



*Personal Computer XT
Hardware Reference
Library*

Technical Reference

LIMITED WARRANTY

The International Business Machines Corporation warrants this IBM Personal Computer Product to be in good working order for a period of 90 days from the date of purchase from IBM or an authorized IBM Personal Computer dealer. Should this Product fail to be in good working order at any time during this 90-day warranty period, IBM will, at its option, repair or replace this Product at no additional charge except as set forth below. Repair parts and replacement Products will be furnished on an exchange basis and will be either reconditioned or new. All replaced parts and Products become the property of IBM. This limited warranty does not include service to repair damage to the Product resulting from accident, disaster, misuse, abuse, or non-IBM modification of the Product.

Limited Warranty service may be obtained by delivering the Product during the 90-day warranty period to an authorized IBM Personal Computer dealer or IBM Service Center and providing proof of purchase date. If this Product is delivered by mail, you agree to insure the Product or assume the risk of loss or damage in transit, to prepay shipping charges to the warranty service location and to use the original shipping container or equivalent. Contact an authorized IBM Personal Computer dealer or write to IBM Personal Computer, Sales and Service, P.O. Box 1328-W, Boca Raton, Florida 33432, for further information.

ALL EXPRESS AND IMPLIED WARRANTIES FOR THIS PRODUCT INCLUDING THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO A PERIOD OF 90 DAYS FROM THE DATE OF PURCHASE, AND NO WARRANTIES, WHETHER EXPRESS OR IMPLIED, WILL APPLY AFTER THIS PERIOD. SOME STATES DO NOT ALLOW LIMITATIONS ON HOW LONG AN IMPLIED WARRANTY LASTS, SO THE ABOVE LIMITATIONS MAY NOT APPLY TO YOU.

IF THIS PRODUCT IS NOT IN GOOD WORKING ORDER AS WARRANTED ABOVE, YOUR SOLE REMEDY SHALL BE REPAIR OR REPLACEMENT AS PROVIDED ABOVE. IN NO EVENT WILL IBM BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF OR INABILITY TO USE SUCH PRODUCT, EVEN IF IBM OR AN AUTHORIZED IBM PERSONAL COMPUTER DEALER HