	*	:	J	Z	j	z
11	B	♂	←	+	;	K	I	k	{
12	C	♀	└	,	<	L	\	l	!
13	D	♪	↔	—	=	M	I	m	}
14	E	♪	▲	.	>	N	^	n	~
15	F	☀	▼	/	?	O	—	o	△

Character Set (80-FF) Quick Reference

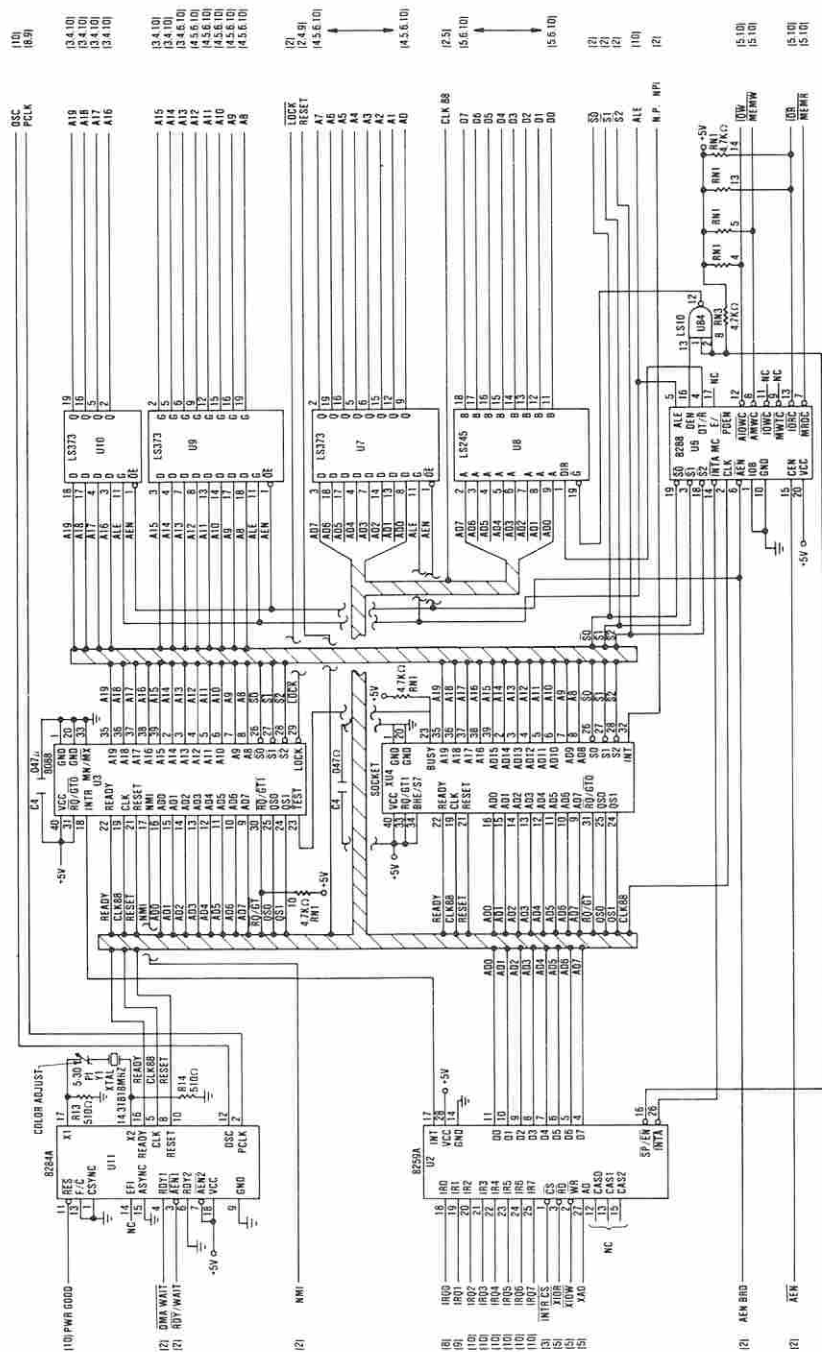
DECIMAL VALUE		128	144	160	176	192	208	224	240
	HEXA DECIMAL VALUE	8	9	A	B	C	D	E	F
0	0	Ç	É	á				∞	≡
1	1	ü	æ	í				β	±
2	2	é	Æ	ó				Γ	≥
3	3	â	ô	ú				π	≤
4	4	ä	ö	ñ				Σ	∫
5	5	à	ò	Ñ				σ	∫
6	6	å	û	ä				μ	÷
7	7	ç	ù	o				τ	≈
8	8	ê	ÿ	ı				ø	°
9	9	ë	Ö	┐				θ	•
10	A	è	Ü	└				Ω	•
11	B	ï	ç	½				δ	√
12	C	î	£	¼				∞	n
13	D	ì	¥	ı				φ	²
14	E	Ä	℞	«				€	
15	F	Å	ƒ	»				∩	BLANK 'FF'

Notes:

APPENDIX D: LOGIC DIAGRAMS

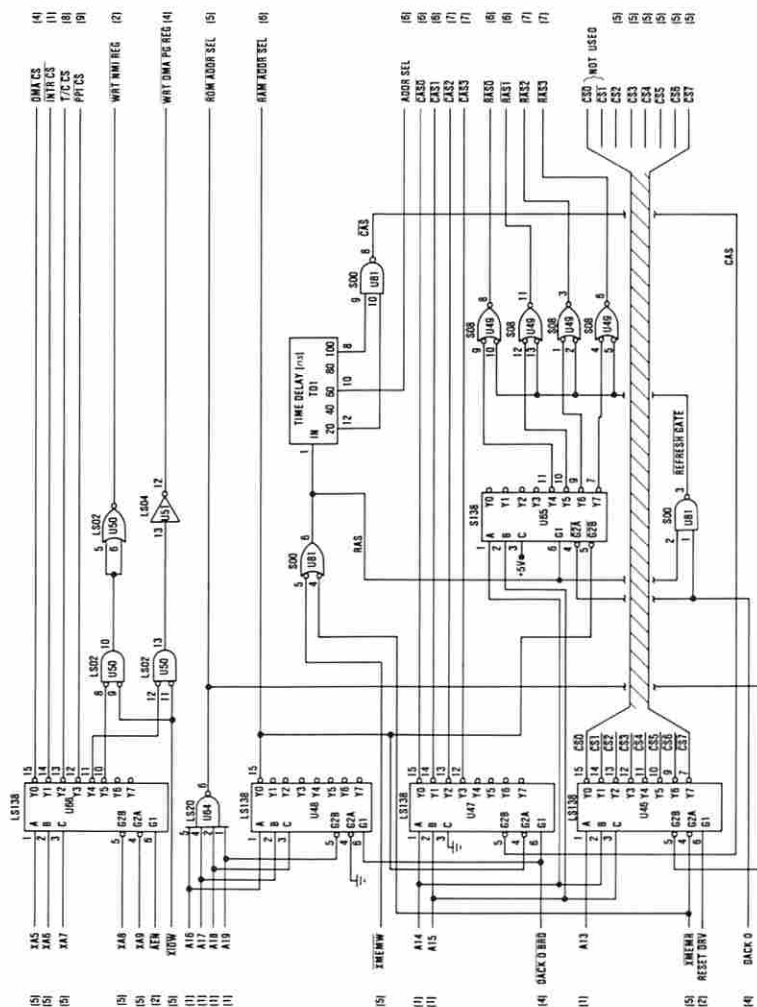
System Board	D-3
Keyboard	D-13
Expansion Board	D-15
Extender Card	D-16
Receiver Card	D-19
Printer	D-22
Printer Adapter	D-25
Monochrome Display Adapter	D-26
Color/Graphics Monitor Adapter	D-36
Color Display	D-42
Monochrome Display	D-44
5-1/4 Inch Diskette Drive Adapter	D-46
5-1/4 Inch Diskette Drive – Type 1	D-50
5-1/4 Inch Diskette Drive – Type 2	D-53
Fixed Disk Drive Adapter	D-55
Fixed Disk Drive – Type 1	D-61
Fixed Disk Drive – Type 2	D-64
32K Memory Expansion Option	D-67
64K Memory Expansion Option	D-70
64/256K Memory Expansion Option	D-73
Game Control Adapter	D-77
Prototype Card	D-78
Asynchronous Communications Adapter	D-79
SDLC Communications Adapter	D-80

Notes:



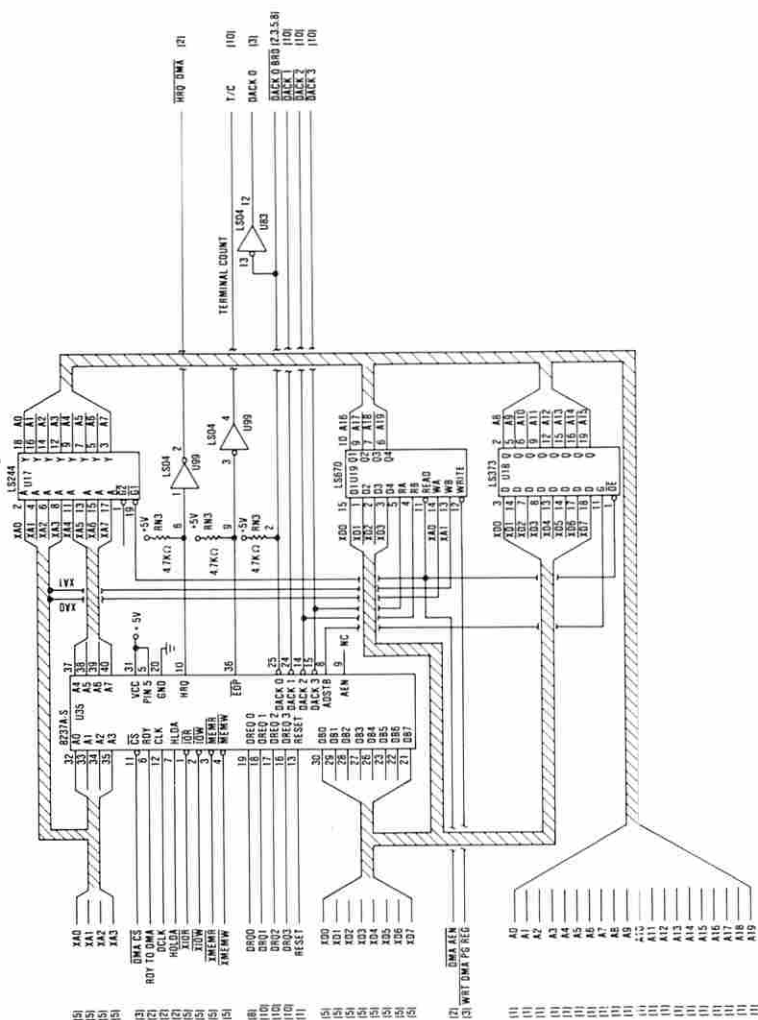
Logic Diagrams D-3



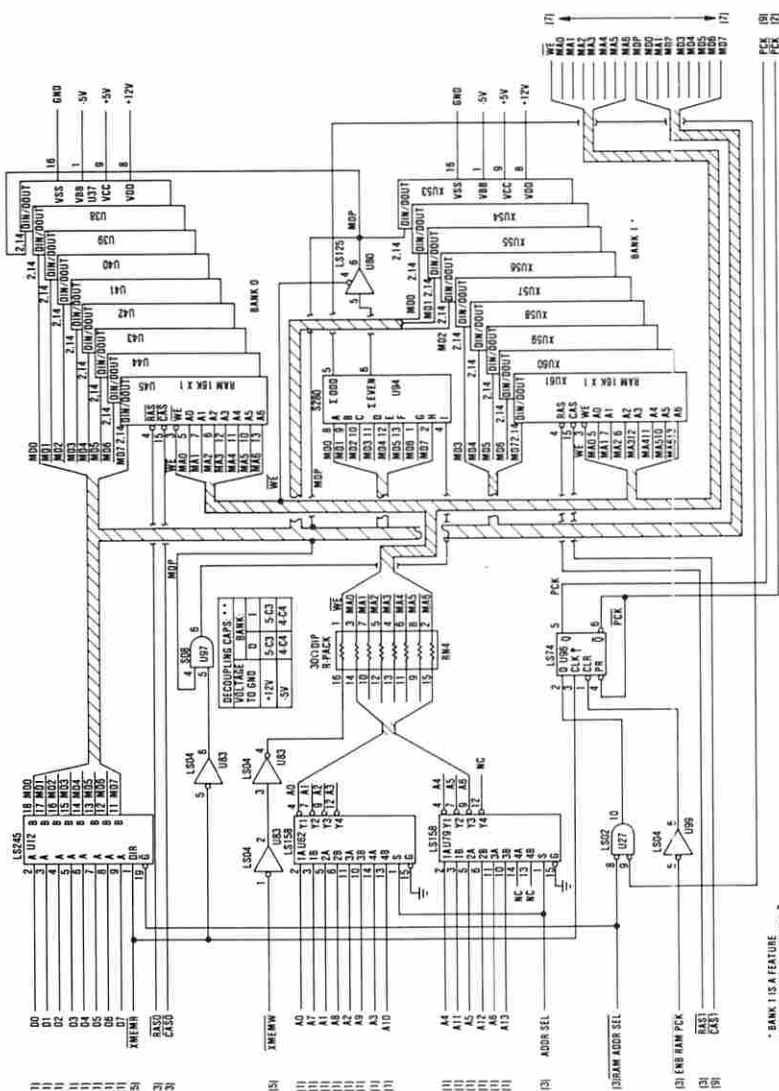


System Board (Sheet 3 of 10)

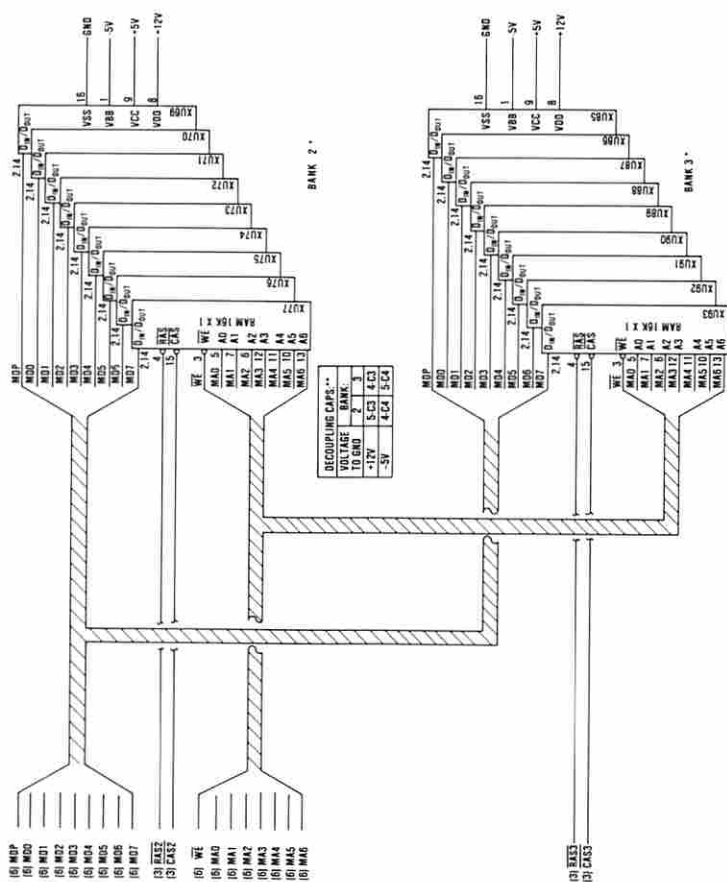
D-6 Logic Diagrams



D-8 Logic Diagrams



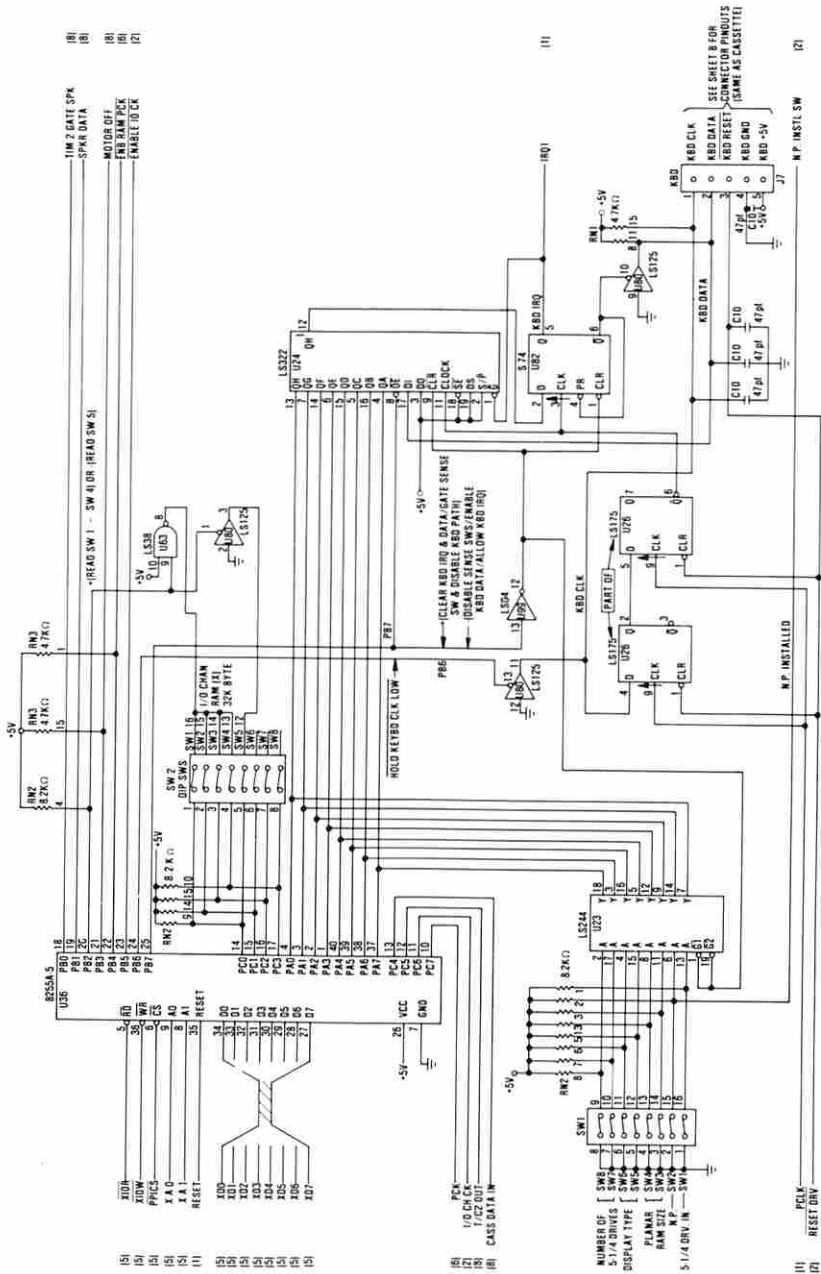
* BANK 1 IS A FEATURE
 ** C3 = 041/F -15WIDE } DECOUPLING
 C4 = 041/F -7WIDE

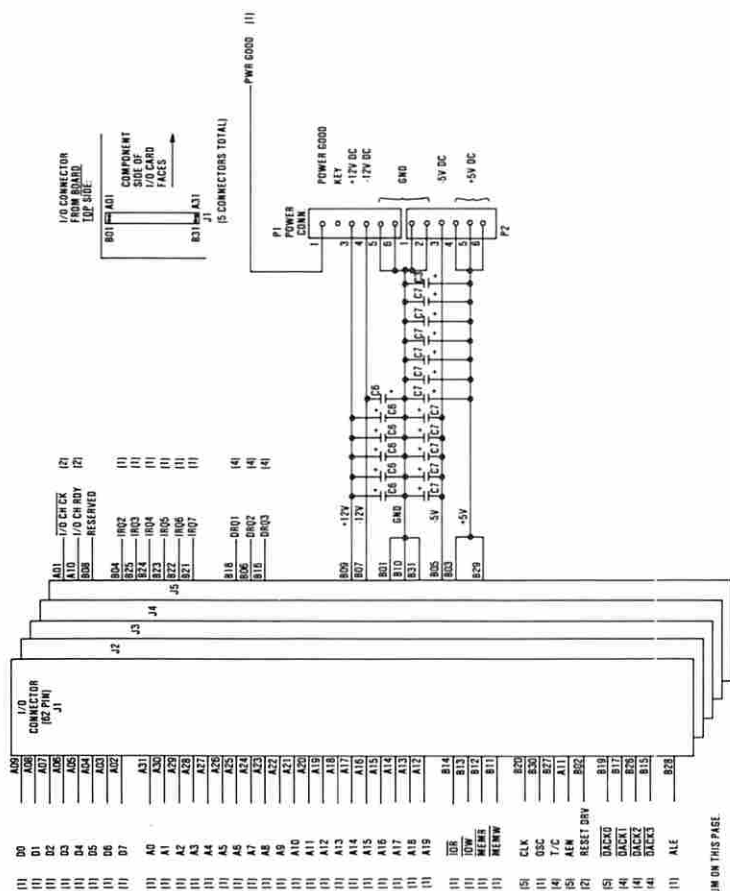


* BANKS 2 & 3 ARE FEATURES.
 ** C3 = 0.01µF ±5%VDDC
 C4 = 0.01µF ±5%VDDC
 C5 = 0.01µF ±5%VDDC
 C6 = 0.01µF ±5%VDDC

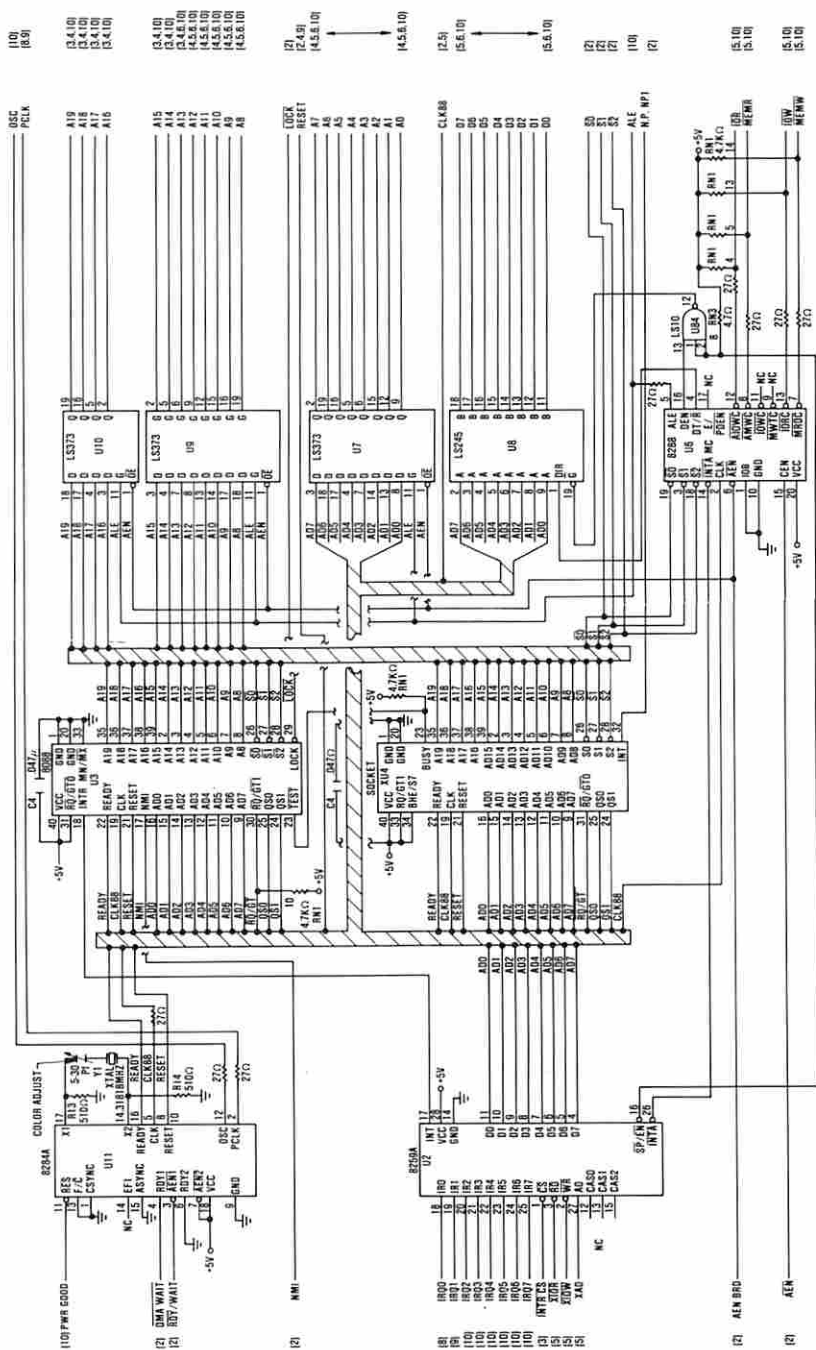
System Board (Sheet 7 of 10)

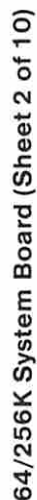
System Board (Sheet 9 of 10)

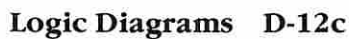




NOTE:
1. ALL CAPS ARE 8.2µF TANTALUM ON THIS PAGE.

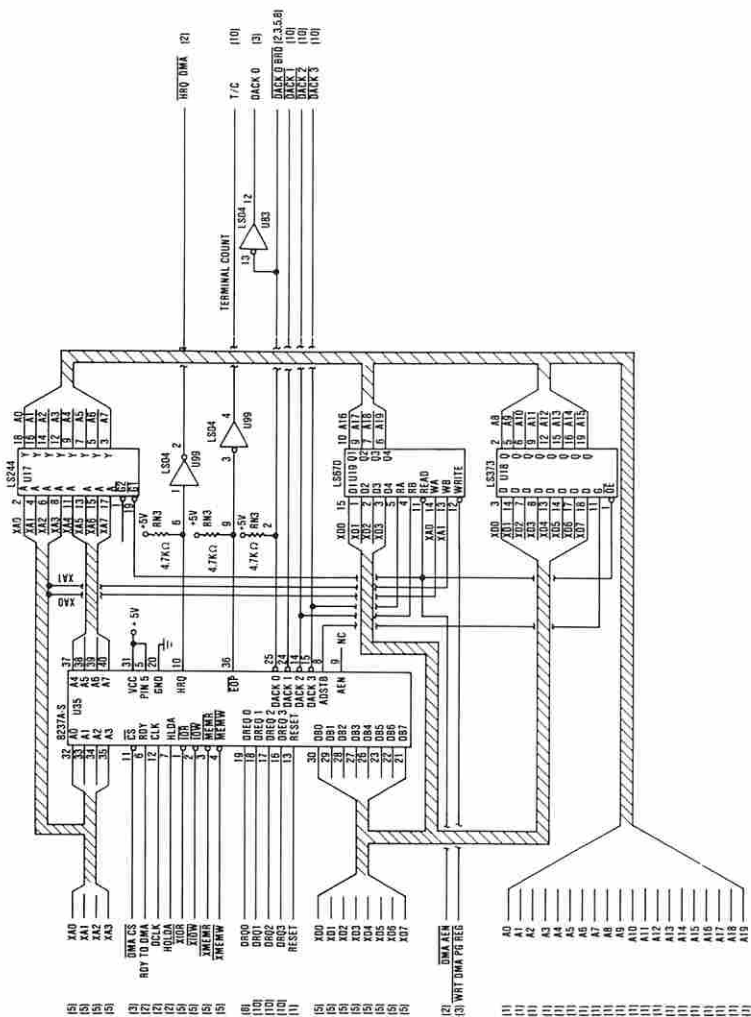




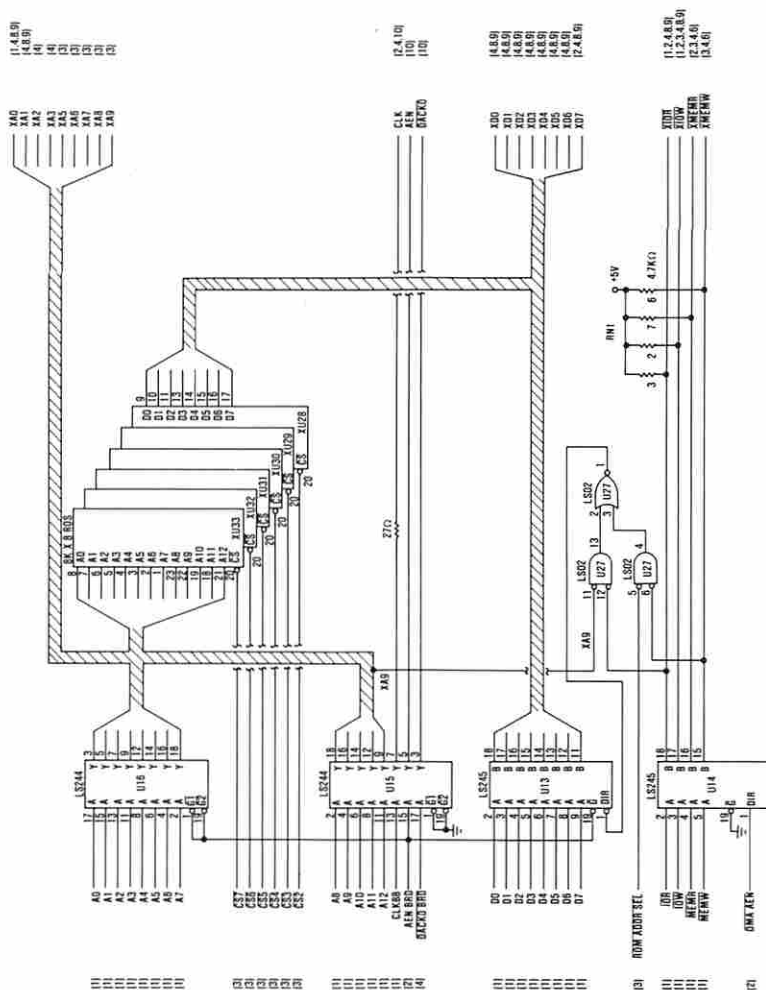


D-12d Logic Diagrams

© Copyright IBM Corp. 1981, 1982, 1983



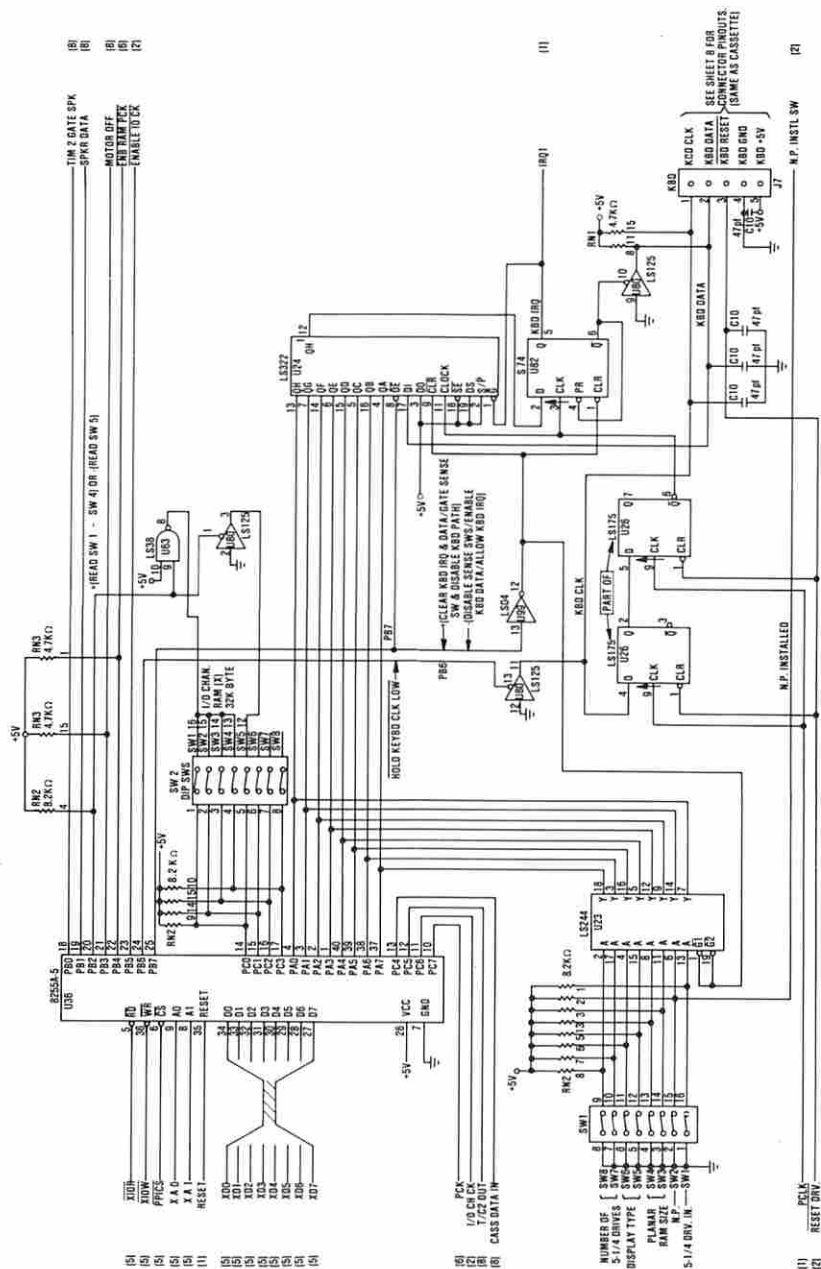
64/256K System Board (Sheet 4 of 10)



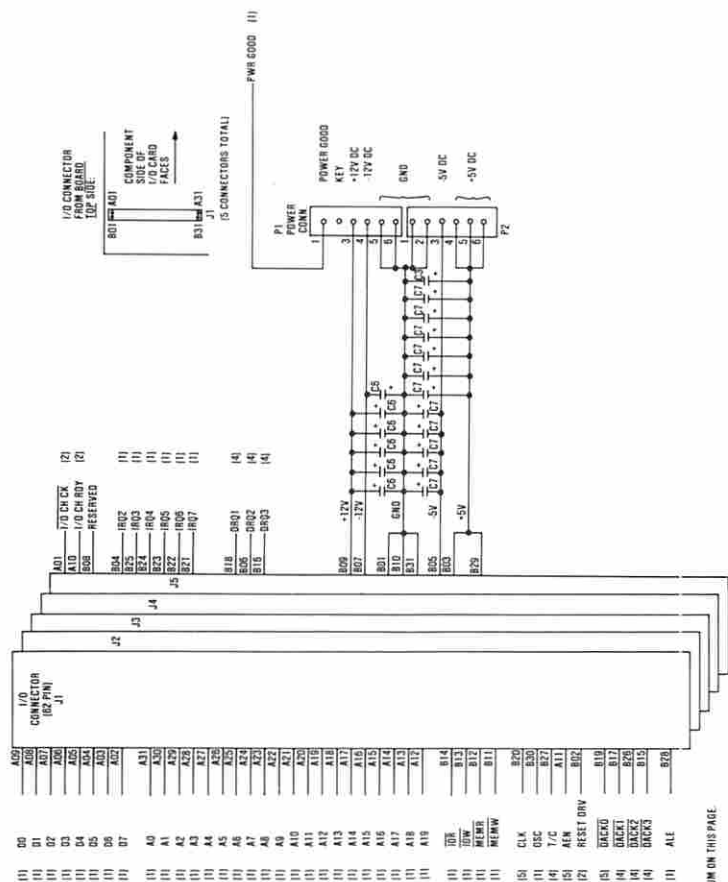


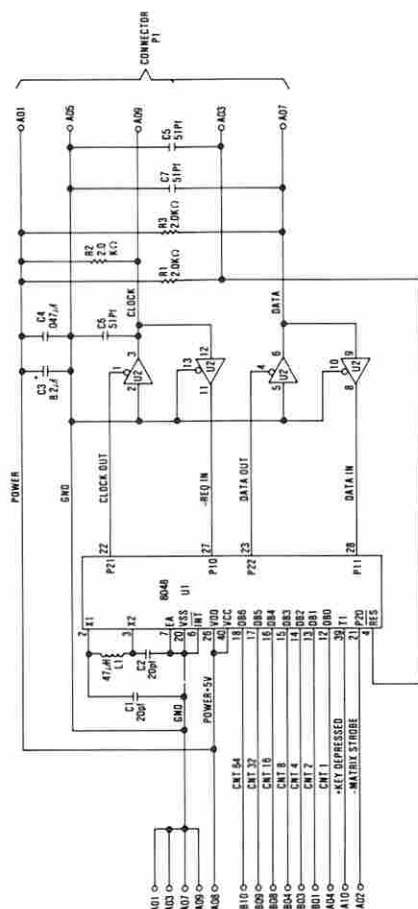
PCK	[9]
PCK	[2]





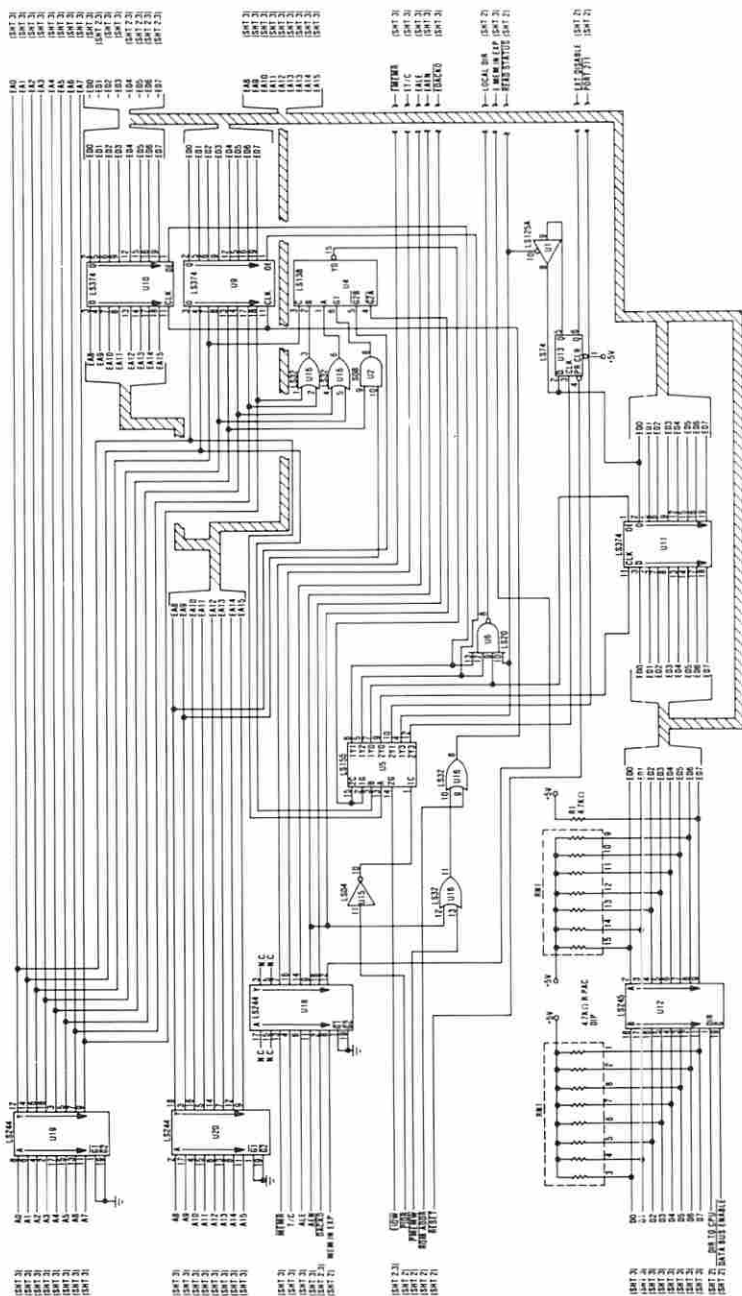
64/256K System Board (Sheet 9 of 10)



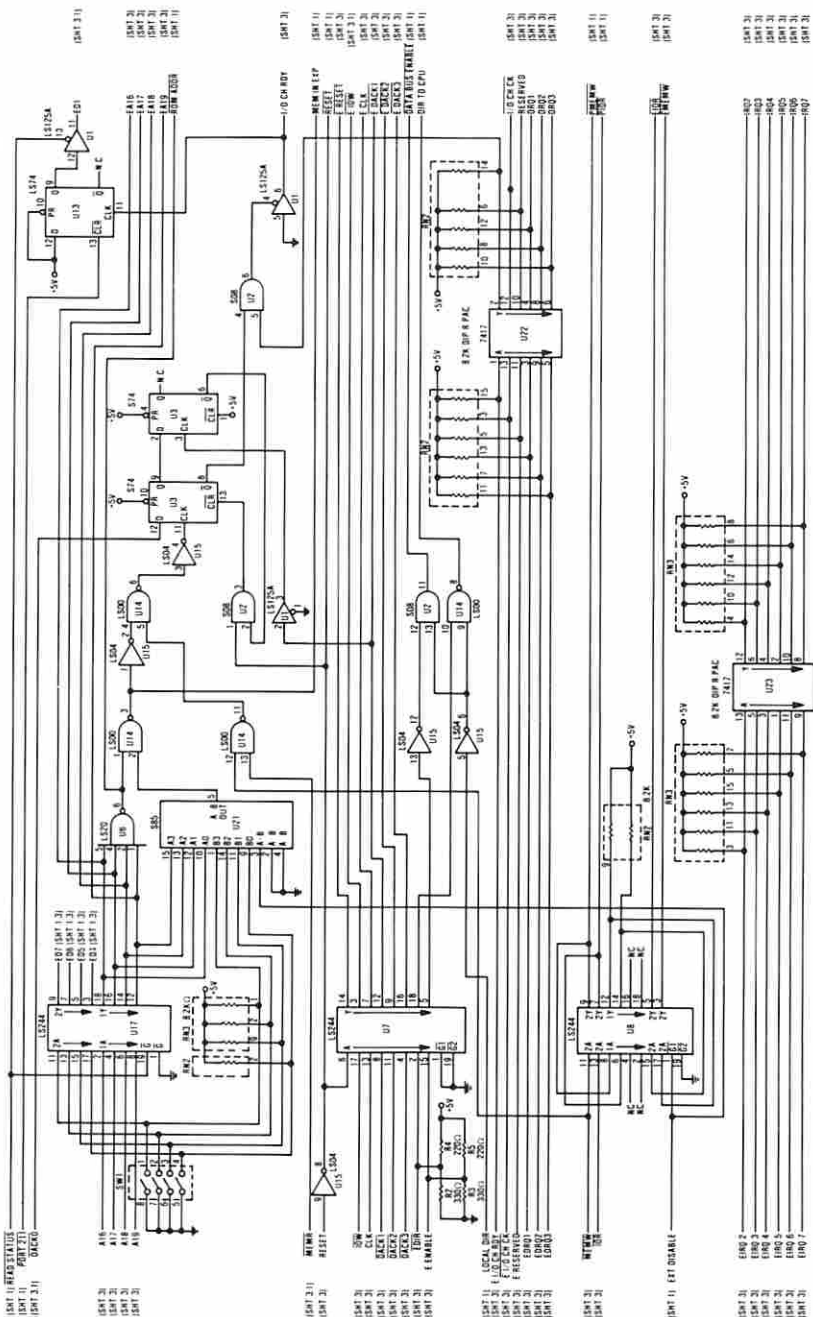


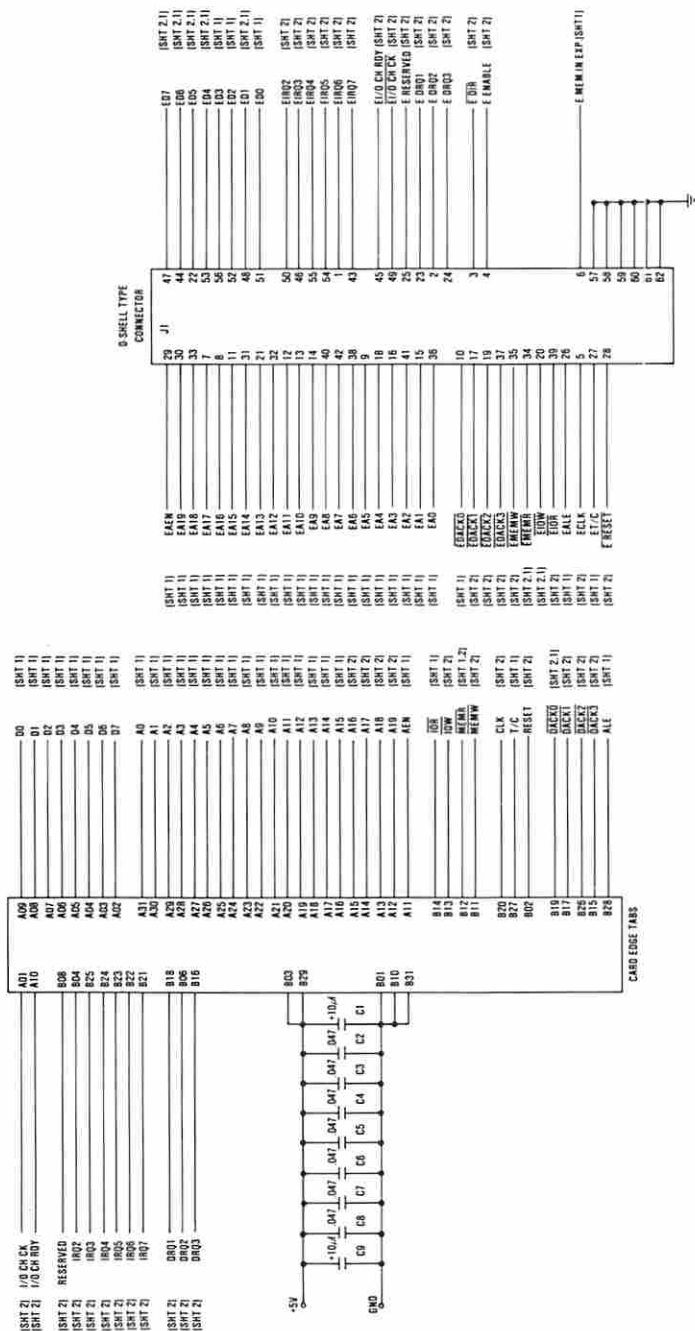
Keyboard (Sheet 1 of 2)





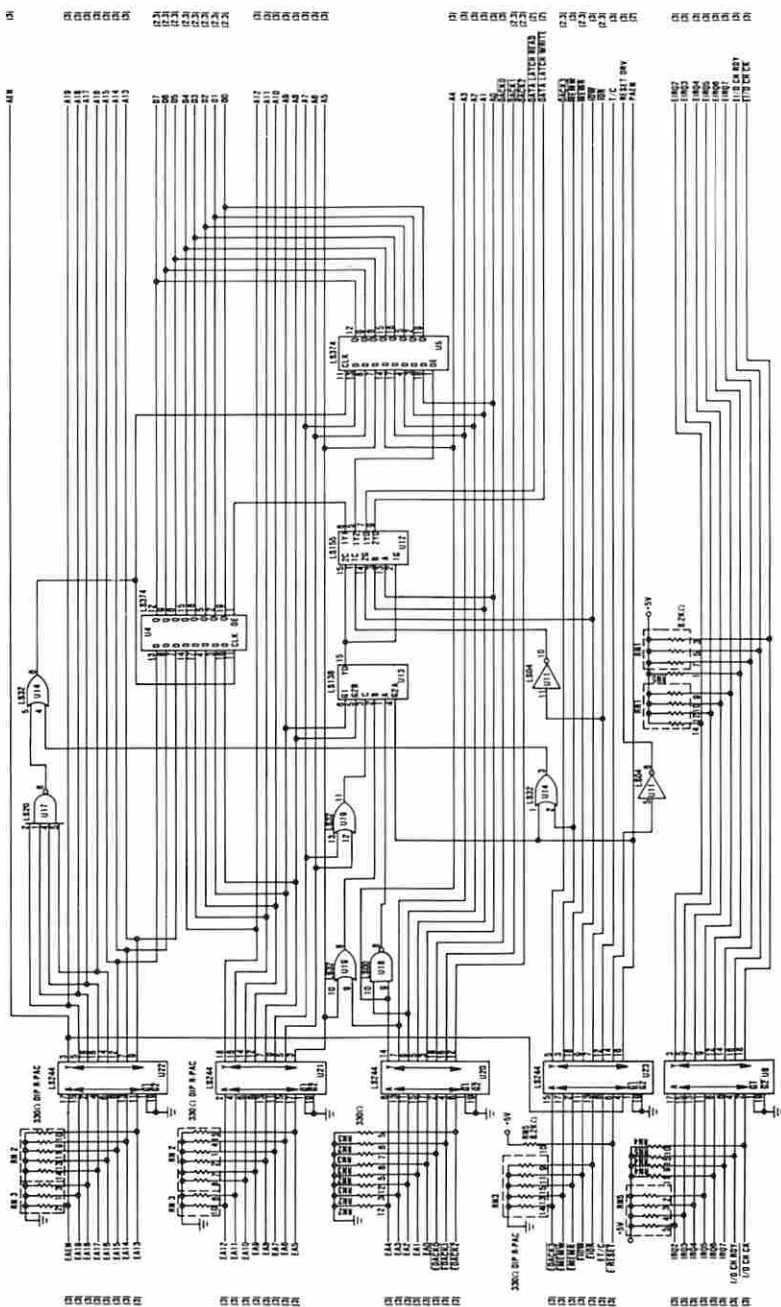
Extender Card (Sheet 1 of 3)

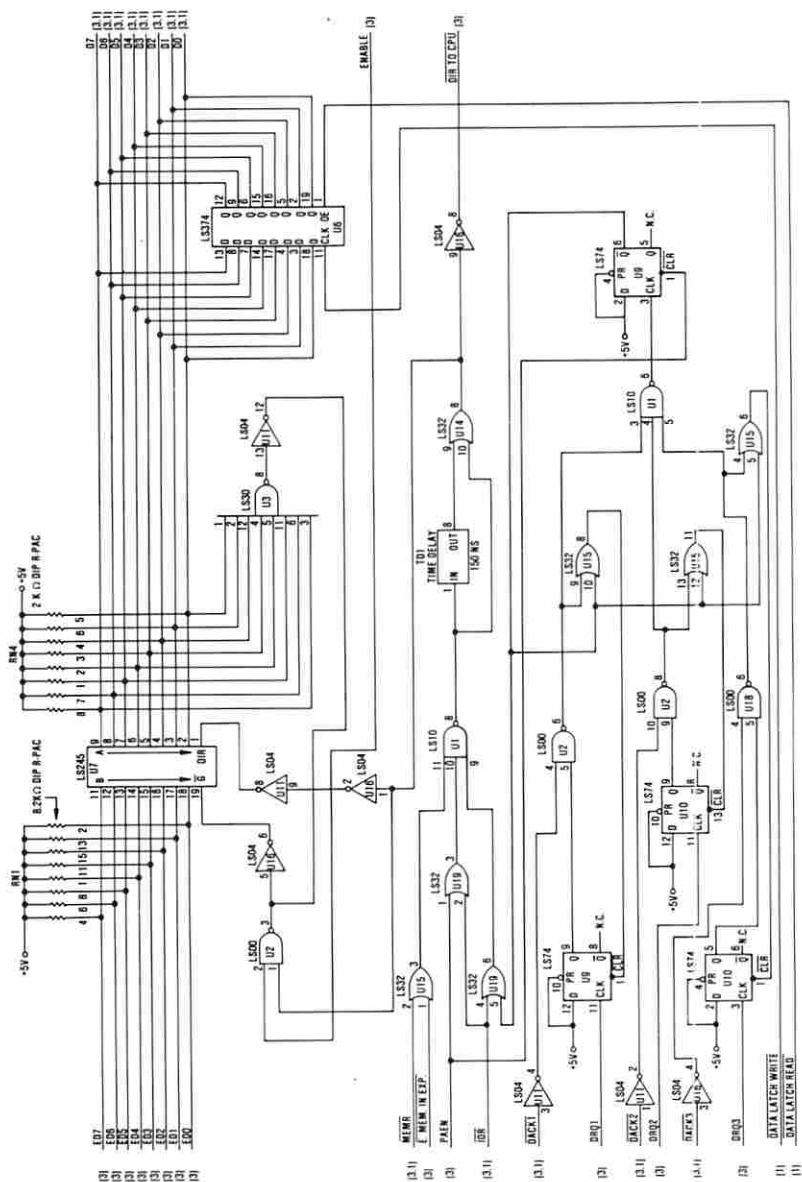


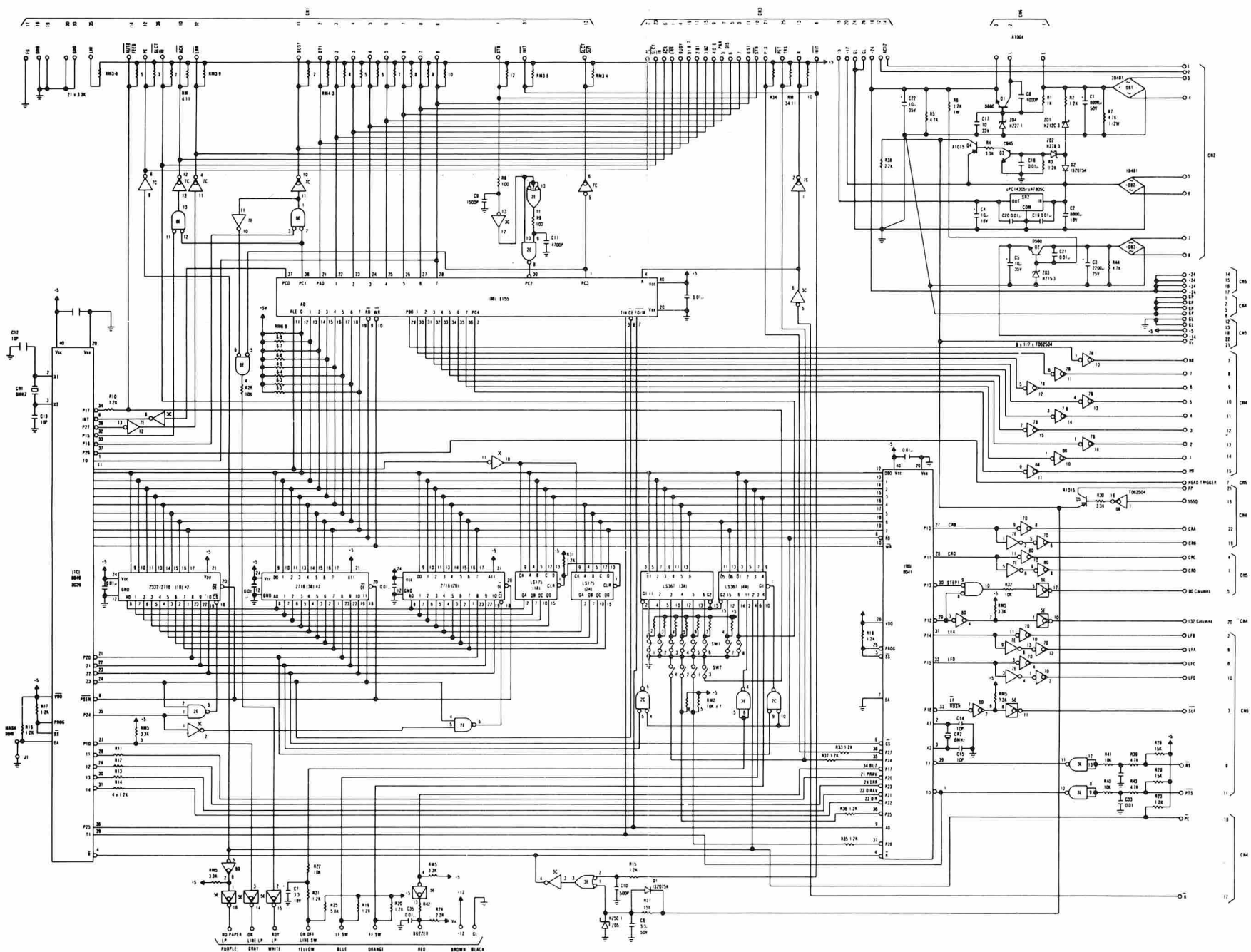


Extender Card (Sheet 3 of 3)

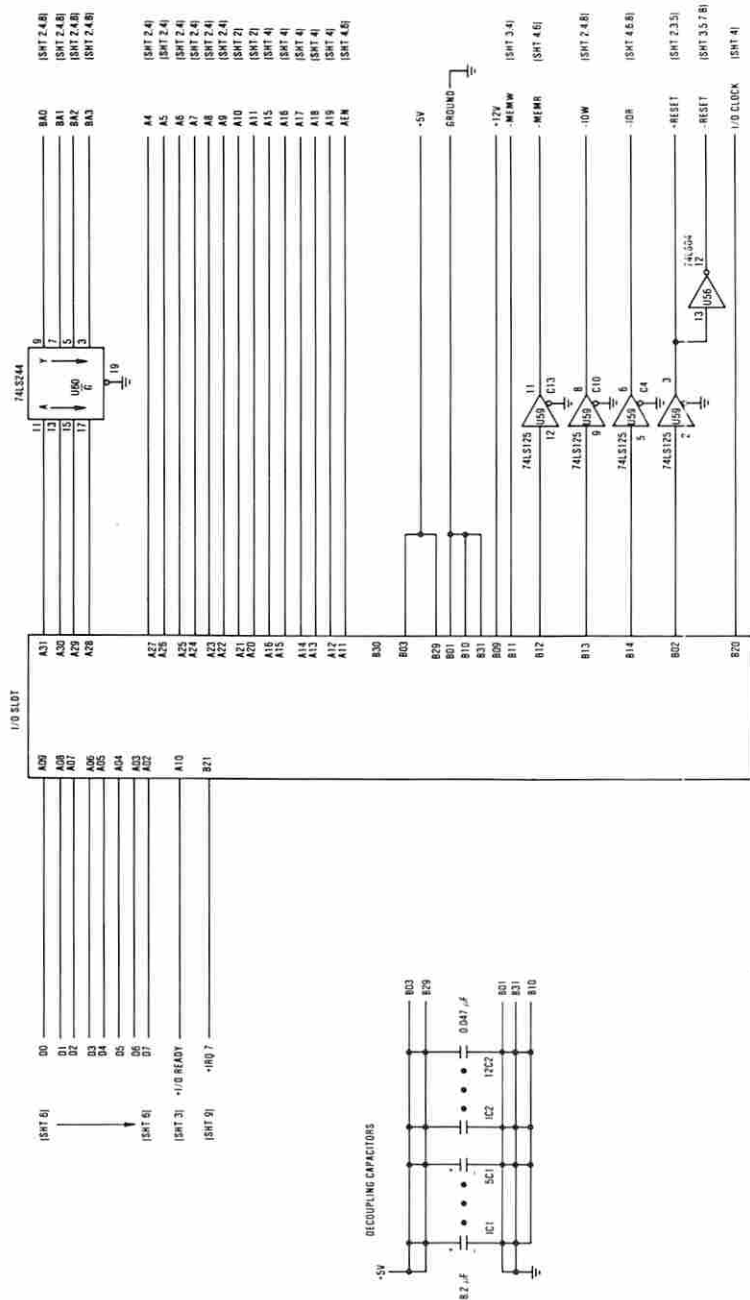
Receiver Card (Sheet 1 of 3)





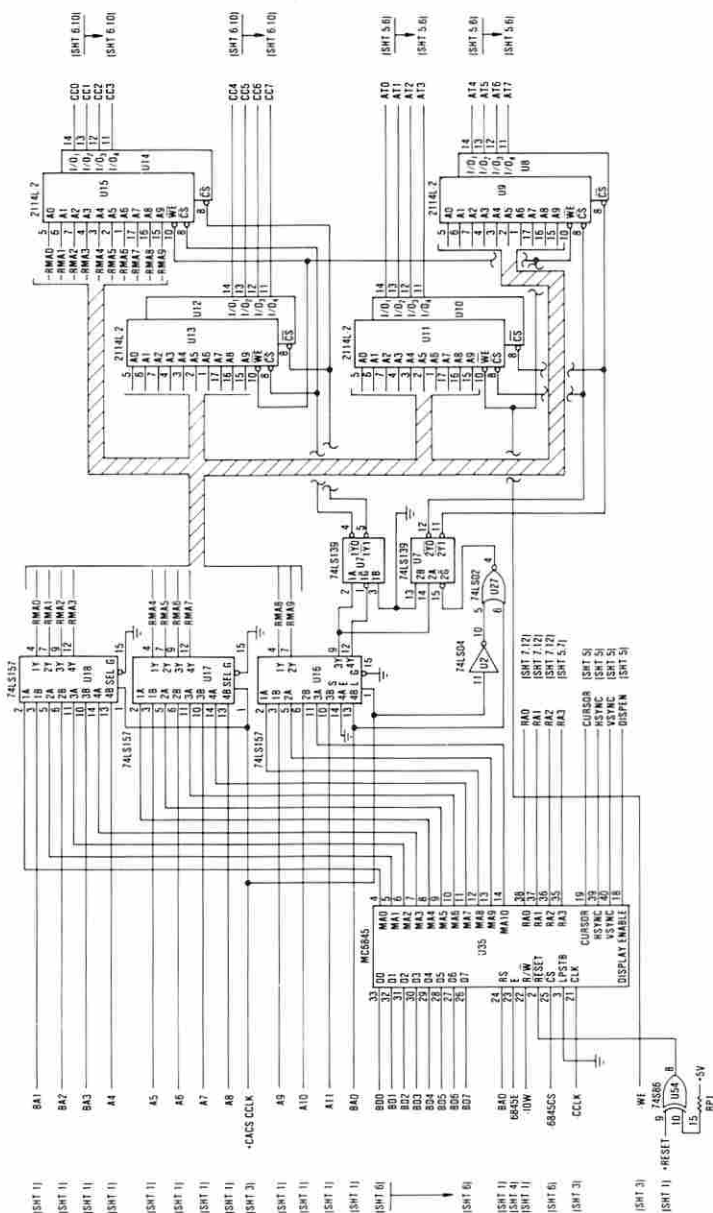


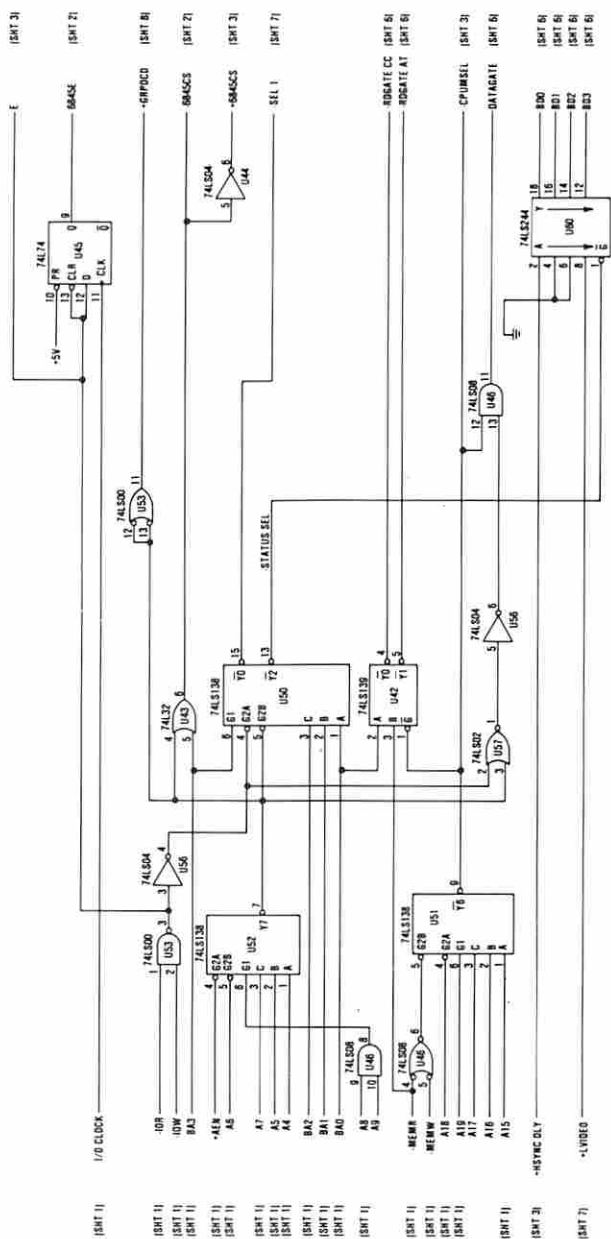
IBM 80 CPS GRAPHICS PRINTER



Monochrome Display Adapter (Sheet 1 of 10)

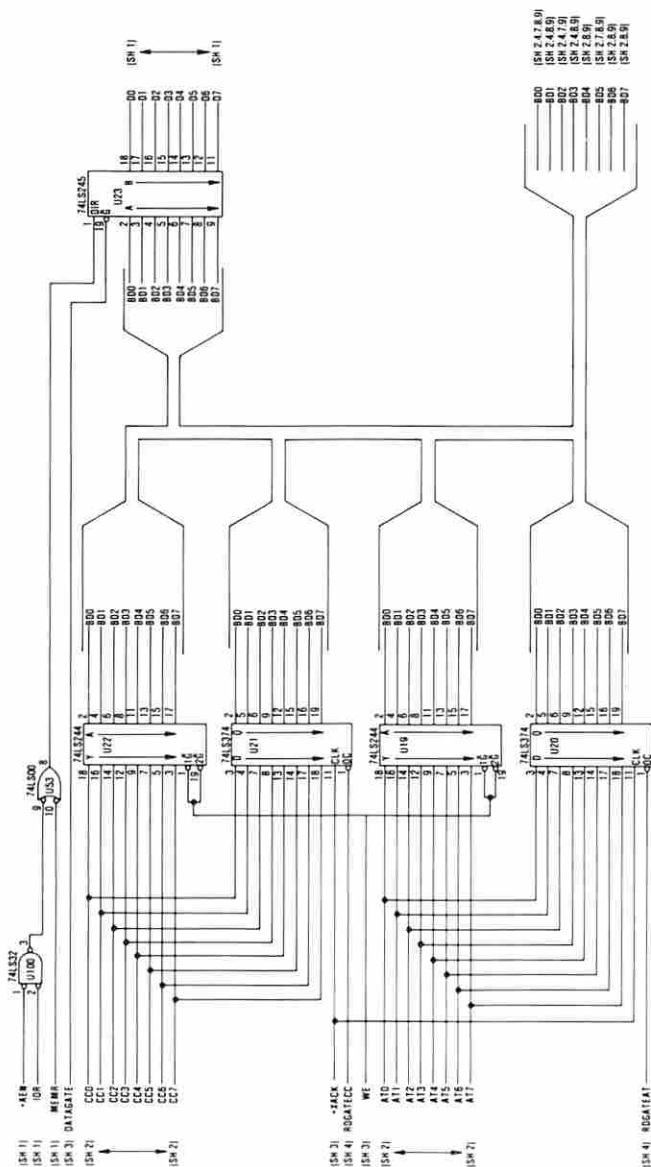
Monochrome Display Adapter (Sheet 2 of 10)



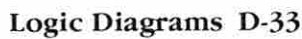


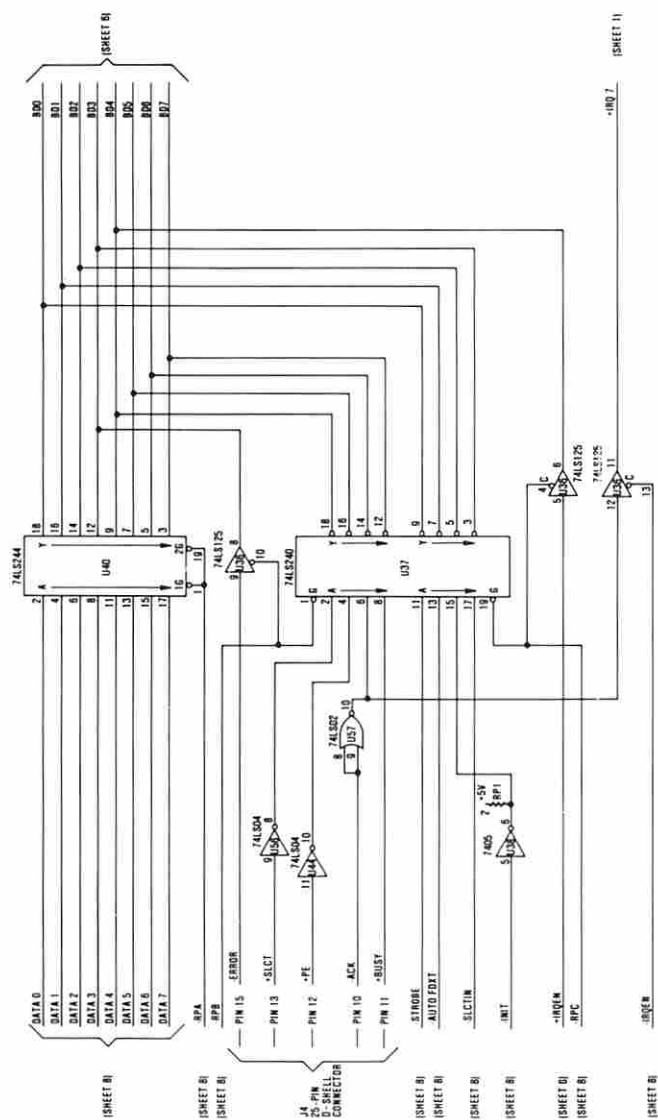
Monochrome Display Adapter (Sheet 4 of 10)





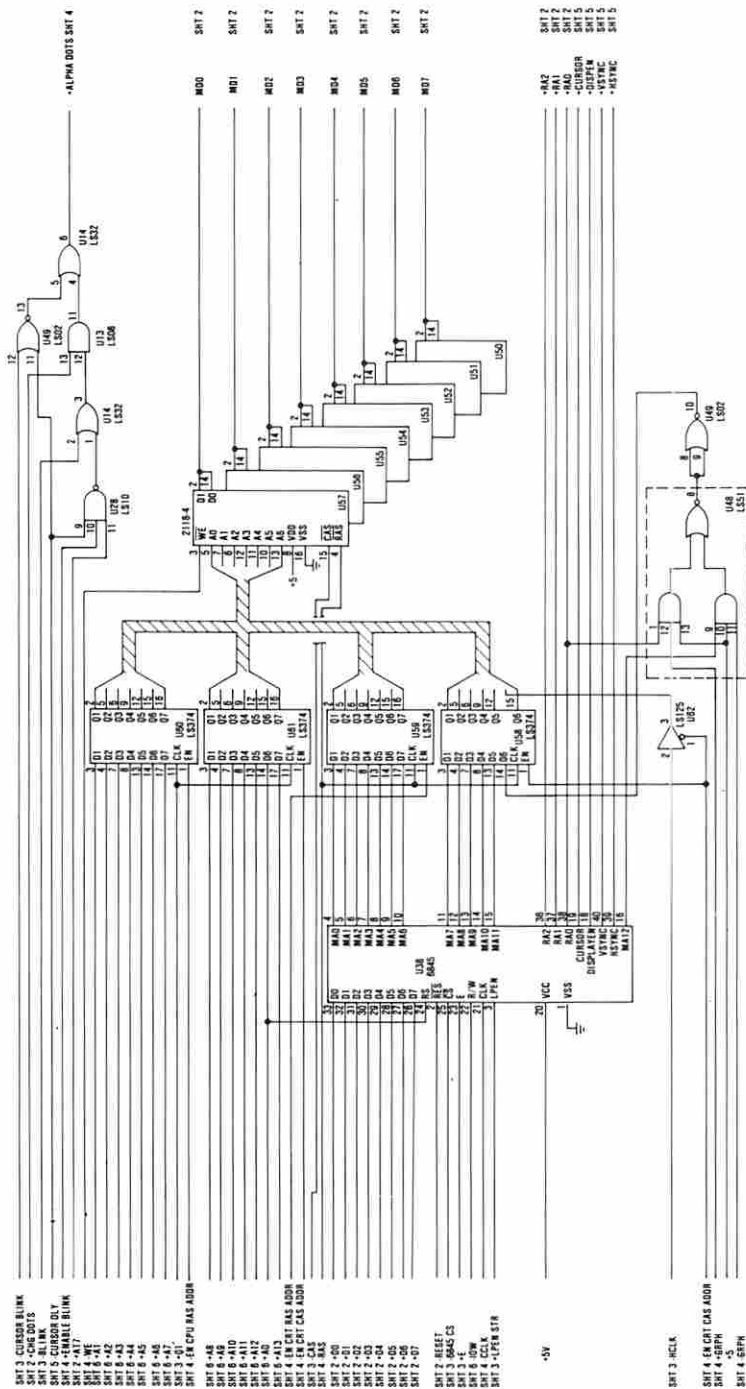
Monochrome Display Adapter (Sheet 6 of 10)



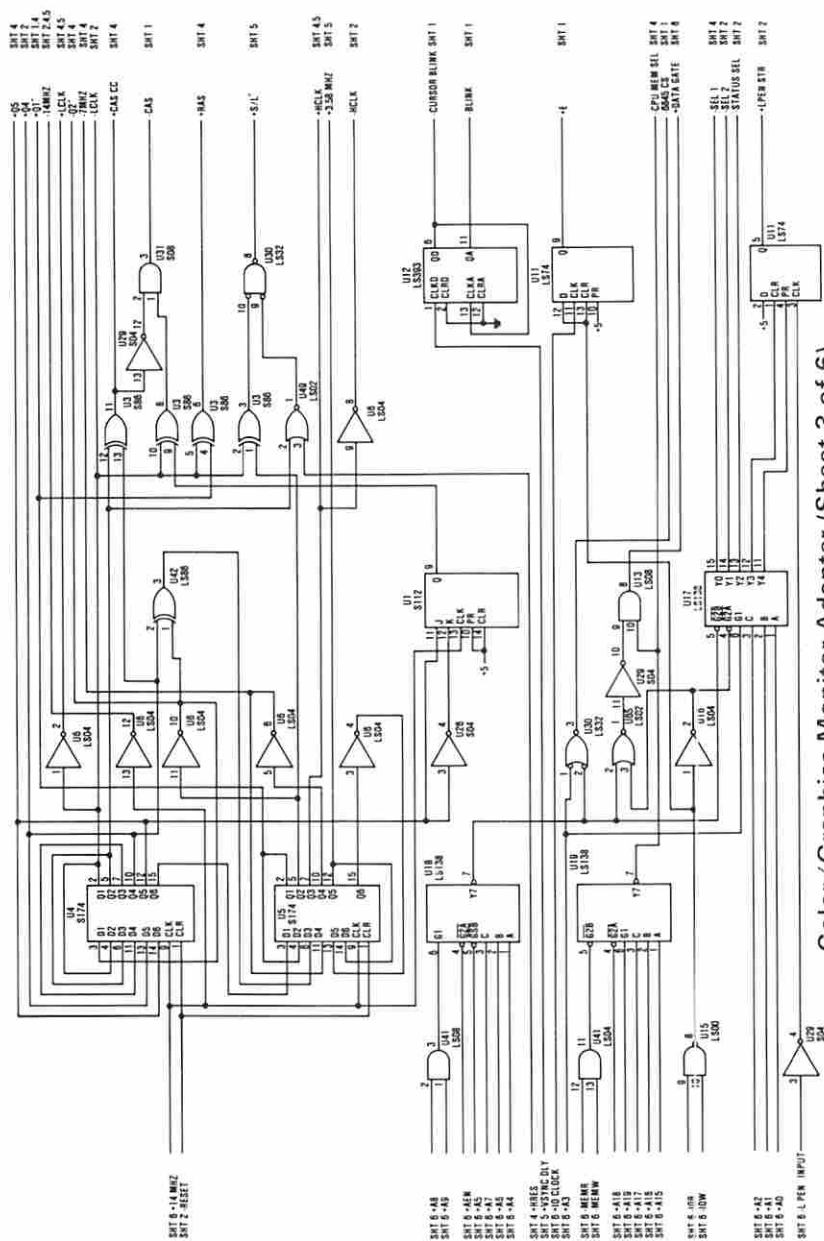


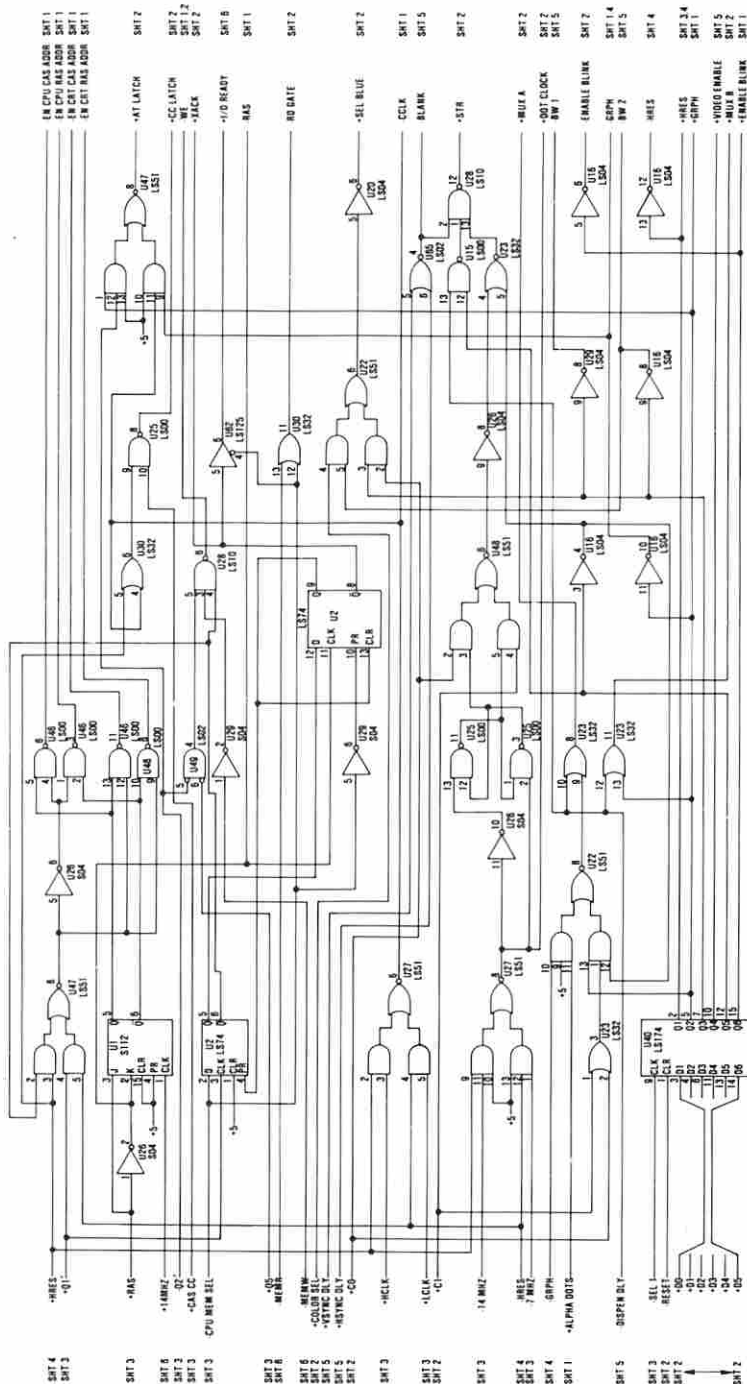
[illegible]

D-36 Logic Diagrams

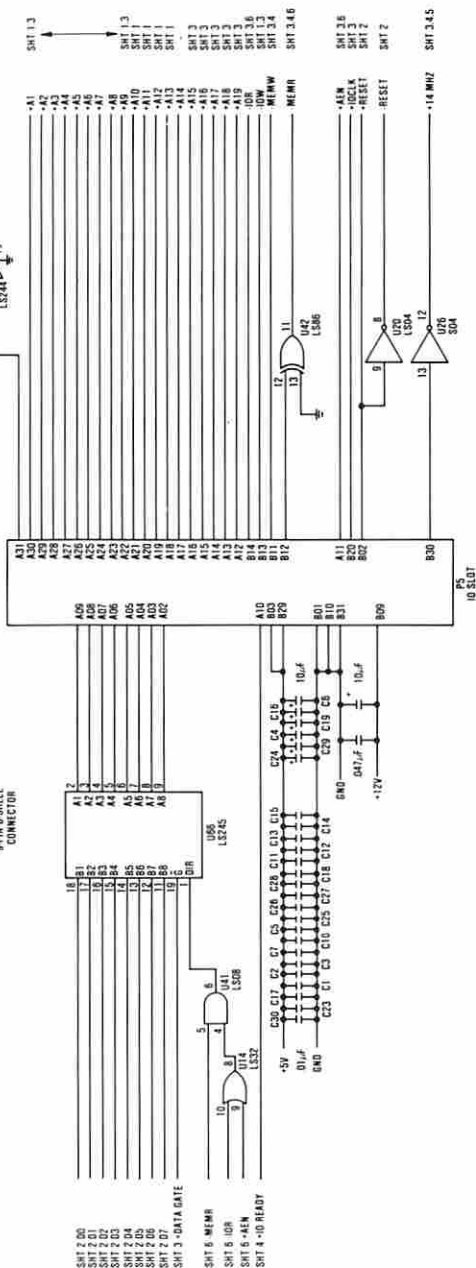


Color/Graphics Monitor Adapter (Sheet 1 of 6)





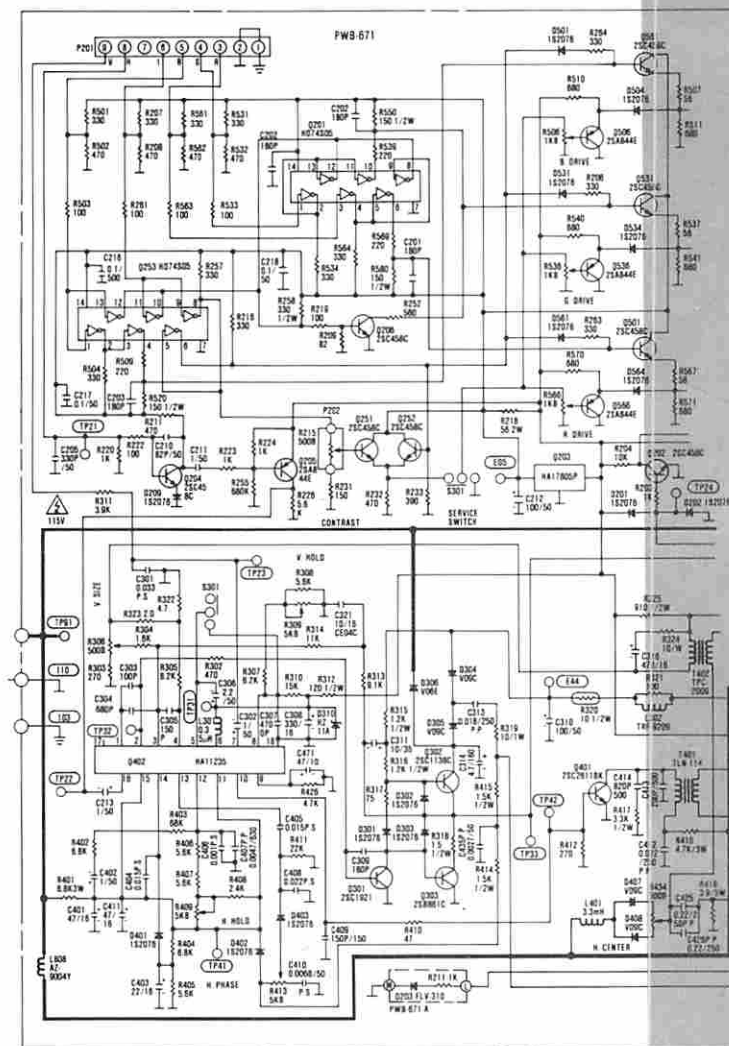




Color/Graphics Monitor Adapter (Sheet 6 of 6)

DANGER

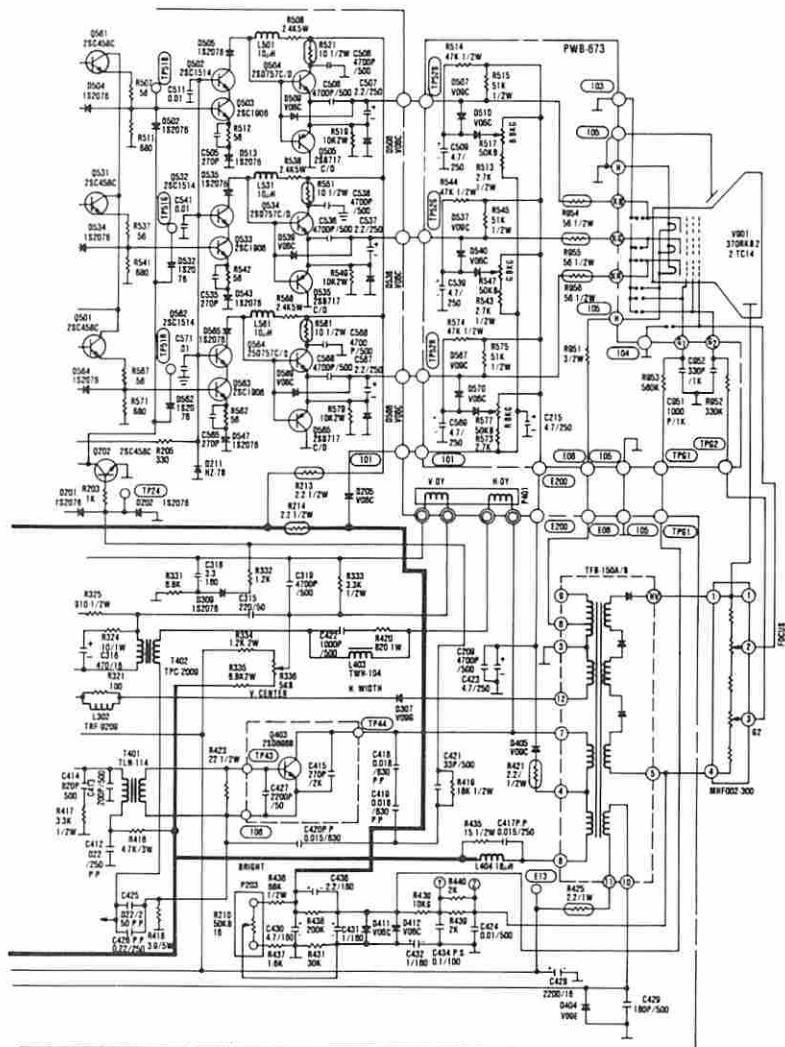
HAZARDOUS VOLTAGES
UP TO 450 VOLTS EXIST
ON THE PRINTED
CIRCUIT BOARDS



Color Display (Sheet 1 of 1)

**HAZARDOUS VOLTAGES
UP TO 450 VOLTS EXIST
ON THE PRINTED
CIRCUIT BOARDS**

1. RESISTOR VALUES ARE IN OHMS K = 1000 OHMS.
2. ALL RESISTORS ARE 1/2 WATT EXCEPT WHERE OTHERWISE INDICATED.
3. CAPACITOR VALUES ARE IN μ F UNLESS OTHERWISE INDICATED P = PF.
4. ALL CAPACITORS ARE 50 VOLTS UNLESS OTHERWISE INDICATED.

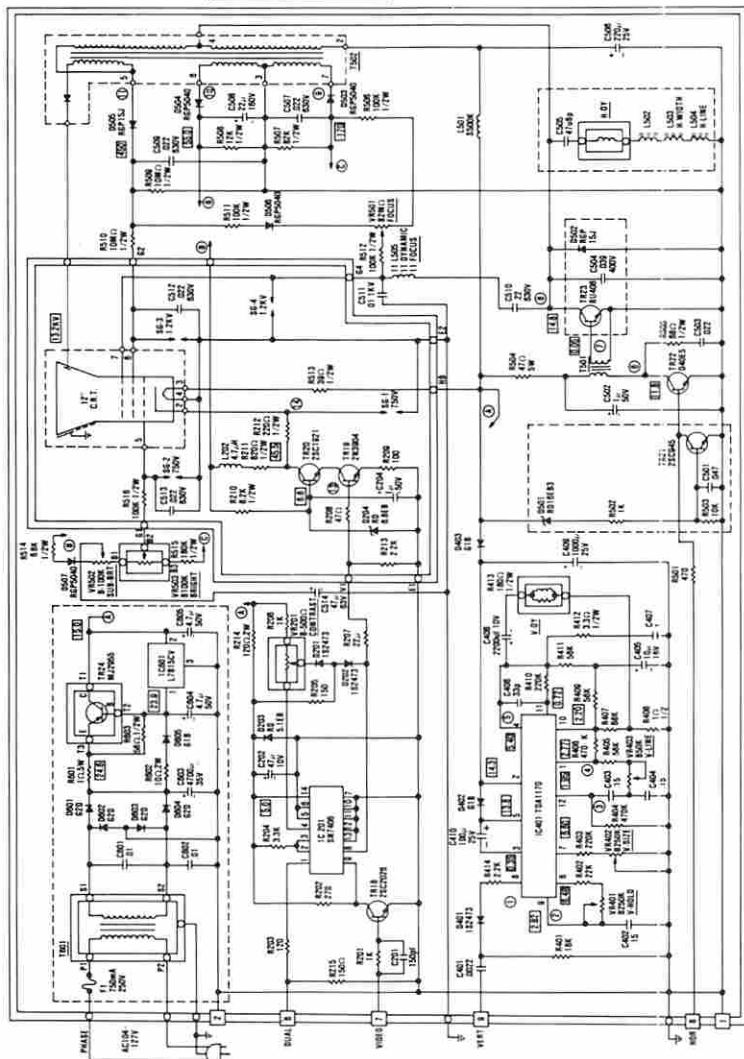


Color Display (Sheet 1 of 1)

DANGER

HAZARDOUS VOLTAGES
UP TO 450 VOLTS EXIST
ON THE PRINTED
CIRCUIT BOARDS

- NOTES
1. RESISTOR VALUES ARE IN OHMS (Ω), K (K), 1000Ω (K), 10000Ω (K).
 2. ALL RESISTORS ARE 1/4W UNLESS OTHERWISE INDICATED.
 3. CAPACITOR VALUES ARE IN P.F. UNLESS OTHERWISE INDICATED.
 4. CAPACITOR VALUES ARE IN P.F. UNLESS OTHERWISE INDICATED.
 5. AC WIRING INFORMATION
- PHASE 1: BLACK (GROUND WIRE)
PHASE 2: GREEN AND YELLOW WIRE
GROUND: GREEN AND YELLOW WIRE
- IMPORTANT: THE PHASE WIRE MUST GO TO THE FUSED SIDE OF TRANSFORMER

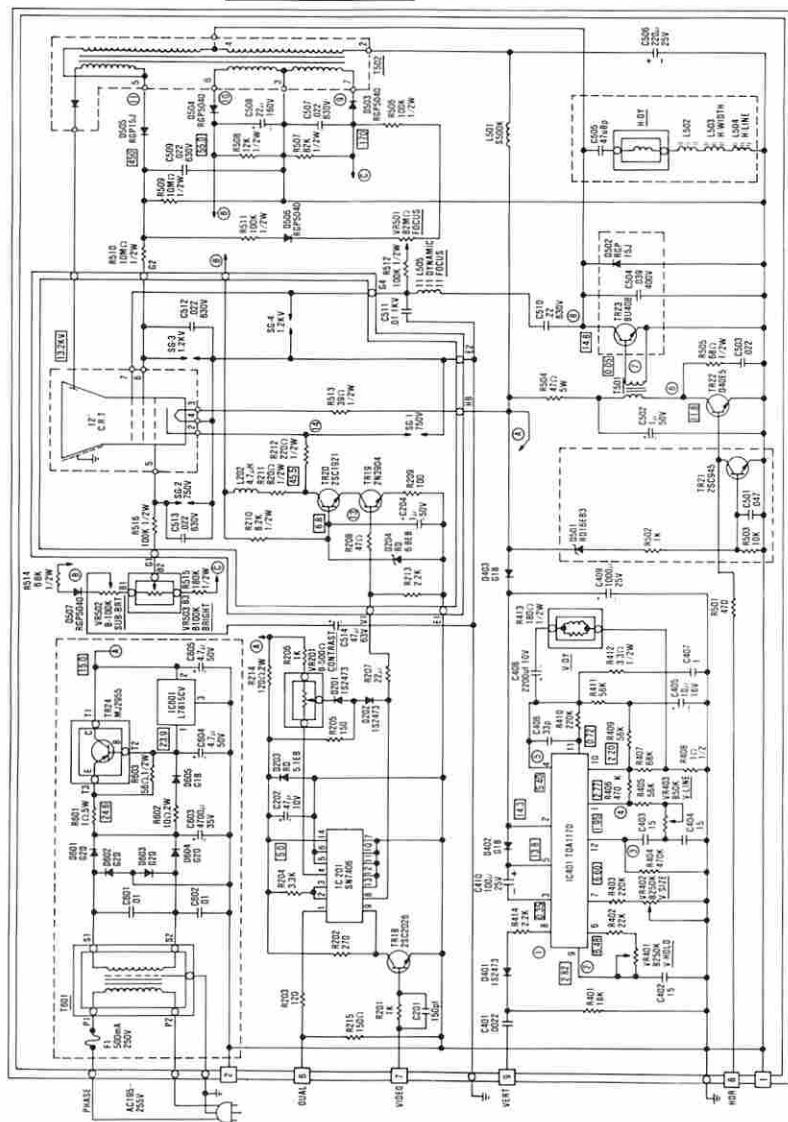


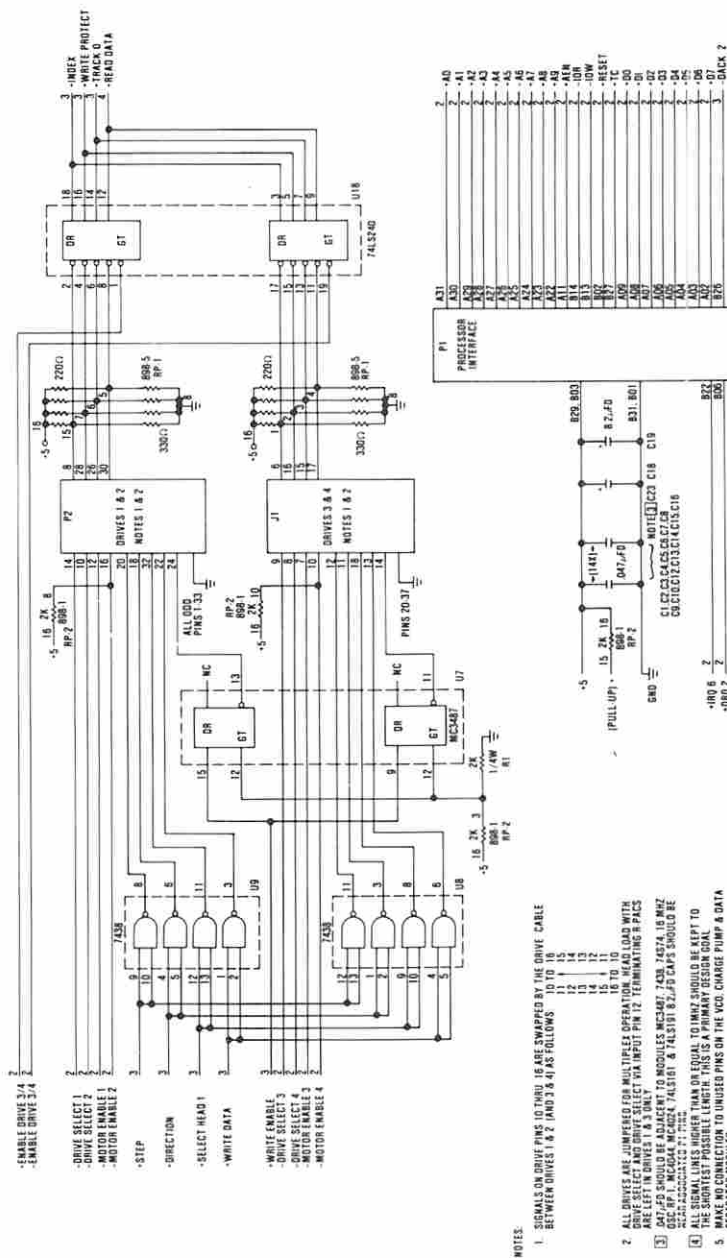
Monochrome Display 120 Vac (Sheet 1 of 1)

DANGER

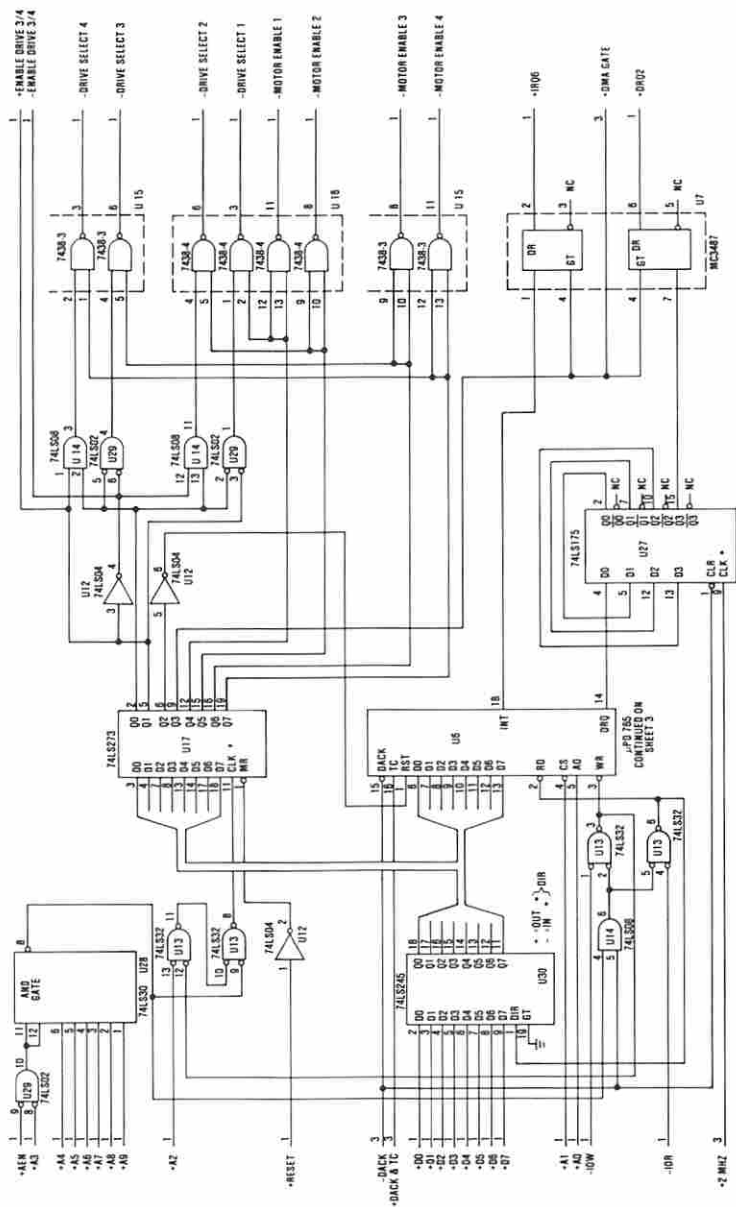
**HAZARDOUS VOLTAGES
UP TO 450 VOLTS EXIST
ON THE PRINTED
CIRCUIT BOARDS**

- NOTES
1. TEST TOLERANCE VALUES ARE IN POUNDS (L.B. = 1,000; M. = 1,000,000).
 2. ALL RESISTORS ARE 1/4W EXCEPT WHERE OTHERWISE INDICATED.
 3. ALL CAPACITORS ARE .50V EXCEPT WHERE OTHERWISE INDICATED.
 4. CAPACITOR VALUES ARE μ UNLESS OTHERWISE INDICATED.
 5. AC WAVEFORM INFORMATION.
 6. PHASE = BLACK/PURPLE WIRE
 7. NEUTRAL = WHITE/BLUE WIRE
 8. GROUND = GREEN AND YELLOW WIRE
- IMPORTANT: THE PHASE WIRE MUST GO TO THE FUSED SIDE OF TRANSFORMER

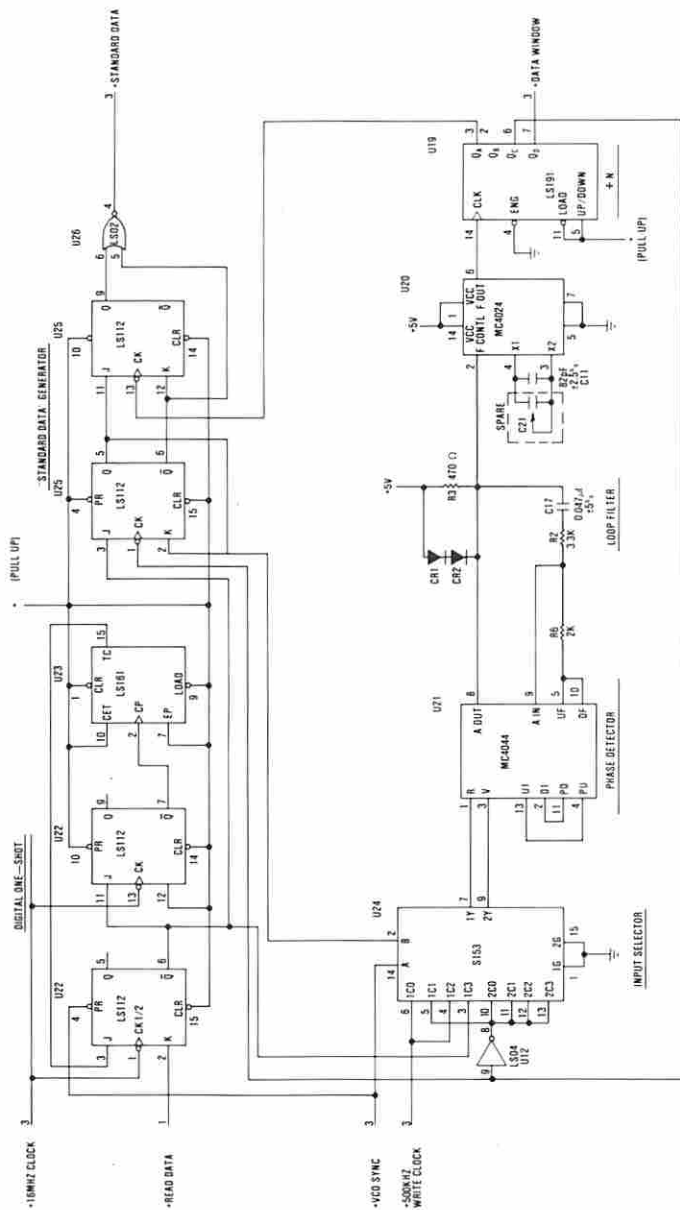




5-1/4 Inch Diskette Drive Adapter (Sheet 1 of 4)

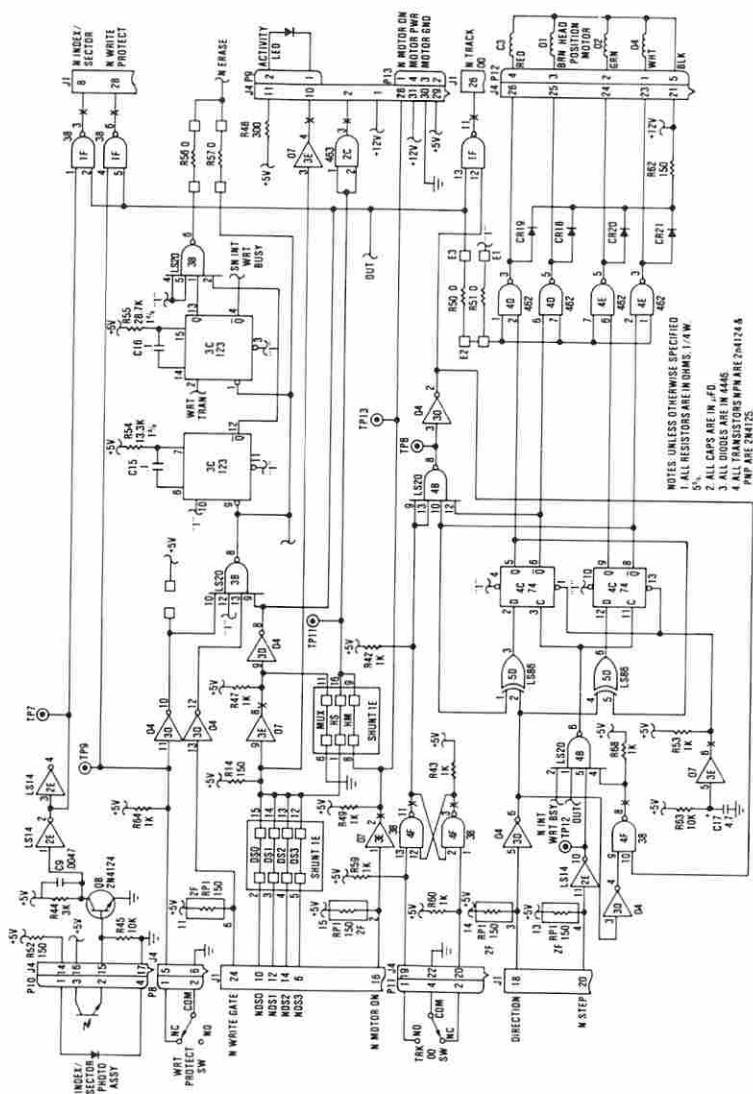


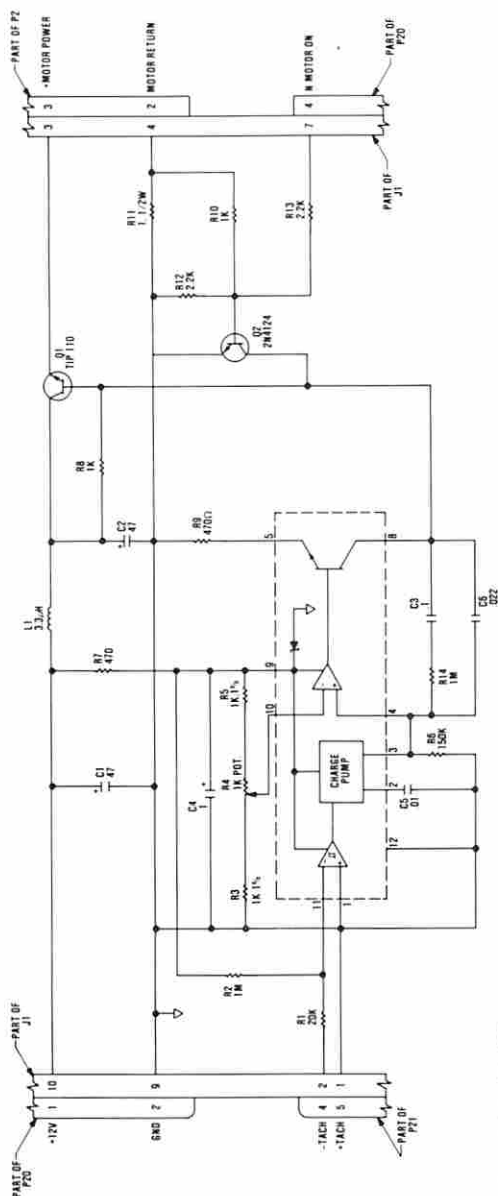
5-1/4 Inch Diskette Drive Adapter (Sheet 2 of 4)



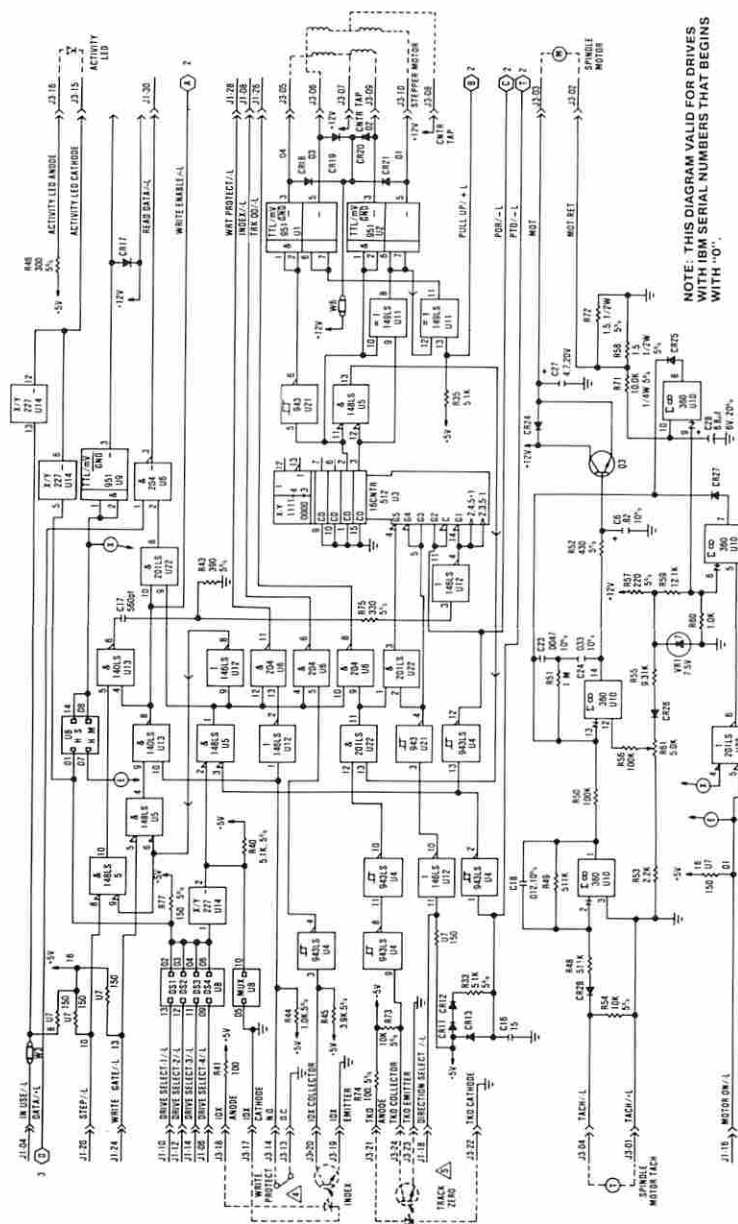
5-1/4 Inch Diskette Drive Adapter (Sheet 4 of 4)

5-1/4 Inch Diskette Drive Type 1 (Sheet 2 of 3)

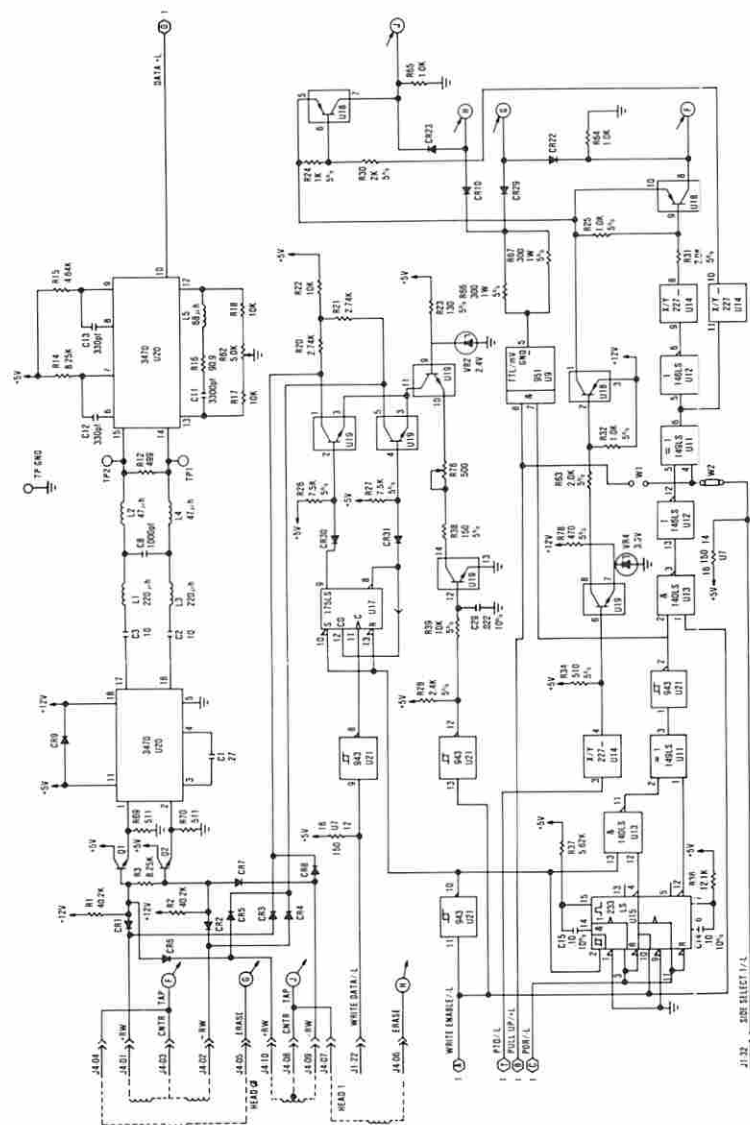




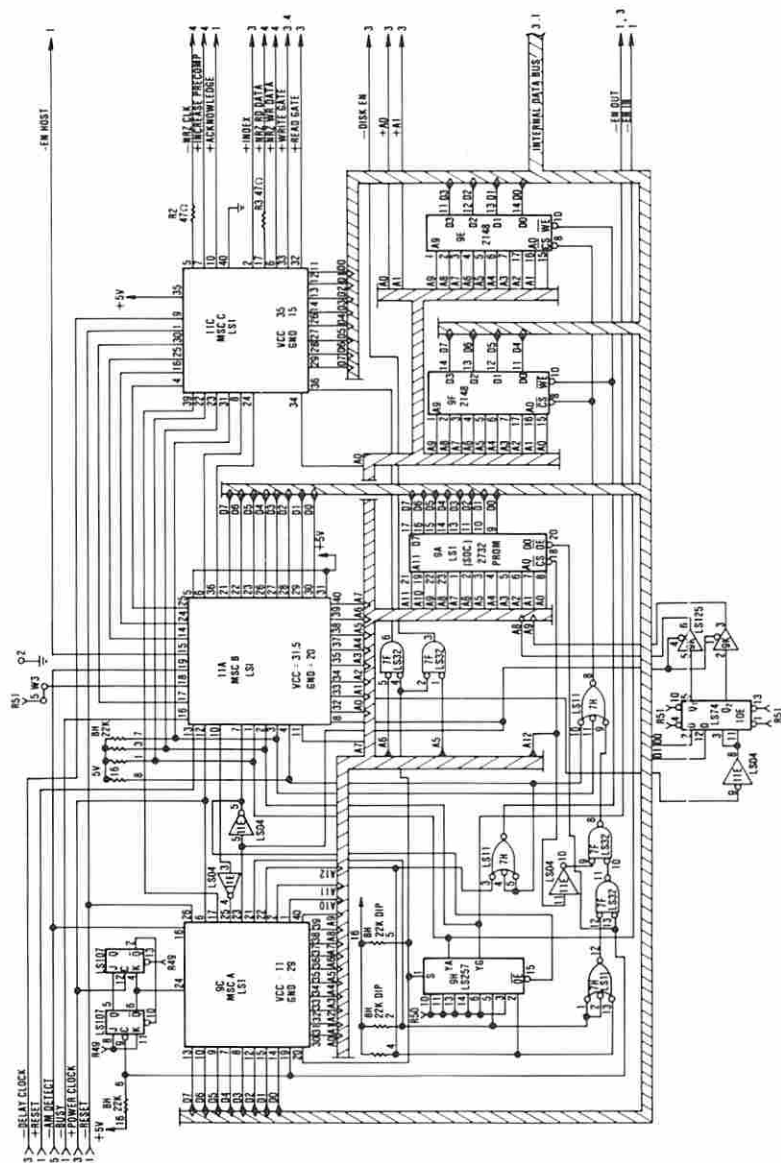
NOTES: UNLESS OTHERWISE SPECIFIED
 1. RESISTORS ARE 1/4W, 5% TOL.
 2. 1% RESISTORS ARE 1/8W.
 3. CAPACITORS ARE IN μF , $\pm 10\%$ TOL.



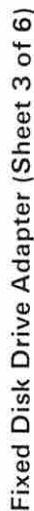
5-1/4 Inch Diskette Drive Type 2 (Sheet 1 of 2)

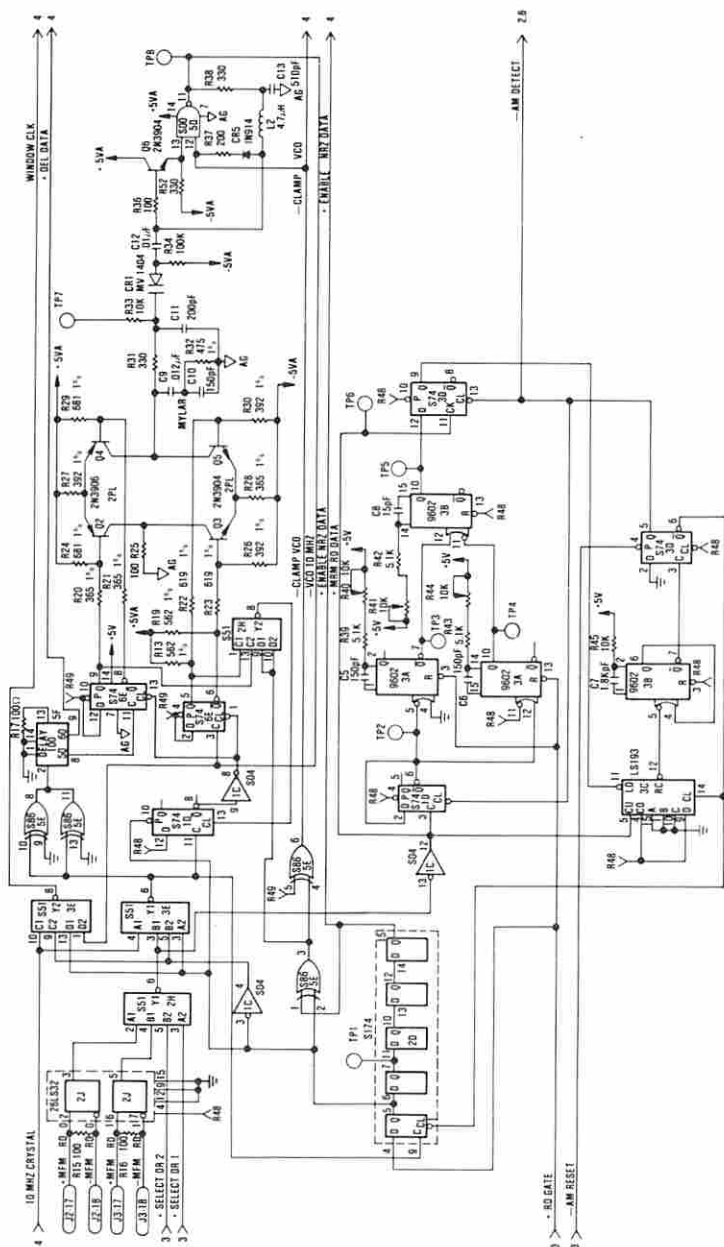


5-1/4 Inch Diskette Drive Type 2 (Sheet 2 of 2)

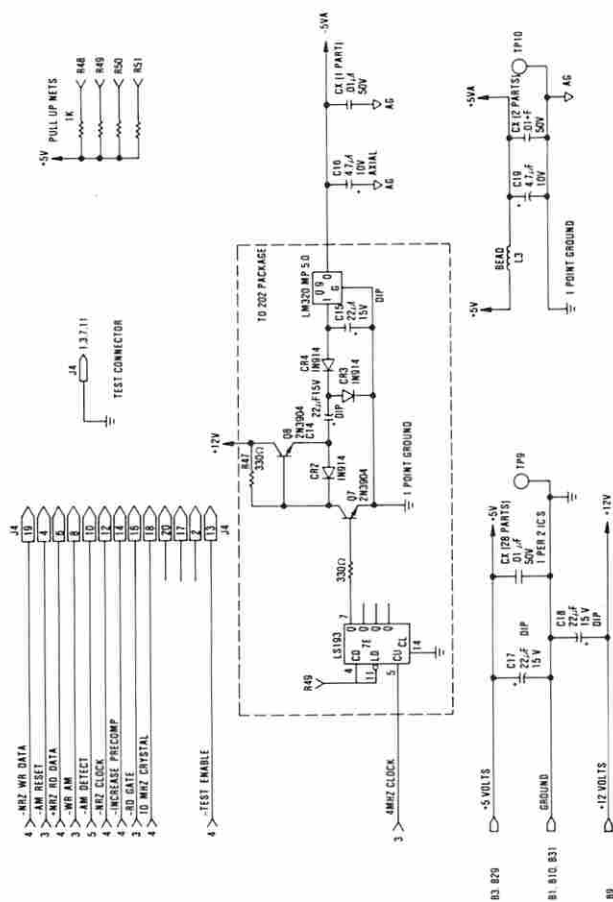


Fixed Disk Drive Adapter (Sheet 2 of 6)



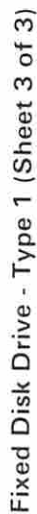


Fixed Disk Drive Adapter (Sheet 5 of 6)



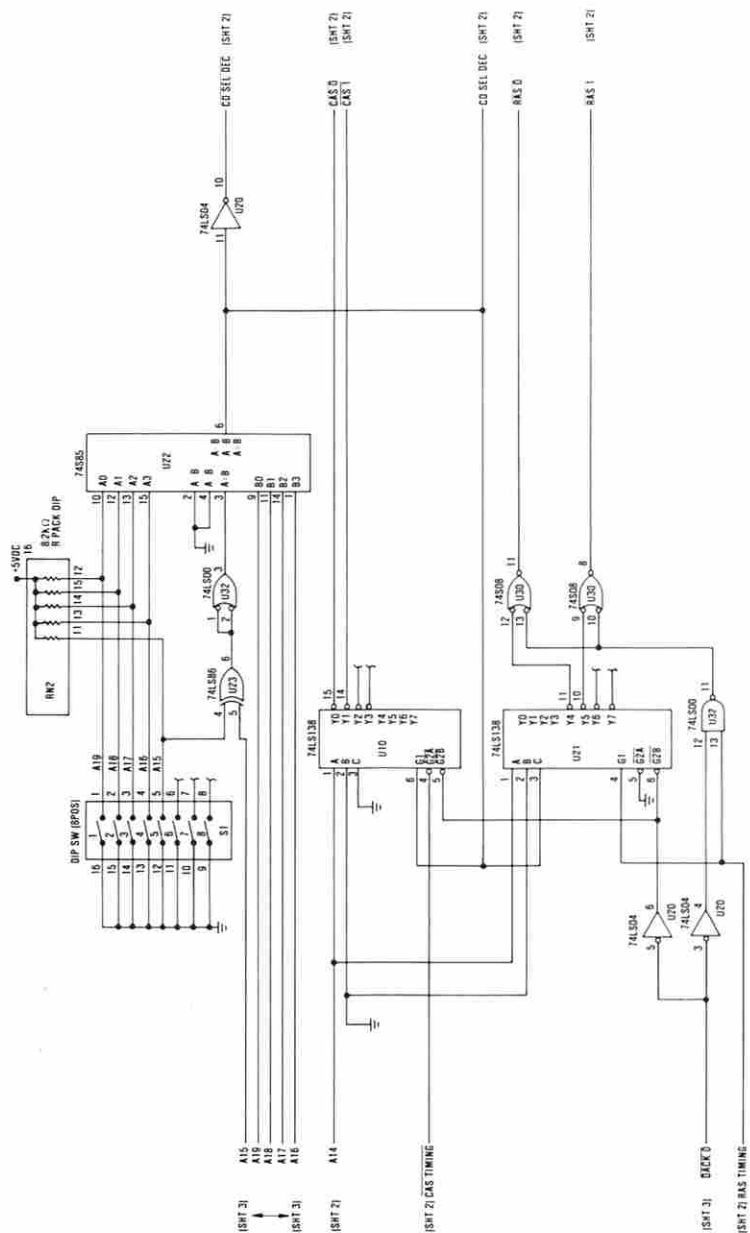
NOTES:
UNLESS OTHERWISE SPECIFIED
1 ALL RESISTORS 1/4 W, 5% CARBON FILTER
2 ALL CAPS - 10V OR GREATER - 10%
3 NO MORE THAN 15 LOADS PER PULLUP NET



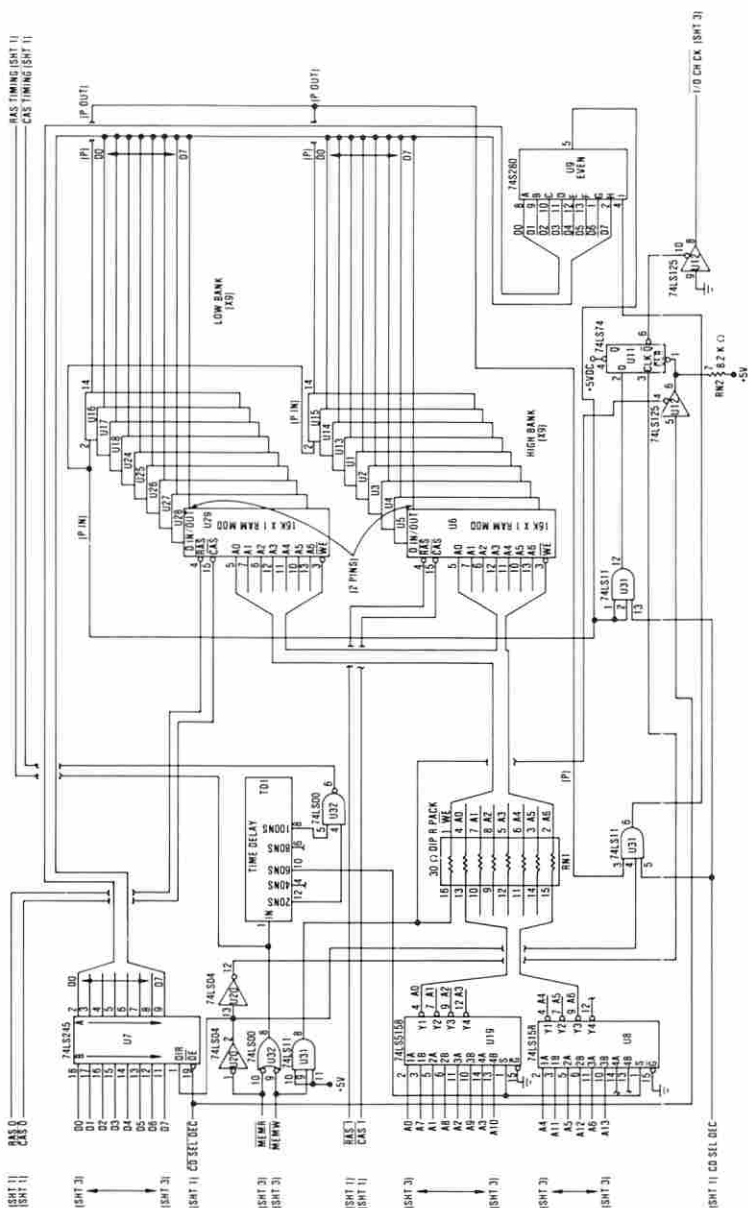




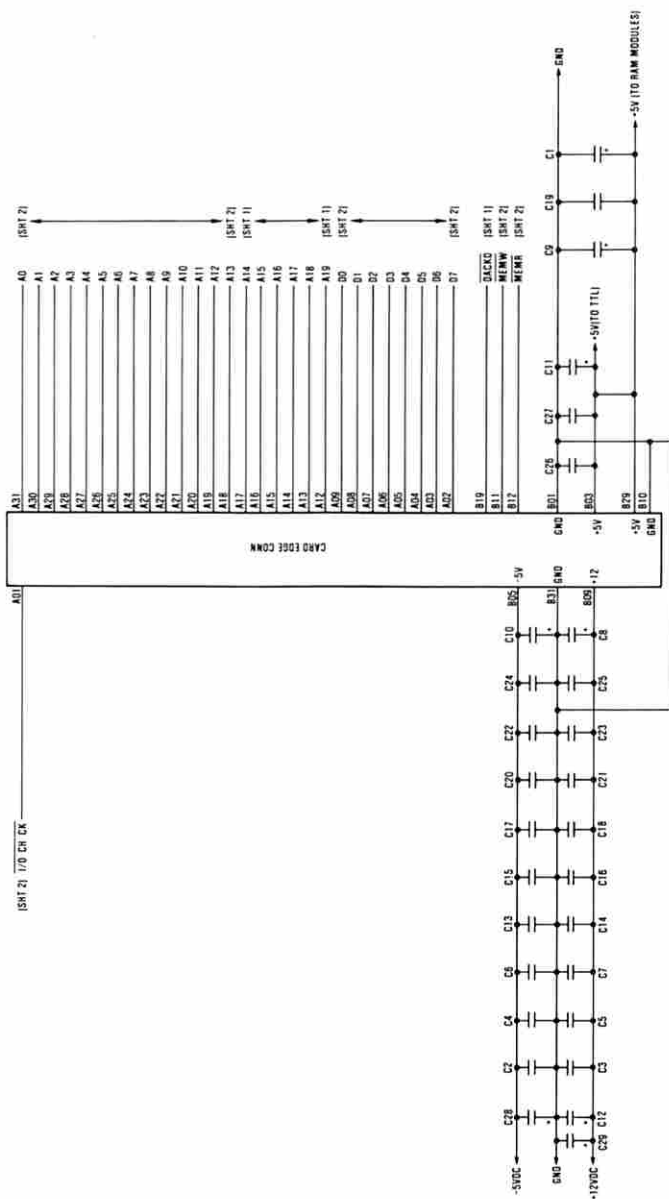




32K Memory Expansion Option (Sheet 1 of 3)

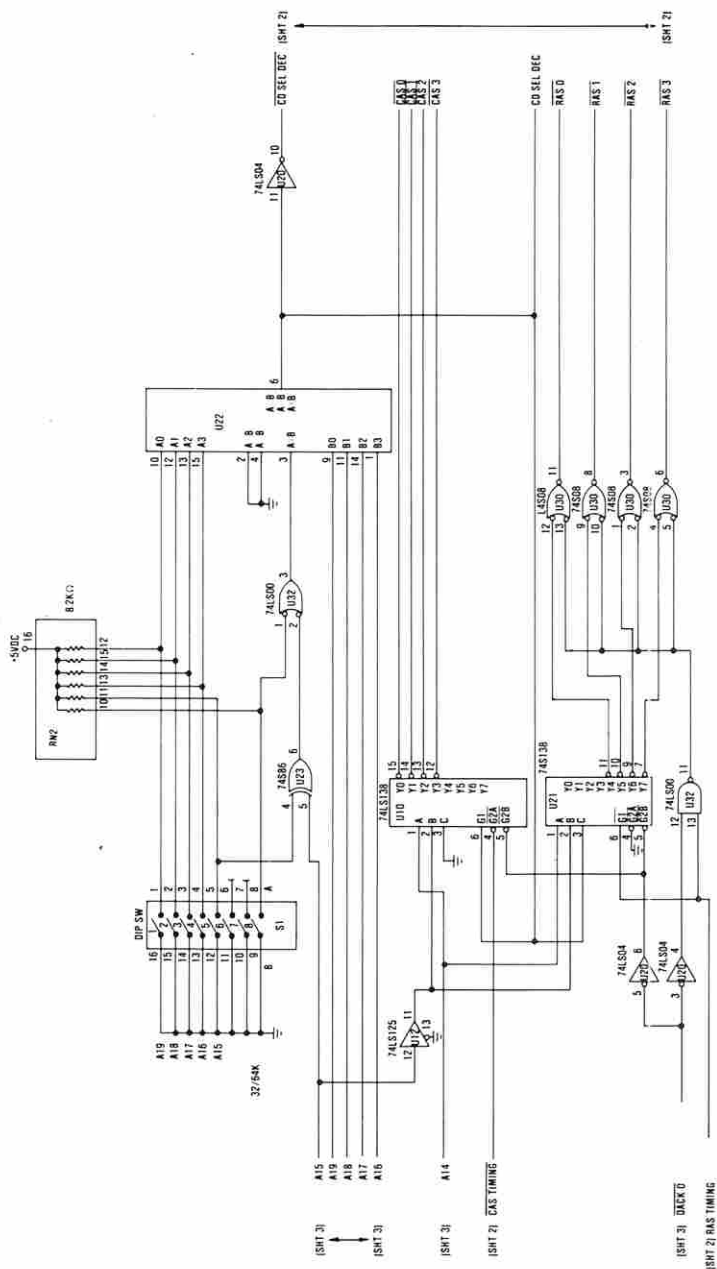


32K Memory Expansion Option (Sheet 2 of 3)

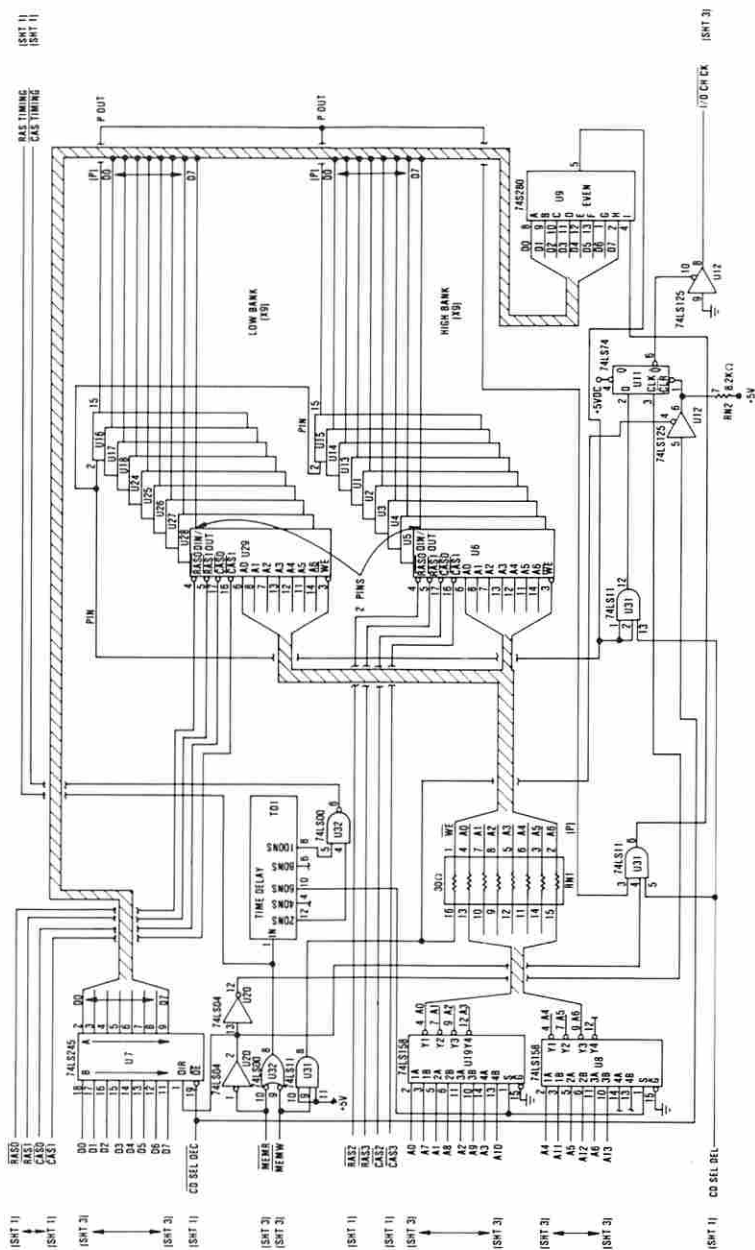


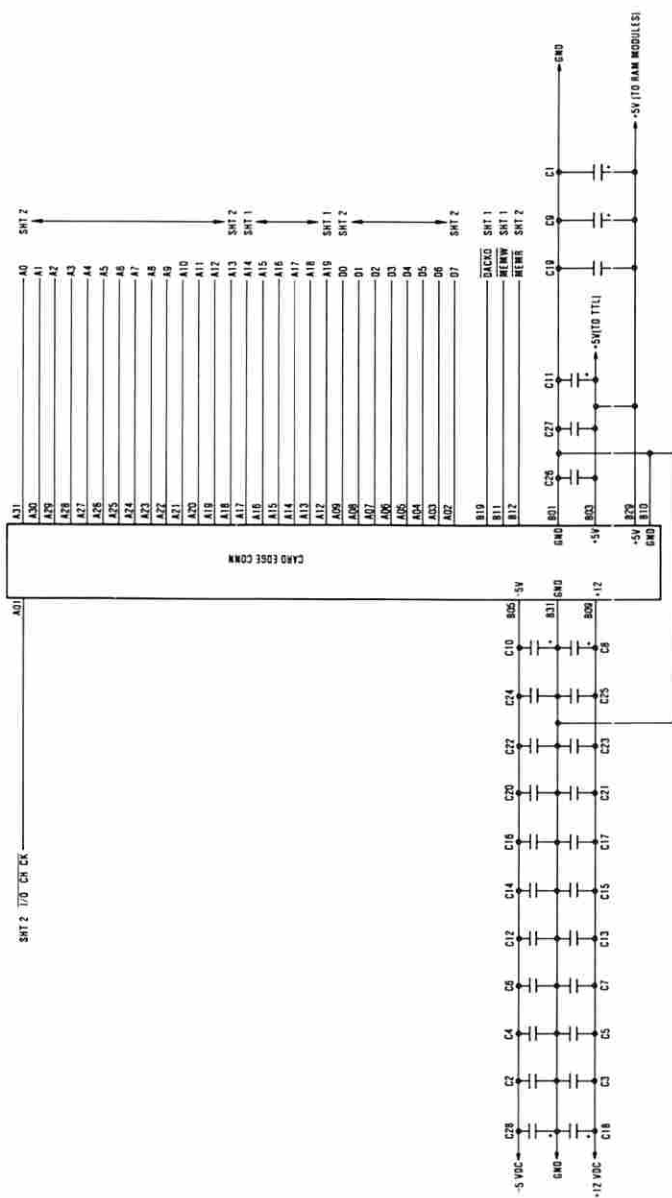
32K Memory Expansion Option (Sheet 3 of 3)

D-70 Logic Diagrams

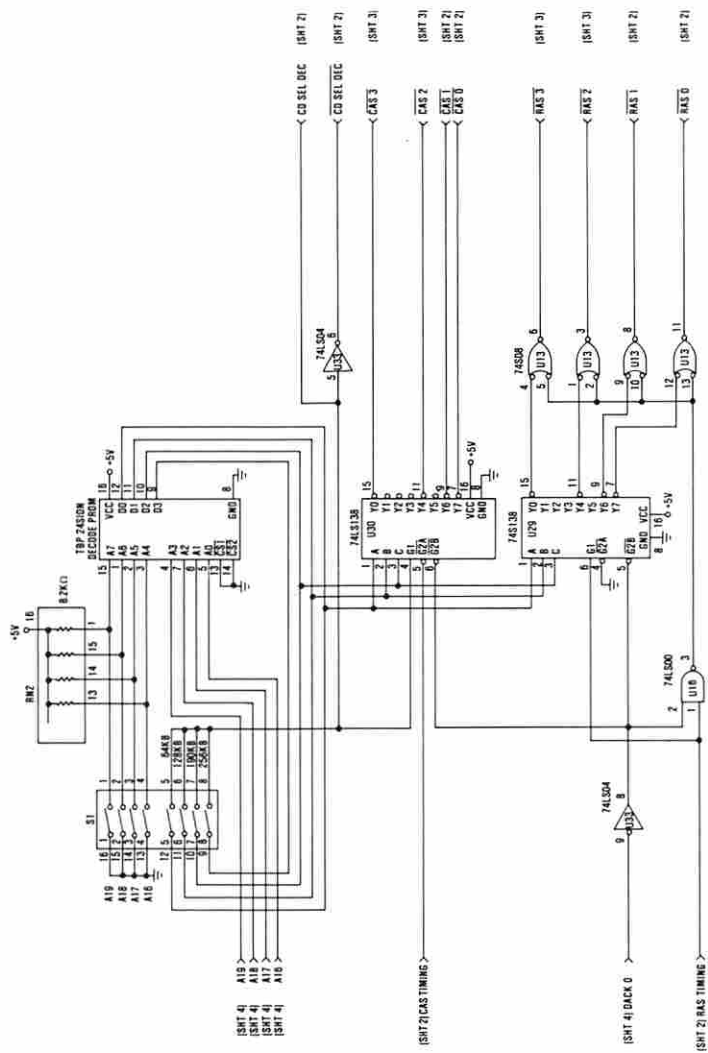


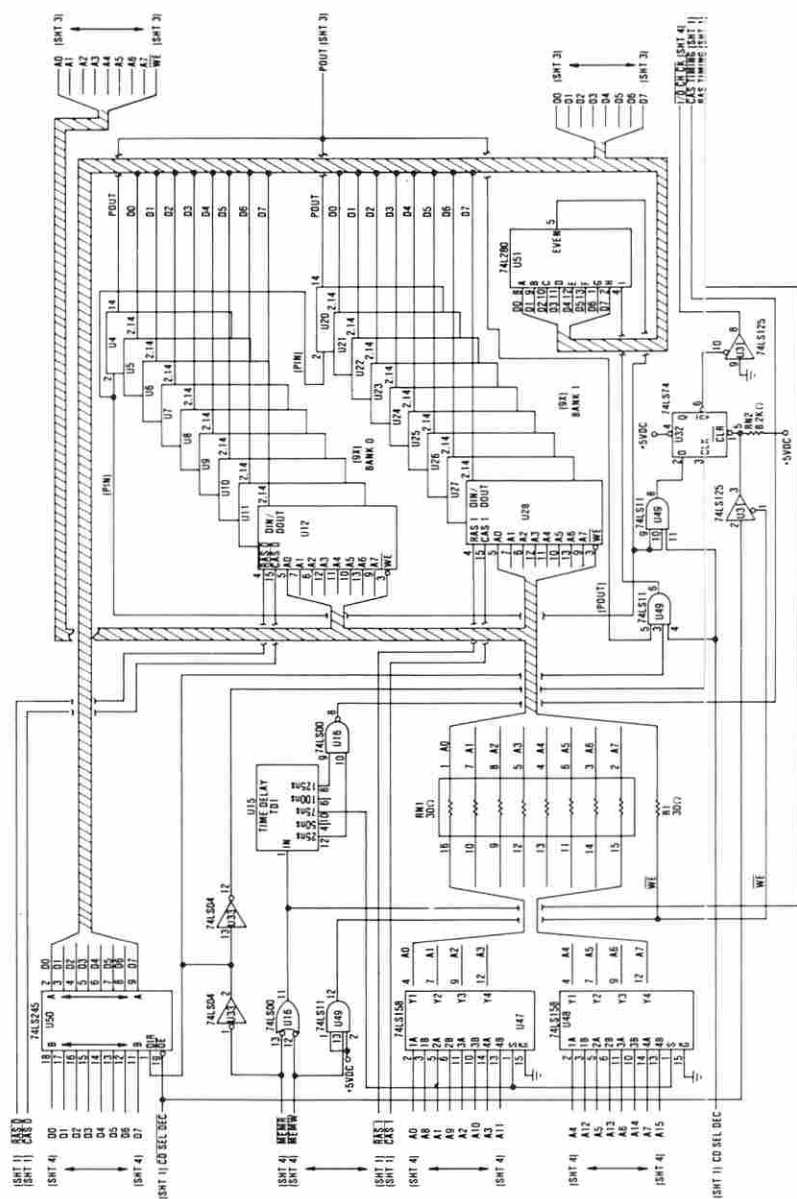
64K Memory Expansion Option (Sheet 1 of 3)



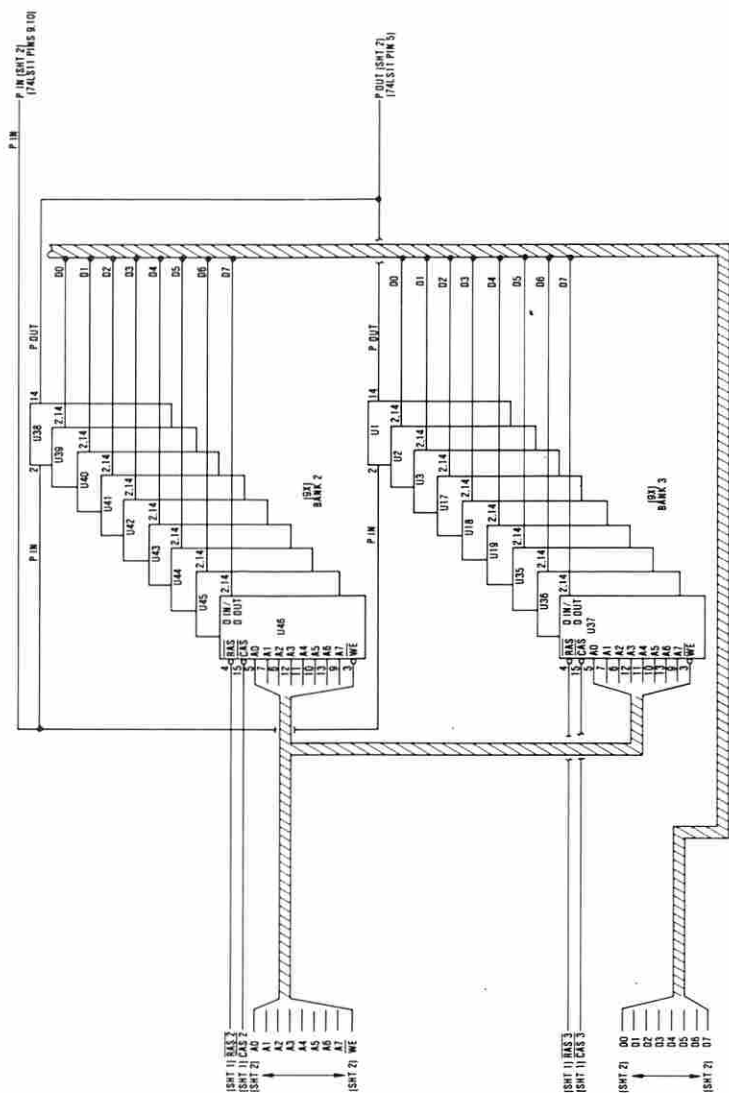


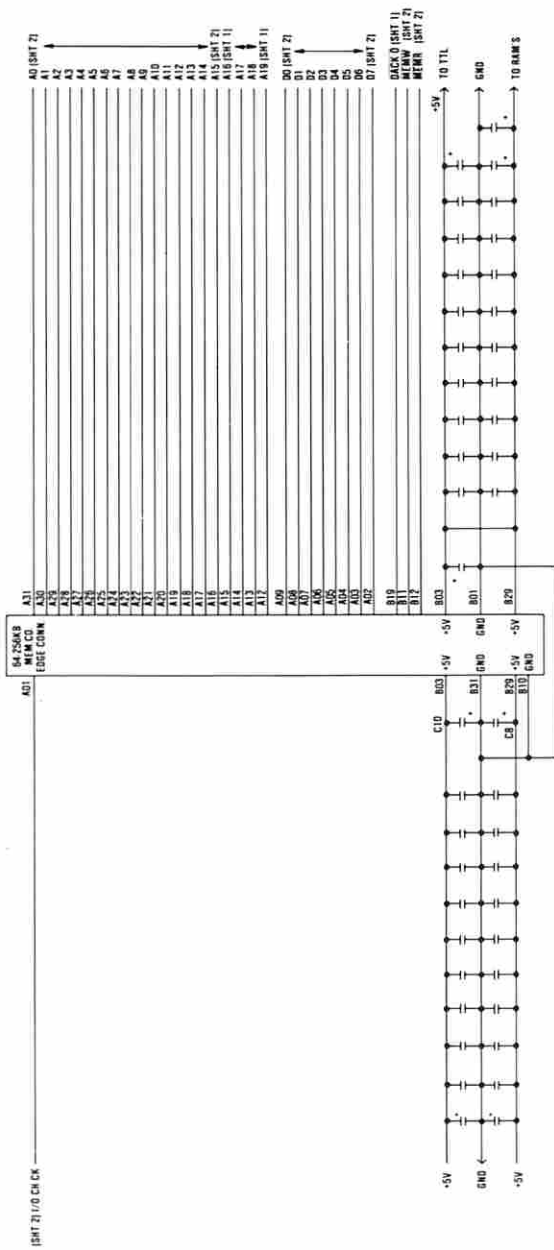
64K Memory Expansion Option (Sheet 3 of 3)

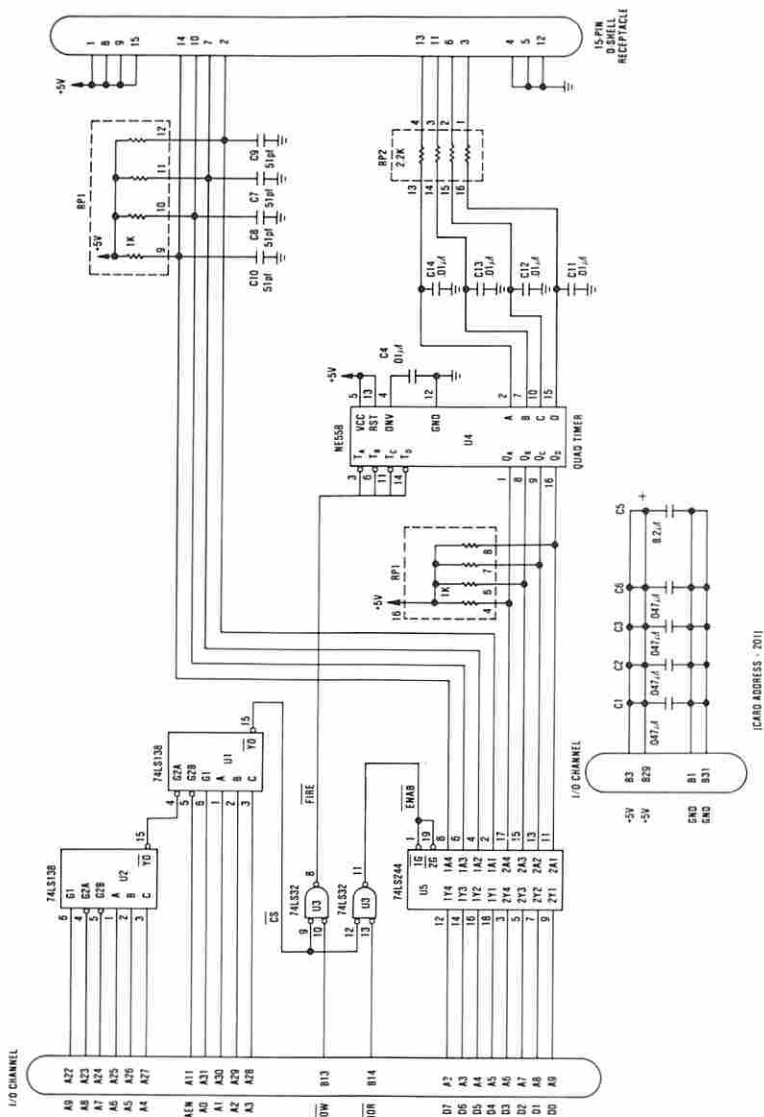




64 / 256K Memory Expansion Option (Sheet 2 of 4)

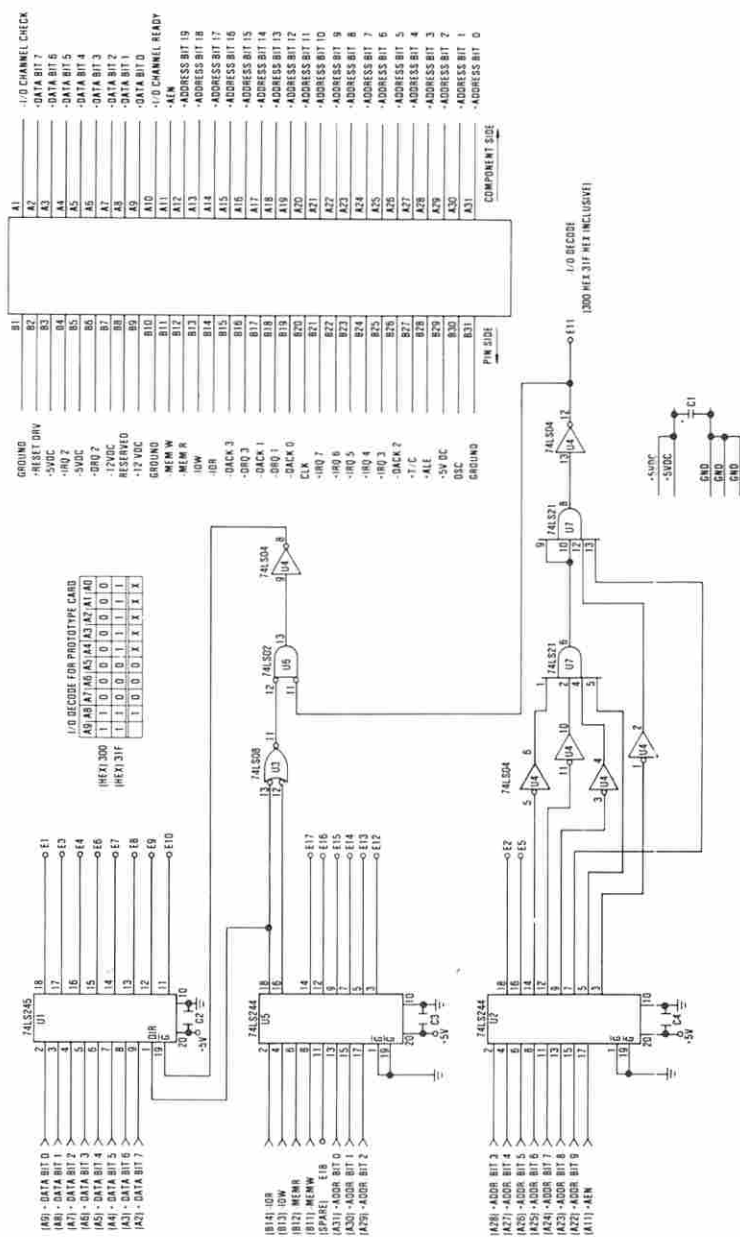




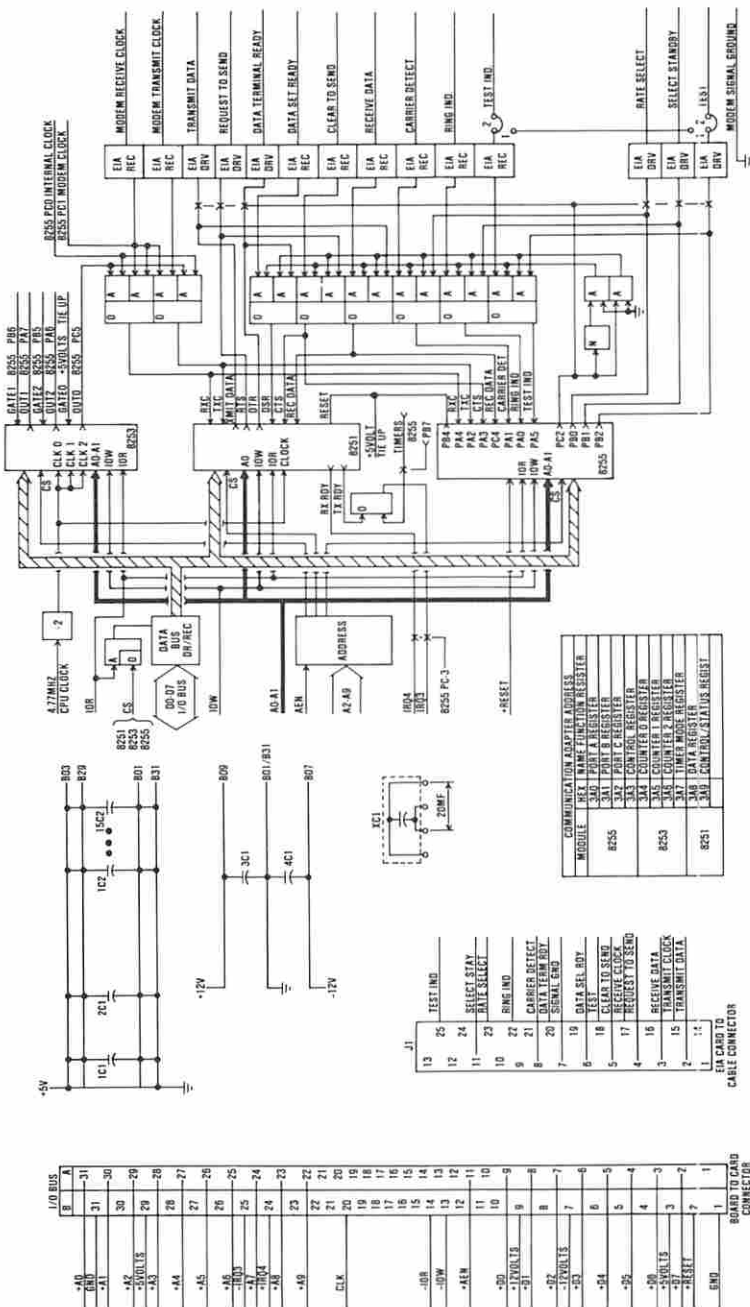


Game Control Adapter (Sheet 1 of 1)

D-78 Logic Diagrams

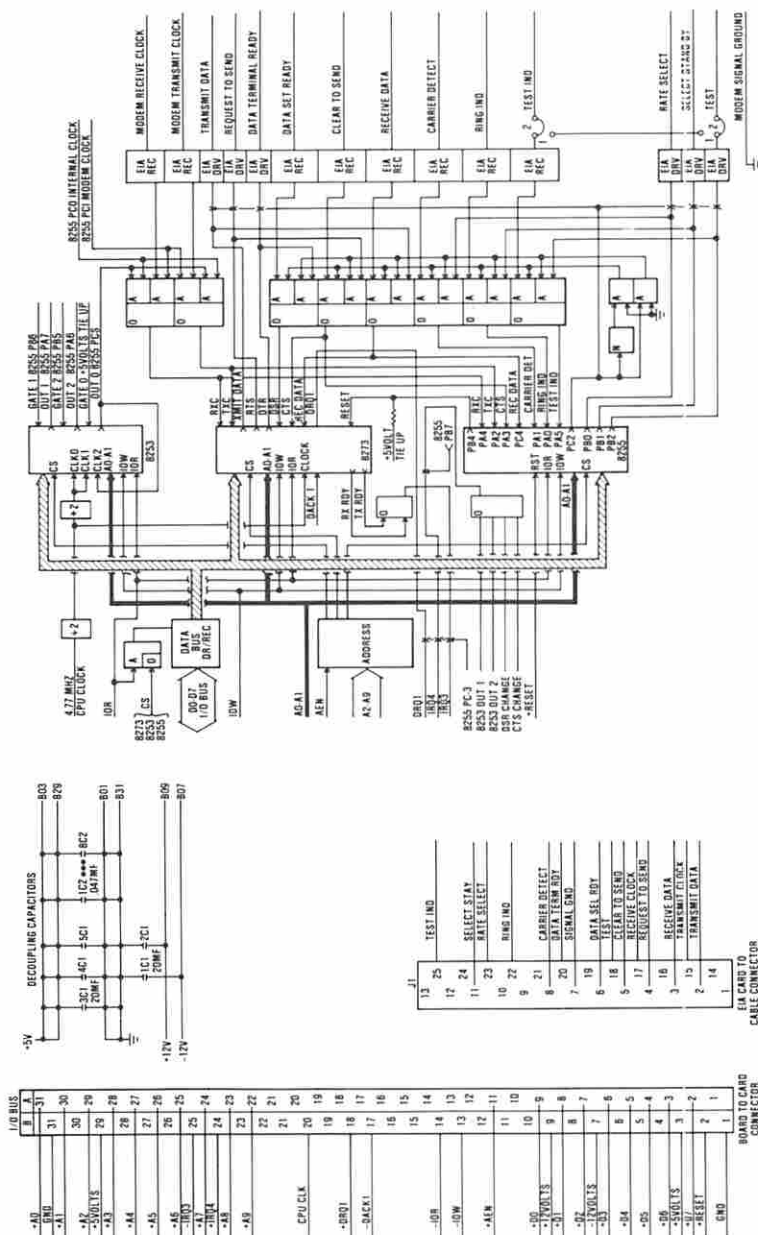


Prototype Card (Sheet 1 of 1)



Binary Synchronous Communications Adapter (Sheet 1 of 2)





APPENDIX E: SPECIFICATIONS

System Unit

Size:

Length--19.6 in (500 mm)

Depth--16.1 in (410 mm)

Height--5.5 in (142 mm)

Weight:

20.9 lb (9.5 kb) Without a diskette drive unit

25.0 lb (11.4 kg) With one diskette drive unit

Power Cable:

Length--6 ft (1.83 m)

Size--18 AWG

Environment:

Air Temperature

System ON, 60° to 90° F (15.6° to 32.2° C)

System OFF, 50° to 110° F (10° to 43° C)

Humidity

System ON, 8% to 80%

System OFF, 20% to 80%

Heat Output:

1083 BTU/hr

Noise Level:

56 dB Without printer

66 dB With printer

Electrical:

Nominal-120 Vac 220/240 Vac

Minimum-104 Vac 180 Vac

Maximum-127 Vac 259 Vac

kVa--0.3175 (maximum) KVA-.3175 (maximum)

Keyboard

Size:

Length--19.6 in (500 mm)

Depth--7.87 in (200 mm)

Height--2.2 in (57 mm)

Weight:

6.5 lb (2.9 kg)

Color Display

Size:

Length--15.4 in (392 mm)

Depth--15.6 in (407 mm)

Height--11.7 in (297 mm)

Weight:

26 lb (11.8 kg)

Heat Output:

240 BTU/hr

Power Cable:

Length--6 ft (1.83 m)

Size--18 AWG

Signal Cable:

Length--5 ft (1.5 m)

Size--22 AWG

Expansion Unit

Size:

Length--19.6 in (500 mm)

Depth--16.1 in (410 mm)

Height--5.5 in (142 mm)

Weight:

33 lb (14.9 kg)

Power Cable:

Length--6 ft (1.83 m)

Size--18 AWG

Signal Cable:

Length--3.28 ft (1 m)

Size--22 AWG

Environment:

Air Temperature

System ON, 60° to 90° F (15.6° to 32.2° C)

System OFF, 50° to 110° F (10° to 43° C)

Humidity

System ON, 8% to 80%

System OFF, 20% to 80%

Heat Output:

717 BTU/hr

Electrical:

Nominal-120 Vac 220/240 Vac

Minimum-104 Vac 180 Vac

Maximum-127 Vac 259 Vac

Monochrome Display

Size:

Length--14.9 in (380 mm)

Depth--13.7 in (350 mm)

Height--11 in (280 mm)

Weight:

17.3 lb (7.9 kg)

Heat Output:

325 BTU/hr

Power Cable:

Length--3 ft (0.914 m)

Size--18 AWG

Signal Cable:

Length--4 ft (1.22 m)

Size--22 AWG

80 CPS Printer

Size:

Length--15.7 in (400 mm)

Depth--14.5 in (370 mm)

Height--4.3 in (110 mm)

Weight:

12.9 lb (5.9 kg)

Power Cable:

Length--6 ft (1.83 m)

Size--18 AWG

Signal Cable:

Length--6 ft (1.83 mm)

Size--22 AWG

Heat Output:

341 BTU/hr (maximum)

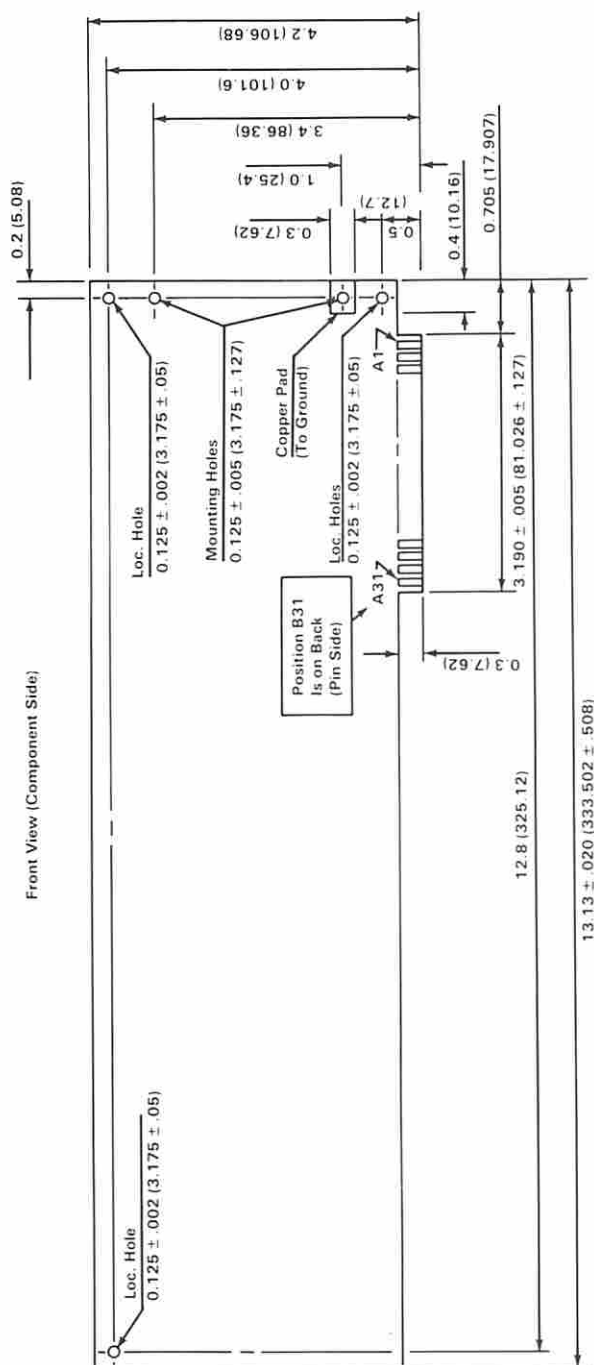
Electrical: (three models available)

Nominal--120 Vac	220 Vac	240 Vac
------------------	---------	---------

Minimum--104 Vac	198 Vac	216 Vac
------------------	---------	---------

Maximum--127 Vac	242 Vac	264 Vac
------------------	---------	---------

Card Specifications



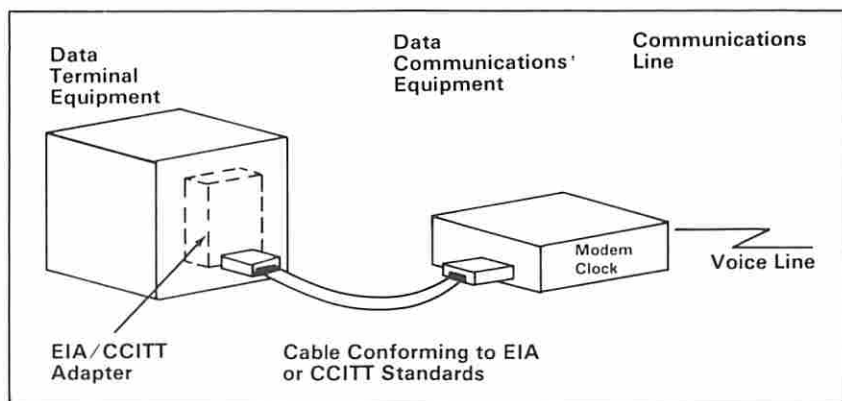
Notes:

1. All Card Dimensions are $\pm .010$ (.254) Tolerance (With Exceptions Indicated on Drawing or in Notes).
2. Max. Card Length is 13.15 (334.01). Smaller Length is Permissible.
3. Loc. and Mounting Holes are Non-Plated Thru. (Loc. 3X, Mtg. 2X).
4. 31 Gold Tabs Each Side. $0.100 \pm .0005$ (2.54 $\pm$.0127) Center to Center, $0.06 \pm .0005$ (1.524 $\pm$.0127) Width.
5. Numbers in Parentheses are in Millimeters. All Others are in Inches.

APPENDIX F: COMMUNICATIONS

Information processing equipment used for communications is called data terminal equipment (DTE). Equipment used to connect the DTE to the communications line is called data communications equipment (DCE).

An adapter is used to connect the data terminal equipment to the data communications line as shown in the following illustration:



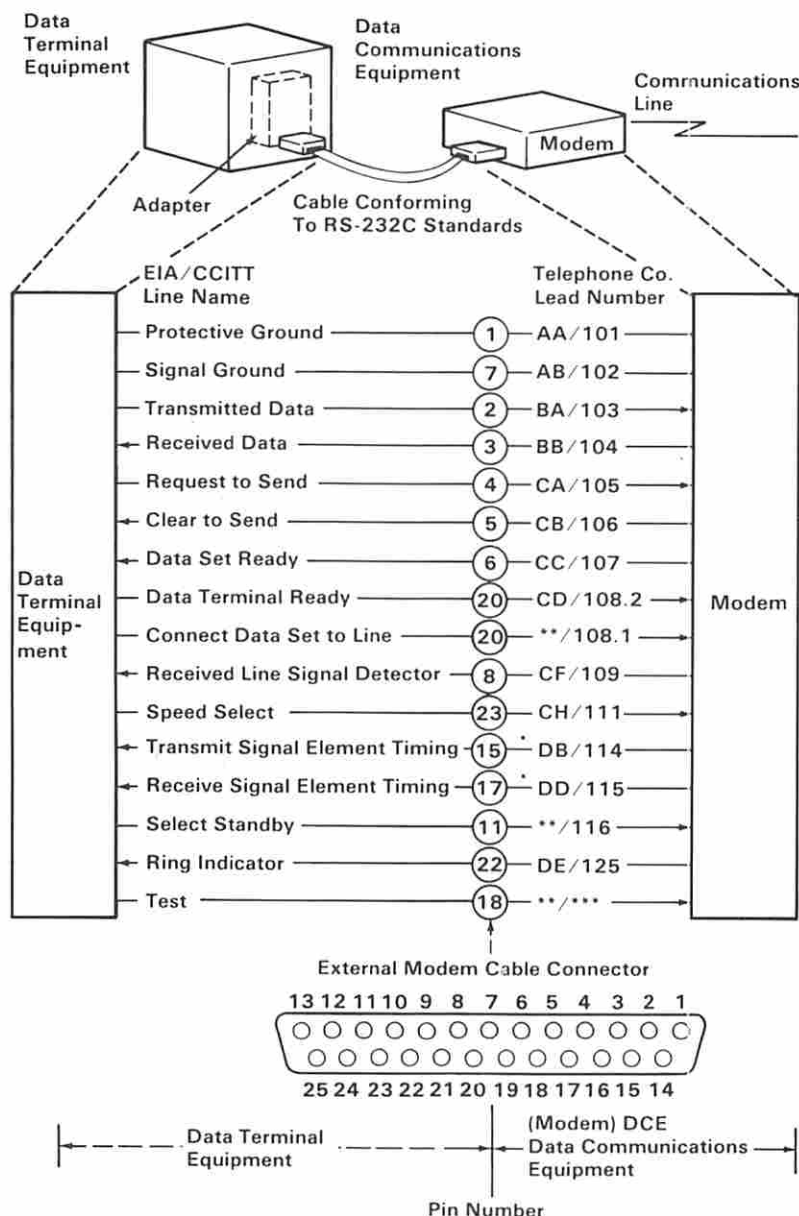
The EIA/CCITT adapter allows data terminal equipment to be connected to data communications equipment using EIA or CCITT standardized connections. An external modem is shown in this example; however, other types of data communications equipment can also be connected to data terminal equipment using EIA or CCITT standardized connections.

EIA standards are labeled RS-x (Recommended Standards-x) and CCITT standards are labeled V.x or X.x, where x is the number of the standard.

The EIA RS-232 interface standard defines the connector type, pin numbers, line names, and signal levels used to connect data terminal equipment to data communications equipment for the purpose of transmitting and receiving data. Since the RS-232 standard was developed, it has been revised three times. The three revised standards are the RS-232A, the RS-232B, and the presently used RS-232C.

The CCITT V.24 interface standard is equivalent to the RS-232C standard; therefore, the descriptions of the EIA standards also apply to the CCITT standards.

The following is an illustration of data terminal equipment connected to an external modem using connections defined by the RS-232C interface standard:



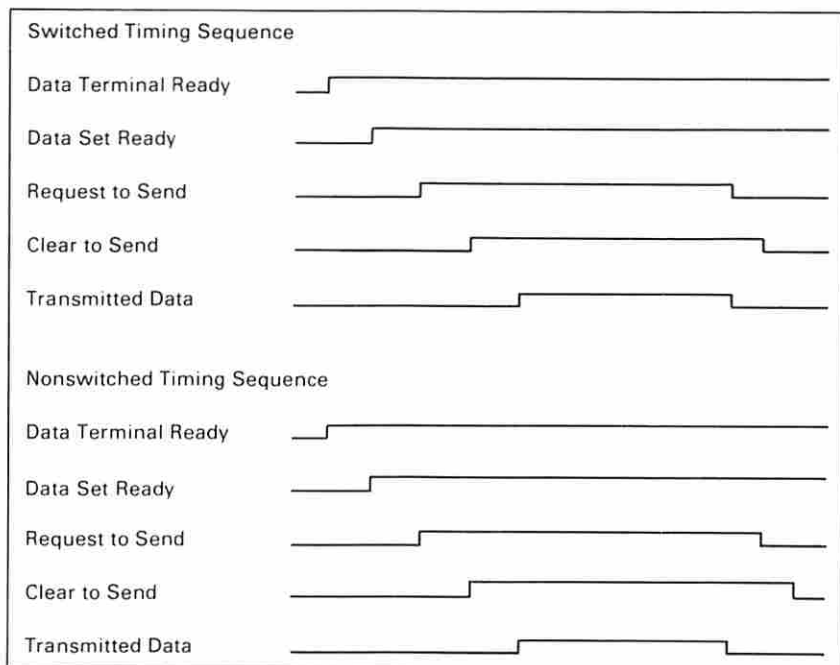
*Not used when business machine clocking is used.

**Not standardized by EIA (Electronics Industry Association).

***Not standardized by CCITT

Establishing a Communications Link

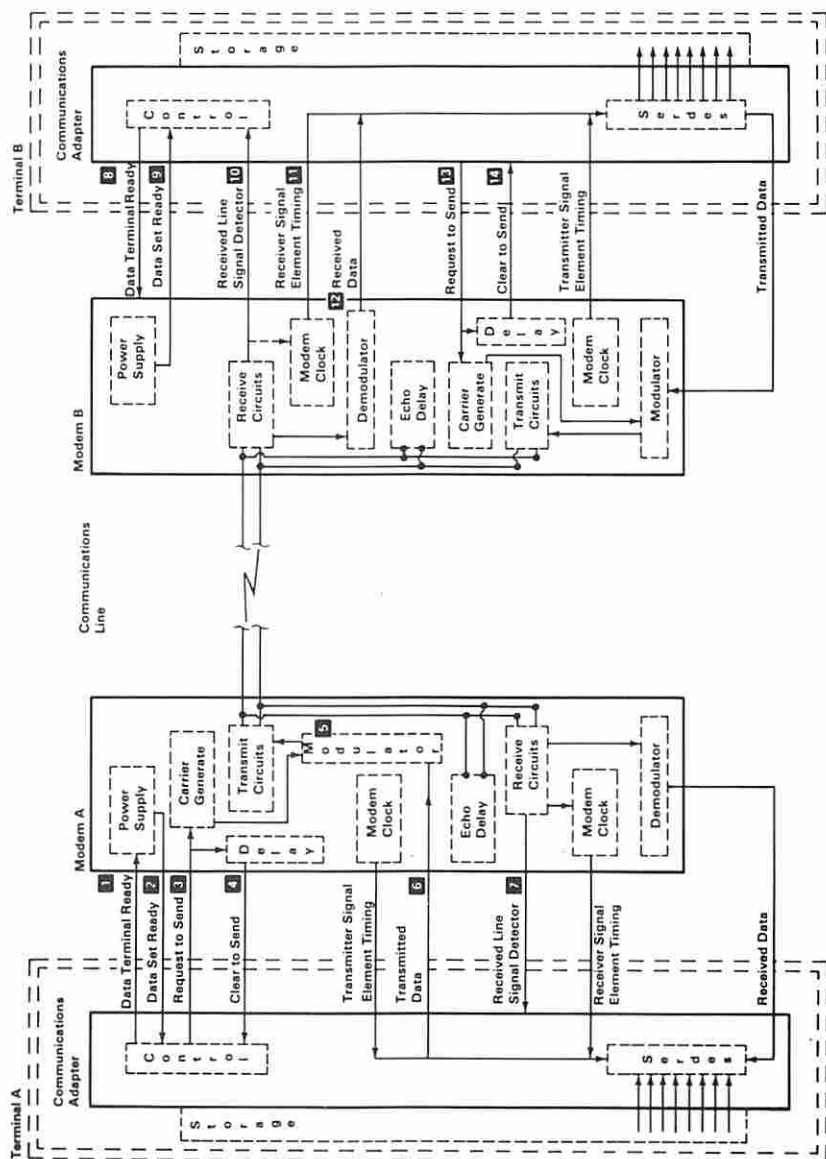
The following bar graphs represent normal timing sequences of operation during the establishment of communications for both switched (dial-up) and nonswitched (direct line) networks.



The following examples show how a link is established on a nonswitched point-to-point line, a nonswitched multipoint line, and a switched point-to-point line.

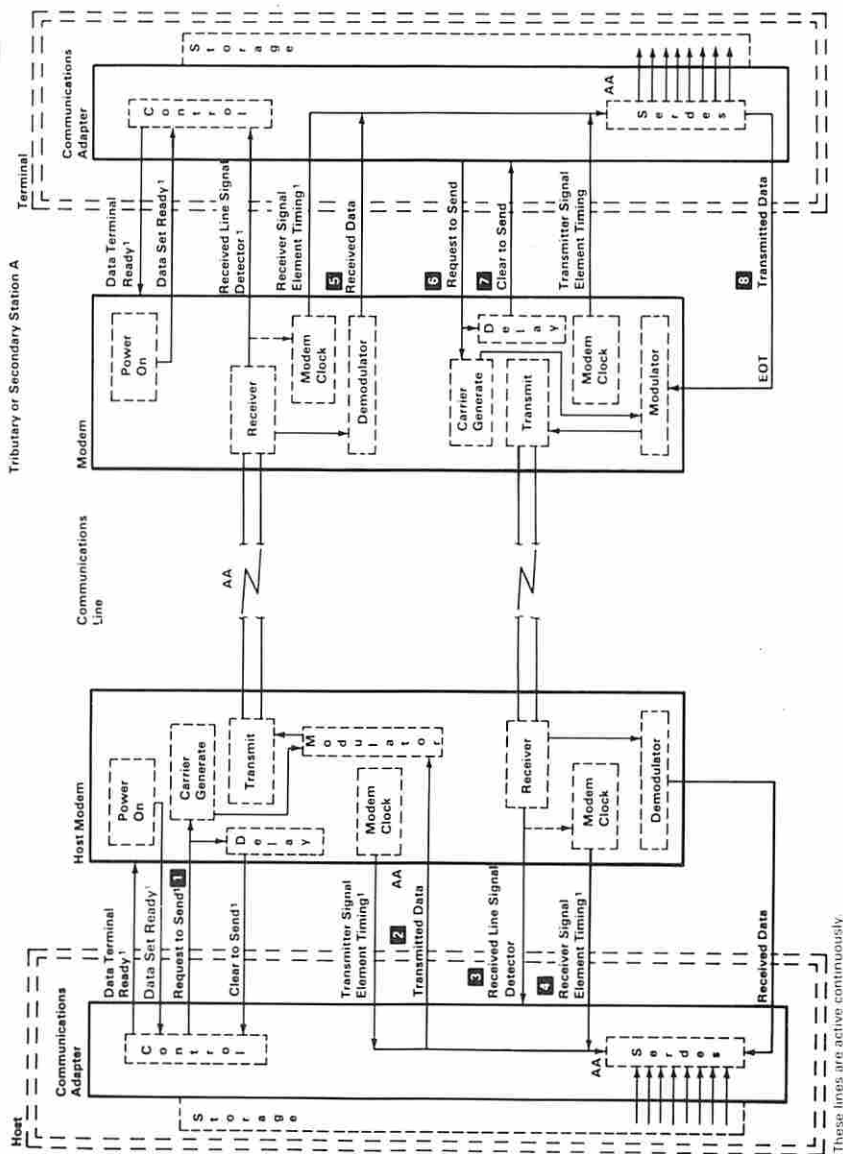
Establishing a Link on a Nonswitched Point-to-Point Line

1. The terminals at both locations activate the 'data terminal ready' lines **1** and **8**.
2. Normally the 'data set ready' lines **2** and **9** from the modems are active whenever the modems are powered on.
3. Terminal A activates the 'request to send' line **3**, which causes the modem at terminal A to generate a carrier signal.
4. Modem B detects the carrier, and activates the 'received line signal detector' line (sometimes called data carrier detect) **10**. Modem B also activates the 'receiver signal element timing' line (sometimes called receive clock) **11** to send receive clock signals to the terminal. Some modems activate the clock signals whenever the modem is powered on.
5. After a specified delay, modem A activates the 'clear to send' line **4**, which indicates to terminal A that the modem is ready to transmit data.
6. Terminal A serializes the data to be transmitted (through the serdes) and transmits the data one bit at a time (synchronized by the transmit clock) onto the 'transmitted data' line **6** to the modem.
7. The modem modulates the carrier signal with the data and transmits it to the modem B **5**.
8. Modem B demodulates the data from the carrier signal and sends it to terminal B on the 'received data' line **12**.
9. Terminal B deserializes the data (through the serdes) using the receive clock signals (on the 'receiver signal element timing' line) **11** from the modem.
10. After terminal A completes its transmission, it deactivates the 'request to send' line **3**, which causes the modem to turn off the carrier and deactivate the 'clear to send' line **4**.
11. Terminal A and modem A now become receivers and wait for a response from terminal B, indicating that all data has reached terminal B. Modem A begins an echo delay (50 to 150 milliseconds) to ensure that all echoes on the line have diminished before it begins receiving. An echo is a reflection of the transmitted signal. If the transmitting modem changed to receive too soon, it could receive a reflection (echo) of the signal it just transmitted.
12. Modem B deactivates the 'received line signal detector' line **10** and, if necessary, deactivates the receive clock signals on the 'receiver signal element timing' line **11**.
13. Terminal B now becomes the transmitter to respond to the request from terminal A. To transmit data, terminal B activates the 'request to send' line **13**, which causes modem B to transmit a carrier to modem A.
14. Modem B begins a delay that is longer than the echo delay at modem A before turning on the 'clear to send' line. The longer delay (called request-to-send to clear-to-send delay) ensures that modem A is ready to receive when terminal B begins transmitting data. After the delay, modem B activates the 'clear to send' line **14** to indicate that terminal B can begin transmitting its response.
15. After the echo delay at modem A, modem A senses the carrier from modem B (the carrier was activated in step 13 when terminal B activated the 'request to send' line) and activates the 'received line signal detector' line **7** to terminal A.
16. Modem A and terminal A are now ready to receive the response from terminal B. Remember, the response was not transmitted until after the request-to-send to clear-to-send delay at modem B (step 14).



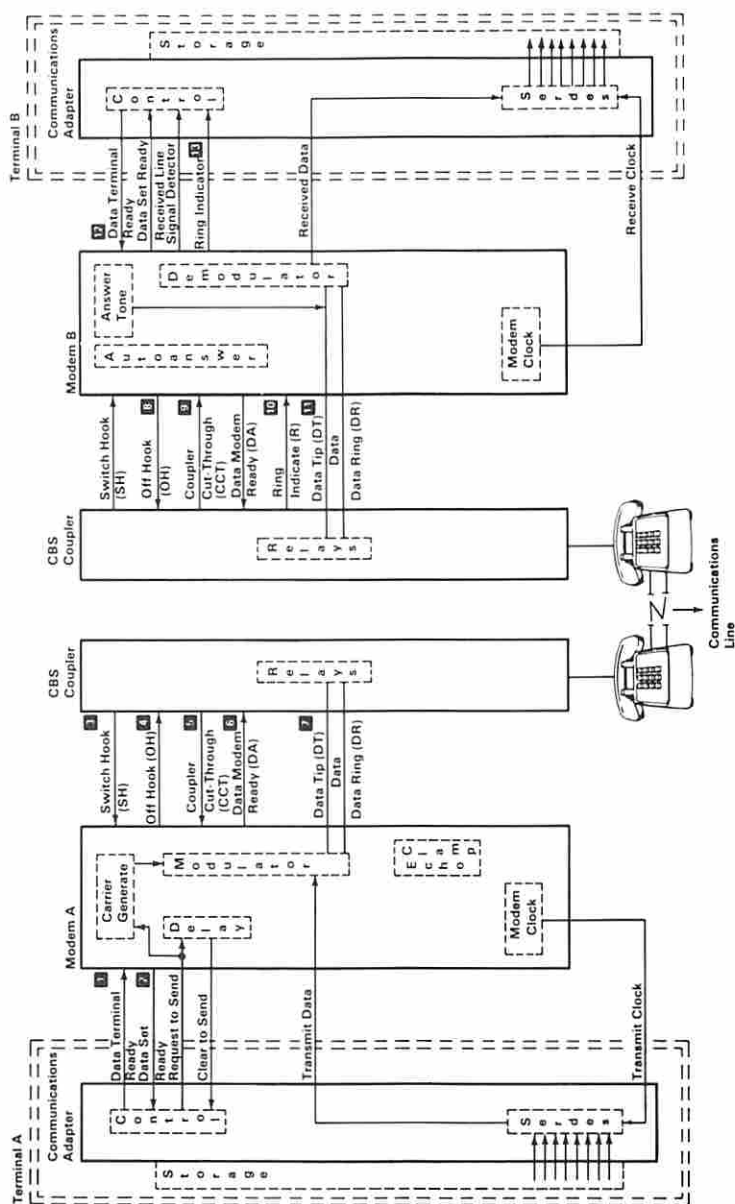
Establishing a Link on a Nonswitched Multipoint Line

1. The control station serializes the address for the tributary or secondary station (AA) and sends its address to the modem on the 'transmitted data' line **2**.
2. Since the 'request to send' line and, therefore, the modem carrier, is active continuously **1**, the modem immediately modulates the carrier with the address, and, thus, the address is transmitted to all modems on the line.
3. All tributary modems, including the modem for station A, demodulate the address and send it to their terminals on the 'received data' line **5**.
4. Only station A responds to the address; the other stations ignore the address and continue monitoring their 'received data' line. To respond to the poll, station A activates its 'request to send' line **6**, which causes the modem to begin transmitting a carrier signal.
5. The control station's modem receives the carrier and activates the 'received line signal detector, line **3** and the 'receiver signal element timing' line **4** to send clock signals to the control station). Some modems activate the clock signals as soon as they are powered on.
6. After a short delay to allow the control station modem to receive the carrier, the tributary modem activates the 'clear to send' line **7**.
7. When station A detects the active 'clear to send' line, it transmits its response. (For this example, assume that station A has no data to send; therefore, it transmits an EOT **8**.)
8. After transmitting the EOT, station A deactivates the 'request to send' line **6**. This causes the modem to deactivate the carrier and the 'clear to send' line **7**.
9. When the modem at the control station (host) detects the absence of the carrier, it deactivates the 'received line signal detector' line **3**.
10. Tributary station A is now in receive mode waiting for the next poll or select transmission from the control station.



Establishing a Link on a Switched Point-To-Point Line

1. Terminal A is in communications mode; therefore, the 'data terminal ready' line **1** is active. Terminal B is in communication mode waiting for a call from terminal A.
2. When the terminal A operator lifts the telephone handset, the 'switch hook' line from the coupler is activated **3**.
3. Modem A detects the 'switch hook' line and activates the 'off hook' line **4**, which causes the coupler to connect the telephone set to the line and activate the 'coupler cut-through' line **5** to the modem.
4. Modem A activates the 'data modem ready' line **6** to the coupler (the 'data modem ready' line is on continuously in some modems).
5. The terminal A operator sets the exclusion key or talk/data switch to the talk position to connect the handset to the communications line. The operator then dials the terminal B number.
6. When the telephone at terminal B rings, the coupler activates the 'ring indicate' line to modem B **10**. Modem B indicates that the 'ring indicate' line was activated by activating the 'ring indicator' line **13** to terminal B.
7. Terminal B activates the 'data terminal ready' line to modem B **12** which activates the autoanswer circuits in modem B. (The 'data terminal ready' line might already be active in some terminals.)
8. The autoanswer circuits in modem B activate the 'off hook' line to the coupler **8**.
9. The coupler connects modem B to the communications line via the 'data tip' and 'data ring' lines **11** and activates the 'coupler cut-through' line **9** to the modem. Modem B then transmits an answer tone to terminal A.
10. The terminal A operator hears the tone and sets the exclusion key or talk/data switch to the data position (or performs an equivalent operation) to connect modem A to the communications line through the 'data tip' and 'data ring' lines **7**.
11. The coupler at terminal A deactivates the 'switch hook' line **3**. This causes modem A to activate the 'data set ready' line **2** indicating to terminal A that the modem is connected to the communications line.
The sequence of the remaining steps to establish the data link is the same as the sequence required on a nonswitched point-to-point line. When the terminals have completed their transmission, they both deactivate the 'data terminal ready' line to disconnect the modems from the line.



Notes:

GLOSSARY

μs: Microsecond.

adapter: An auxiliary system or unit used to extend the operation of another system.

address bus: One or more conductors used to carry the binary-coded address from the microprocessor throughout the rest of the system.

all points addressable (APA): A mode in which all points on a displayable image can be controlled by the user.

alphanumeric (A/N): Pertaining to a character set that contains letters, digits, and usually other characters, such as punctuation marks. Synonymous with alphameric.

American Standard Code for Information Interchange (ASCII): The standard code, using a coded character set consisting of 7-bit coded characters (8 bits including parity check), used for information interchange among data processing systems, data communication systems and associated equipment. The ASCII set consists of control characters and graphic characters.

A/N: Alphanumeric.

analog: (1) pertaining to data in the form of continuously variable physical quantities. (2) Contrast with digital.

AND: A logic operator having the property that if P is a statement, Q is a statement, R is a statement,..., then the AND of P, Q, R,... is true if all statements are true, false if any statement is false.

APA: All points addressable.

ASCII: American Standard Code for Information Interchange.

assembler: A computer program used to assemble. Synonymous with assembly program.

asynchronous communications: A communication mode in which each single byte of data is synchronized, usually by the addition of start/stop bits.

BASIC: Beginner's all-purpose symbolic instruction code.

basic input/output system (BIOS): Provides the device level control of the major I/O devices in a computer system, which provides an operational interface to the system and relieves the programmer from concern over hardware device characteristics.

baud: (1) A unit of signaling speed equal to the number of discrete conditions or signal events per second. For example, one baud equals one-half dot cycle per second in Morse code, one bit per second in a train of binary signals, and one 3-bit value per second in a train of signals each of which can assume one of eight different states. (2) In asynchronous transmission, the unit of modulation rate corresponding to one unit of interval per second; that is, if the duration of the unit interval is 20 milliseconds, the modulation rate is 50 baud.

BCC: Block-check character.

beginner's all-purpose symbolic instruction code (BASIC): A programming language with a small repertoire of commands and a simple syntax, primarily designed for numerical application.

binary: (1) Pertaining to a selection, choice, or condition that has two possible values or states. (2) Pertaining to a fixed radix numeration system having a radix of two.

binary digit: (1) In binary notation, either of the characters 0 or 1. (2) Synonymous with bit.

binary notation: Any notation that uses two different characters, usually the binary digits 0 and 1.

BIOS: Basic input/output system.

bit: In binary notation, either of the characters 0 or 1.

bits per second (bps): A unit of measurement representing the number of discrete binary digits which can be transmitted by a device in one second.

block-check character (BCC): In cyclic redundancy checking, a character that is transmitted by the sender after each message block and is compared with a block-check character computed by the receiver to determine if the transmission was successful.

boolean operation: (1) Any operation in which each of the operands and the result take one of two values. (2) An operation that follows the rules of boolean algebra.

bootstrap: A technique or device designed to bring itself into a desired state by means of its own action; that is, a machine routine whose first few instructions are sufficient to bring the rest of itself into the computer from an input device.

bps: Bits per second.

buffer: (1) An area of storage that is temporarily reserved for use in performing an input/output operation, into which data is read or from which data is written. Synonymous with I/O area. (2) A portion of storage for temporarily holding input or output data.

bus: One or more conductors used for transmitting signals or power.

byte: (1) A binary character operated upon as a unit and usually shorter than a computer word. (2) The representation of a character.

CAS: Column address strobe.

cathode ray tube (CRT): A vacuum tube display in which a beam of electrons can be controlled to form alphanumeric characters or symbols on a luminescent screen, for example by use of a dot matrix.

cathode ray tube display (CRT display): (1) a device that presents data in visual form by means of controlled electron beams. (2) The data display produced by the device as in (1).

CCITT: Comite Consultatif International Telegrafique et Telephonique.

central processing unit (CPU): A functional unit that consists of one or more processors and all or part of internal storage.

channel: A path along which signals can be sent; for example, data channel or I/O channel.

characters per second (cps): A standard unit of measurement for printer output.

code: (1) A set of unambiguous rules specifying the manner in which data may be represented in a discrete form. Synonymous with coding scheme. (2) A set of items, such as abbreviations, representing the members of another set. (3) Loosely, one or more computer programs, or part of a computer program. (4) To represent data or a computer program in a symbolic form that can be accepted by a data processor.

column address strobe (CAS): A signal that latches the column addresses in a memory chip.

Comite Consultatif International Telegrafique et Telephonique (CCITT): Consultative Committee on International Telegraphy and Telephony.

computer: A functional unit that can perform substantial computation, including numerous arithmetic operations, or logic operations, without intervention by a human operator during the run.

configuration: (1) The arrangement of a computer system or network as defined by the nature, number, and the chief characteristics of its functional units. More specifically, the term configuration may refer to a hardware configuration or a software configuration. (2) The devices and programs that make up a system, subsystem, or network.

conjunction: (1) The boolean operation whose result has the boolean value 1 if, and only if, each operand has the boolean value 1. (2) Synonymous with AND operation.

contiguous: (1) Touching or joining at the edge or boundary. (2) Adjacent.

CPS: Characters per second.

CPU: Central processing unit.

CRC: Cyclic redundancy check.

CRT: Cathode ray tube.

CRT display: Cathode ray tube display.

CTS: Clear to send. Associated with modem control.

cyclic redundancy check (CRC): (1) A redundancy check in which the check key is generated by a cyclic algorithm. (2) A system of error checking performed at both the sending and receiving station after a block-check character has been accumulated.

cylinder: (1) The set of all tracks with the same nominal distance from the axis about which the disk rotates. (2) The tracks of a disk storage device that can be accessed without repositioning the access mechanism.

daisy-chained cable: A type of cable that has two or more connectors attached in series.

data: (1) A representation of facts, concepts, or instructions in a formalized manner suitable for communication, interpretation, or processing by humans or automatic means. (2) Any representations, such as characters or analog quantities, to which meaning is, or might be assigned.

decoupling capacitor: A capacitor that provides a low-impedance path to ground to prevent common coupling between states of a circuit.

Deutsche Industrie Norm (DIN): (1) German Industrial Norm. (2) The committee that sets German dimension standards.

digit: (1) A graphic character that represents an integer, for example, one of the characters 0 to 9. (2) A symbol that represents one of the non-negative integers smaller than the radix. For example, in decimal notation, a digit is one of the characters from 0 to 9.

digital: (1) Pertaining to data in the form of digits. (2) Contrast with analog.

DIN: Deutsche Industrie Norm.

DIN connector: One of the connectors specified by the DIN standardization committee.

DIP: Dual in-line package.

direct memory access (DMA): A method of transferring data between main storage and I/O devices that does not require processor intervention.

disk: Loosely, a magnetic disk unit.

diskette: A thin, flexible magnetic disk and a semi-rigid protective jacket, in which the disk is permanently enclosed. Synonymous with flexible disk.

DMA: Direct memory access.

DSR: Data set ready. Associated with modem control.

DTR: Data terminal ready. Associated with modem control.

dual in-line package (DIP): A widely used container for an integrated circuit. DIPs are pins usually in two parallel rows. These pins are spaced 1/10 inch apart and come in different configurations ranging from 14-pin to 40-pin configurations.

EBCDIC: Extended binary-coded decimal interchange code.

ECC: Error checking and correction.

edge connector: A terminal block with a number of contacts attached to the edge of a printed circuit board to facilitate plugging into a foundation circuit.

EIA: Electronic Industries Association.

EIA/CCITT: Electronics Industries Association/Consultative Committee on International Telegraphy and Telephony.

end-of-text character (ETX): A transmission control character used to terminate text.

end-of-transmission character (EOT): A transmission control character used to indicate the conclusion of a transmission, which may have included one or more texts and any associated message headings.

EOT: End-of-transmission character.

EPROM: Erasable programmable read-only memory.

erasable programmable read-only memory (EPROM): A storage device whose contents can be changed by electrical means. EPROM information is not destroyed when power is removed.

error checking and correction (ECC): The detection and correction of all single-bit, double-bit, and some multiple-bit errors.

ETX: End-of-text character.

extended binary-coded decimal interchange code (EBCDIC): A set of 256 characters, each represented by eight bits.

flexible disk: Synonym for diskette.

firmware: Memory chips with integrated programs already incorporated on the chip.

gate: (1) A device or circuit that has no output until it is triggered into operation by one or more enabling signals, or until an input signal exceeds a predetermined threshold amplitude.
(2) A signal that triggers the passage of other signals through a circuit.

graphic: A symbol produced by a process such as handwriting, drawing, or printing.

hertz (Hz): A unit of frequency equal to one cycle per second.

hex: Abbreviation for hexadecimal.

hexadecimal: Pertaining to a selection, choice, or condition that has 16 possible values or states. These values or states usually contain 10 digits and 6 letters, A through F. Hexadecimal digits are equivalent to a power of 16.

high-order position: The leftmost position in a string of characters.

Hz: Hertz.

interface: A device that alters or converts actual electrical signals between distinct devices, programs, or systems.

k: An abbreviation for the prefix kilo; that is, 1,000 in decimal notation.

K: When referring to storage capacity, 2 to the tenth power; 1,024 in decimal notation.

KB: Kilobyte; 1,024 bytes.

kHz: A unit of frequency equal to 1,000 hertz.

kilo (k): One thousand.

latch: (1) A feedback loop in symmetrical digital circuits used to maintain a state. (2) A simple logic-circuit storage element comprising two gates as a unit.

LED: Light-emitting diode.

light-emitting diode (LED): A semi-conductor chip that gives off visible or infrared light when activated.

low-order position: The rightmost position in a string of characters.

m: (1) Milli; one thousand or thousandth part. (2) Meter.

M: Mega; 1,000,000 in decimal notation. When referring to storage capacity, 2 to the twentieth power; 1,048,576 in decimal notation.

mA: Milliampere.

machine language: (1) A language that is used directly by a machine. (2) Another term for computer instruction code.

main storage: A storage device in which the access time is effectively independent of the location of the data.

MB: Megabyte, 1,048,576 bytes.

mega (M): 10 to the sixth power, 1,000,000 in decimal notation. When referring to storage capacity, 2 to the twentieth power, 1,048,576 in decimal notation.

megabyte (MB): 1,048,576 bytes.

megahertz (MHz): A unit of measure of frequency. 1 megahertz equals 1,000,000 hertz.

MFM: Modified frequency modulation.

MHz: Megahertz.

microprocessor: An integrated circuit that accepts coded instructions for execution; the instructions may be entered, integrated, or stored internally.

microsecond (μ s): One-millionth of a second.

milli (m): One thousand or one thousandth.

milliampere (mA): One thousandth of an ampere.

millisecond (ms): One thousandth of a second.

mnemonic: A symbol chosen to assist the human memory; for example, an abbreviation such as "mpy" for "multiply."

mode: (1) A method of operation; for example, the binary mode, the interpretive mode, the alphanumeric mode. (2) The most frequent value in the statistical sense.

modem: (Modulator-Demodulator) A device that converts serial (bit by bit) digital signals from a business machine (or data terminal equipment) to analog signals which are suitable for transmission in a telephone network. The inverse function is also performed by the modem on reception of analog signals.

modified frequency modulation (MFM): The process of varying the amplitude and frequency of the "write" signal. MFM pertains to the number of bytes of storage that can be stored on the recording media. The number of bytes is twice the number contained in the same unit area of recording media at single density.

modulo check: A calculation performed on values entered into a system. This calculation is designed to detect errors.

monitor: (1) A device that observes and verifies the operation of a data processing system and indicates any specific departure from the norm. (2) A television type display, such as the IBM Monochrome Display. (3) Software or hardware that observes, supervises, controls, or verifies the operations of a system.

ms: Millisecond; one thousandth of a second.

multiplexer: A device capable of interleaving the events of two or more activities, or capable of distributing the events of an interleaved sequence to the respective activities.

NAND: A logic operator having the property that if P is a statement, Q is a statement, R is a statement,....,then the NAND of P,Q,R,...is true if at least one statement is false, false if all statements are true.

nanosecond (ns): One-thousandth-millionth of a second.

nonconjunction: (1) The dyadic boolean operation the result of which has the boolean value 0 if, and only if, each operand has the boolean value 1.

non-return-to-zero inverted (NRZI): A transmission encoding method in which the data terminal equipment changes the signal to the opposite state to send a binary 0 and leaves it in the same state to send a binary 1.

NOR: A logic operator having the property that if P is a statement, Q is a statement, R is a statement,....,then the NOR of P,Q,R,...is true if all statements are false, false if at least one statement is true.

NOT: A logical operator having the property that if P is a statement, then the NOT of P is true if P is false, false if P is true.

NRZI: Non-return-to-zero inverted.

ns: Nanosecond; one-thousandth-millionth of a second.

operating system: Software that controls the execution of programs; an operating system may provide services such as resource allocation, scheduling, input/output control, and data management.

OR: (1) A logic operator having the property that if P is a statement, Q is a statement, R is a statement,....,then the OR of P,Q,R,...is true if at least one statement is true, false if all statements are false.

output: Pertaining to a device, process, or channel involved in an output process, or to the data or states involved in an output process.

output process: (1) The process that consists of the delivery of data from a data processing system, or from any part of it. (2) The return of information from a data processing system to an end user, including the translation of data from a machine language to a language that the end user can understand.

overcurrent: A current of higher than specified strength.

overvoltage: A voltage of higher than specified value.

parallel: (1) Pertaining to the concurrent or simultaneous operation of two or more devices, or to the concurrent performance of two or more activities. (2) Pertaining to the concurrent or simultaneous occurrence of two or more related activities in multiple devices or channels. (3) Pertaining to the simultaneity of two or more processes. (4) Pertaining to the simultaneous processing of the individual parts of a whole, such as the bits of a character and the characters of a word, using separate facilities for the various parts. (5) Contrast with serial.

PEL: Picture element.

personal computer: A small home or business computer that has a processor and keyboard and that can be connected to a television or some other monitor. An optional printer is usually available.

picture element (PEL): (1) The smallest displayable unit on a display. (2) Synonymous with pixel, PEL.

pinout: A diagram of functioning pins on a pinboard.

pixel: Picture element.

polling: (1) Interrogation of devices for purposes such as to avoid contention, to determine operational status, or to determine readiness to send or receive data. (2) The process whereby stations are invited, one at a time, to transmit.

port: An access point for data entry or exit.

printed circuit board: A piece of material, usually fiberglass, that contains a layer of conductive material, usually metal. Miniature electronic components on the fiberglass transmit electronic signals through the board by way of the metal layers.

program: (1) A series of actions designed to achieve a certain result. (2) A series of instructions telling the computer how to handle a problem or task. (3) To design, write, and test computer programs.

programming language: (1) An artificial language established for expressing computer programs. (2) A set of characters and rules, with meanings assigned prior to their use, for writing computer programs.

PROM: Programmable read-only memory.

propagation delay: The time necessary for a signal to travel from one point on a circuit to another.

radix: (1) In a radix numeration system, the positive integer by which the weight of the digit place is multiplied to obtain the weight of the digit place with the next higher weight; for example, in the decimal numeration system, the radix of each digit place is 10. (2) Another term for base.

radix numeration system: A positional representation system in which the ratio of the weight of any one digit place to the weight of the digit place with the next lower weight is a positive integer. The permissible values of the character in any digit place range from zero to one less than the radix of the digit place.

RAS: Row address strobe.

RGBI: Red-green-blue-intensity.

read-only memory (ROM): A storage device whose contents cannot be modified, except by a particular user, or when operating under particular conditions; for example, a storage device in which writing is prevented by a lockout.

read/write memory: A storage device whose contents can be modified.

red-green-blue-intensity (RGBI): The description of a direct-drive color monitor which accepts red, green, blue, and intensity signal inputs.

register: (1) A storage device, having a specified storage capacity such as a bit, a byte, or a computer word, and usually intended for a special purpose. (2) On a calculator, a storage device in which specific data is stored.

RF modulator: The device used to convert the composite video signal to the antenna level input of a home TV.

ROM: Read-only memory.

ROM/BIOS: The ROM resident basic input/output system, which provides the device level control of the major I/O devices in the computer system.

row address strobe (RAS): A signal that latches the row addresses in a memory chip.

RS-232C: The standard set by the EIA for communications between computers and external equipment.

RTS: Request to send. Associated with modem control.

run: A single continuous performance of a computer program or routine.

scan line: The use of a cathode beam to test the cathode ray tube of a display used with a personal computer.

schematic: The description, usually in diagram form, of the logical and physical structure of an entire data base according to a conceptual model.

SDLC: Synchronous Data Link Control.

sector: That part of a track or band on a magnetic drum, a magnetic disk, or a disk pack that can be accessed by the magnetic heads in the course of a predetermined rotational displacement of the particular device.

serdes: Serializer/deserializer.

serial: (1) Pertaining to the sequential performance of two or more activities in a single device. In English, the modifiers serial and parallel usually refer to devices, as opposed to sequential and consecutive, which refer to processes. (2) Pertaining to the sequential or consecutive occurrence of two or more related activities in a single device or channel. (3) Pertaining to the sequential processing of the individual parts of a whole, such as the bits of a character or the characters of a word, using the same facilities for successive parts. (4) Contrast with parallel.

sink: A device or circuit into which current drains.

software: (1) Computer programs, procedures, rules, and possibly associated documentation concerned with the operation of a data processing system. (2) Contrast with hardware.

source: The origin of a signal or electrical energy

source circuit: (1) Generator circuit. (2) Control with sink.

SS: Start-stop transmission.

start bit: Synonym for start signal.

start-of-text character (STX): A transmission control character that precedes a text and may be used to terminate the message heading.

start signal: (1) A signal to a receiving mechanism to get ready to receive data or perform a function. (2) In a start-stop system, a signal preceding a character or block that prepares the receiving device for the reception of the code elements. Synonymous with start bit.

start-stop (SS) transmission: Asynchronous transmission such that a group of signals representing a character is preceded by a start signal and followed by a stop signal. (2) Asynchronous transmission in which a group of bits is preceded by a start bit that prepares the receiving mechanism for the reception and registration of a character and is followed by at least one stop bit that enables the receiving mechanism to come to an idle condition pending the reception of the next character.

stop bit: Synonym for stop signal.

stop signal: (1) A signal to a receiving mechanism to wait for the next signal. (2) In a start-stop system, a signal following a character or block that prepares the receiving device for the reception of a subsequent character or block. Synonymous with stop bit.

stroke: (1) An instrument used to determine the exact speed of circular or cyclic movement. (2) A flashing signal displaying an exact event.

STX: Start-of-text character.

Synchronous Data Link Control (SDLC): A protocol for the management of data transfer over a data communications link.

synchronous transmission: Data transmission in which the sending and receiving devices are operating continuously at the same frequency and are maintained, by means of correction, in a desired phase relationship.

text: In ASCII and data communication, a sequence of characters treated as an entity if preceded and terminated by one STX and one ETX transmission control, respectively.

track: (1) The path or one of the set of paths, parallel to the reference edge on a data medium, associated with a single reading or writing component as the data medium moves past the component. (2) The portion of a moving data medium such as a drum, tape, or disk, that is accessible to a given reading head position.

transistor-transistor logic (TTL): A circuit in which the multiple-diode cluster of the diode-transistor logic circuit has been replaced by a multiple-emitter transistor.

TTL: Transistor-transistor logic.

TX Data: Transmit data. Associated with modem control. External connections of the RS-232C asynchronous communications adapter interface.

video: Computer data or graphics displayed on a cathode ray tube, monitor or display.

write precompensation: The varying of the timing of the head current from the outer tracks to the inner tracks of the diskette to keep a constant write signal.

BIBLIOGRAPHY

Intel Corporation. *The 8086 Family User's Manual*

This manual introduces the 8086 family of microcomputing components and serves as a reference in system design and implementation.

Intel Corporation. *8086/8087/8088 Macro Assembly Reference Manual for 8088/8085 Based Development System*

This manual describes the 8086/8087/8088 Macro Assembly Language, and is intended for use by persons who are familiar with assembly language.

Intel Corporation. *Component Data Catalog*

This book describes Intel components and their technical specifications.

Motorola, Inc. *The Complete Microcomputer Data Library*.

This book describes Motorola components and their technical specifications.

National Semiconductor Corporation. *INS 8250 Asynchronous Communications Element*. This book documents physical and operating characteristics of the INS 8250.

Notes:

INDEX

A

- A/N mode (alphanumeric mode) 1-93
- A0-A19 (Address Bits 0 to 19), I/O channel 1-17
- adapter cards with ROM 2-10
- adapter,
 - asynchronous communications 1-197
 - color/graphics monitor 1-93
 - diskette drive 1-117
 - fixed disk drive 1-145
 - game control 1-185
 - monochrome display and printer 1-83
 - printer 1-77
 - synchronous data link control 1-227
- Address Bits 0 to 19 (A0-A19), I/O channel 1-17
- address bits, asynchronous communication 1-199
- Address Enable (AEN), I/O channel 1-19
- Address Latch Enable (ALE), I/O channel 1-17
- address map, I/O 1-8
- AEN (Address Enable), I/O channel 1-19
- ALE (Address Latch Enable), I/O channel 1-17
- all points addressable mode 1-93
- alphanumeric mode, 1-97
 - high resolution 1-98
 - low resolution 1-98
- alt (keyboard extended code) 2-15
- APA mode (all points addressable mode) 1-93
- asynchronous communications adapter, 1-197
 - address bits 1-199
 - address jumper module 1-224
 - block diagram 1-198
 - connector specifications 1-225
 - current loop interface 1-201
 - divisor latch least significant bit 1-211
 - divisor latch most significant bit 1-212
 - I/O decodes 1-199
 - INS8250 functional pin description 1-203
 - INS8250 input signals 1-203

asynchronous communications adapter (continued)

- INS8250 input/output signals 1-207
- INS8250 output signals 1-206
- interface description 1-200
- interface format jumper module 1-224
- interrupt control functions 1-216
- interrupt enable register 1-217
- interrupt identification register 1-214
- interrupts 1-200
- line status register 1-213
- line-control register 1-209
- modem control register 1-218
- modem status register 1-220
- modes of operation 1-198
- programmable baud rate generator 1-211
- programming considerations 1-208
- receiver buffer register 1-222
- reset functions 1-208
- transmitter holding register 1-223
- voltage interchange information 1-202
- attributes, character
(see character attributes)

B

- BASIC reserved interrupts 2-7
- BASIC,
 - DEF SEG 2-8
 - reserved interrupts 2-7
 - screen editor keyboard functions 2-20
 - workspace variables 2-8
- baud rate generator 1-209
- bell (printer) 1-64
- bibliography H-1
- BIOS,
 - cassette logic (see cassette logic BIOS)
 - fixed disk ROM A-87
 - memory map 2-9
 - parameter passing 2-3
 - software interrupt listing 2-4
 - system ROM A-2
 - use of 2-2

- block diagram,
 - 8273 SDLC protocol controller 1-229
 - asynchronous communications adapter 1-198
 - cassette circuits 1-26
 - color/graphics monitor adapter 1-95
 - diskette drive adapter 1-118
 - expansion board 1-40
 - extender card 1-48
 - fixed disk drive adapter 1-146
 - game control adapter 1-185
 - keyboard interface 1-30
 - monochrome display adapter 1-84
 - printer adapter 1-78
 - prototype card 1-192
 - receiver card 1-50
 - speaker drive system 1-20
 - synchronous data link control adapter 1-227
 - system 1-2
- break (keyboard extended code) 2-16

C

- cable,
 - communications adapter 1-257
 - expansion unit 1-39
 - printer 1-53
- cancel (printer) 1-65
- cancel ignore paper end (printer) 1-67
- cancel skip perforation (printer) 1-70
- caps lock (keyboard extended code) 2-16
- card dimensions and specifications E-4
- card,
 - dimensions and specifications E-4
 - extender 1-45
 - prototype 1-191
 - receiver 1-49
- carriage return (printer) 1-65
- cassette circuit block diagram,
 - motor control 1-27
 - read hardware 1-26
 - write hardware 1-26
- cassette interface, 1-25
 - connector specifications 1-28

- cassette logic,
 - BIOS 2-21
 - cassette read 2-23
 - cassette write 2-22
 - data record architecture 2-24
 - data record components 2-24
 - error recovery 2-24
 - interrupt 2-21
 - software algorithms 2-21
- cassette read 2-23
- cassette ROM BIOS 2-21
- cassette write 2-22
- CCITT, F-1
 - standards F-1
- character attributes
 - color/graphics monitor adapter 1-99
 - monochrome display adapter 1-99
- character codes,
 - keyboard 2-11
- character set,
 - graphics printer (set 1) 1-72
 - graphics printer (set 2) 1-74
 - quick reference C-12
- clear printer buffer (printer) 1-65
- CLK (system clock), I/O channel 1-17
- color display, 1-115
 - operating characteristics 1-115
 - specifications E-2
- color select register 1-107
- color/graphics monitor adapter, 1-93
 - 6845 register description 1-106
 - alphanumeric mode 1-97
 - alphanumeric mode (high-resolution) 1-98
 - alphanumeric mode (low-resolution) 1-98
 - block diagram 1-95
 - character attributes 1-99
 - color-select register 1-107
 - composite connector specifications 1-113
 - connector specifications 1-113
 - direct-drive connector specifications 1-113
 - display buffer basic operation 1-104
 - graphics mode 1-100
 - graphics mode (high resolution) 1-103

- color/graphics monitor adapter (continued)
 - graphics mode (low resolution) 1-100
 - graphics mode (medium resolution) 1-101
 - light pen connector specifications 1-114
 - major components 1-96
 - memory requirements 1-112
 - mode control and status register 1-107
 - mode register summary 1-110
 - mode-select register 1-109
 - programming considerations 1-105
 - RF modulator connector specifications 1-114
 - sequence of events 1-111
 - status register 1-111
 - summary of available colors 1-105
- colors, summary of available 1-105
- command status register 0 1-130
- command status register 1 1-131
- command status register 2 1-132
- command status register 3 1-133
- command summary,
 - diskette drive adapter 1-124
 - fixed disk drive adapter 1-153
- communications adapter cable, 1-257
 - connector specifications 1-258
- communications, F-1
 - establishing a link F-3
- component diagram,
 - system board 1-12
- compressed (printer) 1-65
- compressed off (printer) 1-65
- connector specifications,
 - asynchronous communications adapter 1-225
 - cassette interface 1-28
 - color/graphics monitor adapter 1-113
 - communications adapter cable 1-258
 - diskette drive adapter (external) 1-139
 - diskette drive adapter (internal) 1-138
 - game control adapter 1-190
 - keyboard interface 1-38
 - monochrome display adapter 1-89
 - printer adapter 1-81
 - synchronous data link control adapter 1-255

- connectors,
 - power supply (system unit) 1-23
 - power supply (expansion unit) 1-43
- consideration, programming
 - (see programming considerations)
- control byte, fixed disk drive adapter 1-152
- control codes, printer 1-63
- control/read/write logic 1-230
- ctrl (keyboard extended code) 2-15
- current loop interface 1-201

D

- D0-D7 (Data Bits 0 to 7), I/O channel 1-17
- DACK0-DACK3 (DMA Acknowledge 0 to 3), I/O channel 1-19
- Data Bits 0 to 7 (D0-D7), I/O channel 1-17
- data flow,
 - system board 1-6
- data record architecture, cassette 2-24
- data record components, cassette 2-24
- data register 1-151
- data transfer mode register 1-244
- DEF SEG (default segment workspace) 2-8
- default workspace segment (DEF SEG) 2-8
- diagram, block (see block diagram)
- digital output register 1-119
- diskette drive adapter, 1-117
 - adapter inputs 1-137
 - adapter outputs 1-136
 - block diagram 1-118
 - command status register 0 1-130
 - command status register 1 1-131
 - command status register 2 1-132
 - command status register 3 1-133
 - command summary 1-124
 - connector specifications (external) 1-139
 - connector specifications (internal) 1-138
 - digital-output register 1-119
 - DPC registers 1-133
 - drive A and B interface 1-136
 - drive constants 1-134
 - FDC constants 1-134
 - floppy disk controller 1-120

- diskette drive adapter (continued)
 - functional description 1-119
 - programming considerations 1-122
 - programming summary 1-133
 - symbol descriptions 1-122
 - system I/O channel interface 1-134
- diskette drive, 1-141
 - electrical specifications 1-142
 - mechanical specifications 1-142
 - switch settings 1-14
- diskettes 1-143
- display adapter type switch settings 1-14
- display,
 - color 1-115
 - monochrome 1-91
- divisor latch,
 - least significant bit 1-211
 - most significant bit 1-212
- DMA Acknowledge 0 to 3 (DACK0-DACK3), I/O channel 1-19
- DMA Request 1 to 3 (DRQ1-DRQ3), I/O channel 1-19
- DOS reserved interrupts 2-7
- DOS,
 - keyboard functions 2-20
 - reserved interrupts 2-7
- double strike (printer) 1-68
- double strike off (printer) 1-68
- double width (printer) 1-65, 1-71
- double width off (printer) 1-65
- DPC registers 1-133
- DRQ1-DRQ3 (DMA Request 1 to 3), I/O channel 1-19

E

- EIA, F-1
 - standards F-1
- emphasized (printer) 1-68
- emphasized off (printer) 1-68
- error recovery, cassette 2-24
- escape (printer) 1-66
- establishing a communications link F-3
- expansion board, 1-39
 - block diagram 1-40
- expansion channel 1-41

- expansion unit, 1-39
 - cable 1-39
 - expansion board 1-39
 - expansion channel 1-41
 - extender card 1-45
 - interface information 1-51
 - power supply 1-42
 - power supply connectors 1-43
 - receiver card 1-49
 - specifications E-2
- extender card, 1-45
 - block diagram 1-48
 - programming considerations 1-47
 - switch settings 1-46

F

- fixed disk controller 1-145
- fixed disk drive 1-161
- fixed disk drive adapter, 1-145
 - block diagram 1-146
 - command summary 1-153
 - control byte 1-152
 - data register 1-151
 - electrical specifications 1-162
 - fixed disk controller 1-145
 - interface specifications 1-159
 - mechanical specifications 1-162
 - programming considerations 1-147
 - programming summary 1-157
 - ROM BIOS listing A-87
 - sense bytes 1-147
 - status register 1-147
 - system I/O channel interface 1-158
- fixed disk ROM BIOS A-87
- floppy disk controller 1-120
- form feed (printer) 1-64

G

- game control adapter, 1-185
 - block diagram 1-185
 - connector specifications 1-190
 - functional description 1-186
 - I/O channel description 1-187
 - interface description 1-187
 - joy stick schematic diagram 1-189
- glossary, G-1
- graphics mode, 1-100
 - high resolution 1-103
 - low resolution 1-100
 - medium resolution 1-101

H

- hardware interrupt listing 1-8
- home head (printer) 1-67
- horizontal tab (printer) 1-64

I

- I/O address map 1-8
- I/O bit map, 8255A 1-9
- I/O CH CK (I/O Channel Check), I/O channel 1-17
- I/O CH RDY (I/O Channel Ready), I/O channel 1-18
- I/O Channel Check (I/O CH CK), I/O channel 1-17
- I/O channel interface,
 - diskette drive adapter 1-134
 - fixed disk drive adapter 1-158
 - prototype card 1-193
- I/O Channel Ready (I/O CH RDY), I/O channel 1-18
- I/O channel, 1-15
 - I/O Channel Check (I/O CH CK) 1-17
 - I/O Read Command (IOR) 1-18
 - I/O Write Command (IOW) 1-18
 - Address Bits 0 to 19 (A0-A19) 1-17
 - Address Enable (AEN) 1-19
 - Address Latch Enable (ALE) 1-17
 - Data Bits 0 to 7 (D0-D7) 1-17
 - description 1-17

I/O channel (continued)

- diagram 1-16
- DMA Request 1 to 3 (DRQ1-DRQ3) 1-19
- I/O Channel Ready (I/O CH RDY) 1-18
- Interrupt Request 2 to 7 (IRQ2-IRQ7) 1-18
- Memory Read Command (MEMR) 1-18
- Memory Write Command (MEMW) 1-18
- Oscillator (OSC) 1-17
- Reset Drive (RESET DRV) 1-17
- System Clock (CLK) 1-17
- Terminal Count (T/C) 1-19
- I/O Read Command (IOR), I/O channel 1-18
- I/O Write Command (IOW), I/O channel 1-18
- IBM 10MB Fixed Disk Drive 1-161
- IBM 5-1/4" Diskette Drive 1-141
- IBM 5-1/4" Diskette Drive Adapter 1-117
- IBM 80 CPS Graphics Printer 1-53
- IBM Asynchronous Communications Adapter 1-197
- IBM Color Display 1-115
- IBM Color/Graphics Monitor Adapter 1-93
- IBM Communications Adapter Cable 1-257
- IBM Fixed Disk Drive Adapter 1-145
- IBM Game Control Adapter 1-185
- IBM Memory Expansion Options 1-163
- IBM Monochrome Display and Printer Adapter 1-83
- IBM Monochrome Display 1-91
- IBM Printer Adapter 1-77
- IBM Prototype Card 1-191
- IBM Synchronous Data Link Controller Adapter 1-227
- ignore paper end (printer) 1-67
- INS8250,
(see National Semiconductor INS8250)
- Intel 8088 microprocessor,
 - arithmetic B-7
 - conditional transfer operations B-14
 - control transfer B-11
 - data transfer B-5
 - hardware interrupt listing 1-8
 - instruction set index B-18
 - instruction set matrix B-16
 - logic B-9
 - memory segmentation model B-4

- Intel 8088 microprocessor (continued)
 - operand summary B-3
 - processor control B-15
 - register model B-2
 - second instruction byte summary B-3
 - segment override prefix B-4
 - software interrupt listing 2-4
 - string manipulation B-10
 - use of segment override B-4
- Intel 8253-5 Programmable Interval Timer)
 - (see synchronous data link control communications adapter)
- Intel 8255A Programmable Peripheral Interface
 - I/O bit map 1-9
- Intel 8255A-5 Programmable Peripheral Interface
 - (see synchronous data link control communications adapter)
- Intel 8273 SDLC Protocol Controller
 - (see synchronous data link control communications adapter)
 - block diagram 1-229
- interrupt enable register 1-217
- interrupt identification register 1-214
- interrupt listing,
 - 8088 hardware 1-8
 - 8088 software 2-4
- Interrupt Request 1 to 7 (IRQ1-IRQ7), I/O channel 1-18
- interrupts,
 - 8088 hardware 1-8
 - 8088 software 2-4
 - asynchronous communications adapter 1-200
 - BASIC reserved 2-7
 - cassette logic 2-21
 - DOS reserved 2-7
 - special 2-5
- IOR (I/O Read Command), I/O channel 1-18
- IOW (I/O Write Command), I/O channel 1-18
- IRQ1-IRQ7 (Interrupt Request 1 to 7), I/O channel 1-18

J

- joy stick,
 - positions 1-186
 - schematic diagram 1-189
- jumper module, asynchronous communications adapter 1-224

K

keyboard extended codes

- alt 2-15
- break 2-16
- caps lock 2-16
- ctrl 2-15
- pause 2-17
- print screen 2-17
- scroll lock 2-16
- shift 2-15
- shift key priorities 2-16
- shift states 2-15
- system reset 2-16

keyboard 1-29

- BASIC screen editor special functions 2-20
- character codes 2-11
- commonly used functions 2-18
- diagram 1-31
- DOS special functions 2-20
- encoding 2-11
- extended functions 2-14
- interface block diagram 1-30
- interface connector specifications 1-38
- scan codes 1-37
- specifications E-1

L

light pen connector specifications 1-114

line control register 1-209

line feed (printer) 1-64

line status register 1-213

logic diagrams D-1

M

memory expansion options, 1-163

- DIP module start address 1-166
- memory module description 1-164
- memory module pin configuration 1-165
- memory option switch settings 1-169

- memory expansion options (continued)
 - R/W memory operating characteristics 1-164
 - switch-configurable start address 1-166
- memory locations,
 - reserved 2-8
- memory map,
 - BIOS 2-9
 - system 1-10
- Memory Read Command (MEMR), I/O channel 1-18
- memory switch settings,
 - extender card 1-46
 - memory options 1-163
 - system board 1-13
- Memory Write Command (MEMW), I/O channel 1-18
- MEMR (Memory Read Command), I/O channel 1-18
- MEMW (Memory Write Command), I/O channel 1-18
- microprocessor (see Intel 8088 microprocessor)
- mode control and status register 1-107
- mode select register 1-109
- modem control register 1-218
- modem status register 1-220
- monochrome display 1-91
- monochrome display and printer adapter 1-83
- monochrome display adapter 1-83
 - 6845 CRT control port 1 1-88
 - 6845 CRT status port 1-88
 - block diagram 1-84
 - character attributes 1-99
 - connector specifications 1-89
 - I/O address and bit map 1-87
 - programming considerations 1-85
- monochrome display, 1-91
 - operating characteristics 1-91
 - specifications E-3
- Motorola 6845 CRT Controller,
 - (see color/graphics monitor adapter)
 - (see monochrome display adapter)

N

- National Semiconductor INS8250 Asynchronous
(see asynchronous communications adapter)
- functional pin description 1-203
- input signals 1-203
- input/output signals 1-207
- output signals 1-206
- null (printer) 1-64

O

- one bit delay mode register 1-245
- operating mode register 1-242
- OSC (Oscillator) 1-17
- Oscillator (OSC), I/O channel 1-17
- over-voltage/over-current (expansion unit) 1-44
- over-voltage/over-current (system unit)-24

P

- parameter passing (ROM BIOS) 2-3
- pause (keyboard extended code) 2-17
- power good signal (expansion unit) 1-44
- power good signal (system unit) 1-24
- power supply (expansion unit) 1-42
 - connectors 1-43
 - input requirements 1-42
 - over-voltage/current protection 1-44
 - pin assignments 1-43
 - power good signal 1-44
 - Vac output 1-42
 - Vdc output 1-42
- power supply (system unit) 1-21
 - connectors and pin assignments 1-23
 - input requirements 1-22
 - over-voltage/current protection 1-24
 - pin assignments 1-23
 - power good signal 1-24
 - Vac output 1-22
 - Vdc output 1-22

- print screen (keyboard extended code) 2-17
- printer adapter, 1-77
 - block diagram 1-78
 - connector specifications 1-81
 - programming considerations 1-79
- printer control codes, 1-63
 - 1/8-inch line feeding 1-66
 - 1920 bit-image graphics mode 1-71
 - 480 bit-image graphics mode 1-69
 - 7/72-inch line feeding 1-66
 - 960 bit-image graphics mode 1-70
 - 960 bit-image graphics mode normal speed 1-71
- bell 1-64
- cancel 1-65
- cancel ignore paper end 1-67
- cancel skip perforation 1-70
- carriage return 1-65
- clear printer buffer 1-65
- compressed 1-65
- compressed off 1-65
- double strike 1-68
- double strike off 1-68
- double width 1-65, 1-71
- double width off 1-65
- emphasized 1-68
- emphasized off 1-68
- escape 1-66
- form feed 1-64
- home head 1-67
- horizontal tab 1-64
- ignore paper end 1-67
- line feed 1-64
- null 1-64
- select character set 1 1-66
- select character set 2 1-66
- set horizontal tab stops 1-68
- set lines per page 1-67
- set skip perforation 1-70
- set variable line feeding 1-67
- set vertical tabs 1-64
- starts variable line feeding 1-66

- printer control codes (continued)
 - subscript/superscript 1-71
 - subscript/superscript off 1-71
 - underline 1-66
 - unidirectional printing 1-71
 - vertical tab 1-64
- printer, 1-53
 - cable 1-53
 - connector pin assignment 1-59
 - control codes 1-63
 - graphics character set 1 1-72
 - graphics character set 2 1-74
 - interface signal descriptions 1-59
 - modes 1-62
 - parallel interface 1-58
 - parallel interface timing diagram 1-58
 - specifications 1-54, E-3
 - switch locations 1-56
 - switch settings 1-56
- processor (see Intel 8088 microprocessor)
- programmable baud rate generator 1-211
- programming considerations,
 - asynchronous communications adapter 1-208
 - color/graphics monitor adapter 1-105
 - diskette drive adapter 1-122
 - extender card 1-47
 - fixed disk drive adapter 1-147
 - monochrome display adapter 1-85
 - printer adapter 1-79
 - receiver card 1-49
 - SDLC adapter 1-237
- prototype card, 1-191
 - block diagram 1-192
 - external interface 1-196
 - I/O channel interface 1-193
 - layout 1-193
 - system loading and power limitations 1-195

Q

- quick reference, character set C-12

R

- receiver buffer register 1-222
- receiver card, 1-49
 - block diagram 1-50
 - programming considerations 1-49
- register,
 - 6845 description (color/graphics adapter) 1-106
 - color select (color/graphics adapter) 1-107
 - command status 0 (diskette drive adapter) 1-130
 - command status 1 (diskette drive adapter) 1-131
 - command status 2 (diskette drive adapter) 1-132
 - command status 3 (diskette drive adapter) 1-133
 - data (fixed disk drive adapter) 1-151
 - data transfer mode (SDLC) 1-244
 - digital output (diskette drive adapter) 1-119
 - DPC (diskette drive adapter) 1-133
 - interrupt enable (asynchronous communications) 1-217
 - interrupt identification (asynchronous communications) 1-214
 - line control (asynchronous communications) 1-209
 - line status (asynchronous communications) 1-213
 - mode control and status (color/graphics) 1-107
 - mode select (color/graphics) 1-109
 - modem control (asynchronous communications) 1-218
 - modem status (asynchronous communications) 1-220
 - one-bit delay mode (SDLC) 1-245
 - operating mode (SDLC) 1-242
 - receiver buffer (asynchronous communications) 1-222
 - serial I/O mode (SDLC) 1-244
 - status (color/graphics) 1-111
 - status (fixed disk drive adapter) 1-147
 - transmitter holding (asynchronous communications) 1-223
- reserved interrupts,
 - BASIC and DOS 2-7
- reserved memory locations 2-8
- Reset Drive (RESET DRV), I/O channel 1-17
 - RESET DRV (Reset Drive), I/O channel 1-17
- RF modulator connector specifications 1-114
- ROM BIOS, 2-2
 - Cassette A-74
 - Fixed Disk A-87
 - System A-2

ROM, adapter cards with 2-10
RS-232C,
interface standards F-2

S

scan codes,
 keyboard 1-37
scroll lock (keyboard extended code) 2-16
SDLC (see synchronous data link control communications
 adapter)
select character set 1 (printer) 1-66
select character set 2 (printer) 1-66
sense bytes, fixed disk drive adapter 1-145
serial I/O mode register 1-242
set horizontal tab stops (printer) 1-68
set lines per page (printer) 1-67
 set skip perforation (printer) 1-70
set variable line feeding (printer) 1-67
set vertical tabs (printer) 1-64
shift (keyboard extended code) 2-15
shift key priorities (keyboard code) 2-16
shift states (keyboard extended code) 2-15
software interrupt listing 2-4
speaker connector 1-20
speaker drive system 1-20
speaker interface 1-20
specifications,
 80 cps printers E-3
 color display E-2
 expansion unit E-2
 keyboard E-1
 monochrome display E-3
 printer 1-54
 system unit E-1
stack area 2-7
starts variable line feeding (printer) 1-66
status register,
 color/graphics monitor adapter 1-111
 fixed disk drive adapter 1-147
 synchronous data link control adapter 1-238
subscript/superscript (printer) 1-71

- subscript/superscript off (printer) 1-71
- switch settings,
 - diskette drive 1-14
 - display adapter type 1-14
 - extender card 1-46
 - memory options 1-169
 - printer 1-56
 - system board 1-13
 - system board memory 1-13
- synchronous data link control communications adapter, 1-227
 - 8253-5 interval timer control word 1-241
 - 8253-5 programmable interval timer 1-237
 - 8255A-5 port A assignments 1-236
 - 8255A-5 port B assignments 1-236
 - 8255A-5 port C assignments 1-237
 - 8255A-5 programmable peripheral interface 1-236
 - 8273 command phase flow chart 1-248
 - 8273 commands 1-247
 - 8273 control/read/write registers 1-231
 - 8273 data interfaces 1-232
 - 8273 elements of data transfer interface 1-232
 - 8273 mode register commands 1-245
 - 8273 modem control block 1-233
 - 8273 modem control port A 1-233
 - 8273 modem control port B 1-234
 - 8273 modem interface 1-233
 - 8273 protocol controller operations 1-228
 - 8273 protocol controller structure 1-229
 - 8273 register selection 1-230
 - 8273 SDLC protocol controller block diagram 1-229
 - 8273 transmit/receiver timing 1-235
- block diagram 1-227
- command phase 1-246
- connector specifications 1-255
- control/read/write logic 1-230
- data transfer mode register 1-244
- device addresses 1-253
- execution phase 1-249
- general receive 1-250
- initialization/configuration commands 1-242
- initializing the SDLC adapter 1-240
- interface information 1-254
- interrupt information 1-253
- one bit delay code register 1-245

synchronous data link control communications adapter (continued)

- operating mode register 1-242
- partial byte received codes 1-252
- processor interface 1-230
- programming considerations 1-237
- protocol control module features 1-228
- protocol controller operations 1-228
- result code summary 1-252
- result phase 1-249
- selective receive 1-251
- serial data timing block 1-235
- serial I/O mode register 1-244
- status register format 1-238
- transmit 1-250
- system block diagram 1-2
- system board, 1-3
 - component diagram 1-12
 - data flow 1-6
 - R/W memory operating characteristics 1-164
 - switch settings 1-13
- System Clock (CLK), I/O channel 1-17
- system memory map 1-10
- system reset (keyboard extended code) 2-16
- system ROM BIOS A-2
- system unit, 1-3
 - cassette interface 1-28
 - I/O channel 1-15
 - I/O channel diagram 1-16
 - keyboard interface 1-38
 - power supply 1-21
 - speaker interface 1-20
 - specifications E-1
 - system board 1-3

T

- T/C (Terminal Count), I/O channel 1-19
- transmitter holding register 1-223

U

- underline (printer) 1-66
- unidirectional printer (printer) 1-71

V

- Vac output,
 - expansion unit 1-42
 - system unit 1-22
- Vdc output,
 - expansion unit 1-42
 - system unit 1-22
- vectors with special meaning 2-5
- vertical tab (printer) 1-64
- voltage interchange
 - asynchronous communications adapter 1-202

Numerics

- 1/8-inch line feeding (printer) 1-66
- 1920 bit-image graphics mode (printer) 1-71
- 480 bit-image graphics mode (printer) 1-69
- 6845,
 - (see color/graphics monitor adapter)
 - (see monochrome display adapter)
- 7/72-inch line feeding (printer) 1-66
- 8088,
 - (see Intel 8088 microprocessor)
- 8250,
 - (see asynchronous communications adapter)
- 8253-5,
 - (see synchronous data link control adapter)
- 8255A 1-9
- 8255A-5,
 - (see synchronous data link control adapter)
- 8273,
 - (see synchronous data link control adapter)
- 960 bit-image graphics mode (printer) 1-70
- 960 bit-image graphics mode normal speed (printer) 1-71



IBM United Kingdom International Products Limited,
PO Box 41, North Harbour, Portsmouth PO6 3AU, England.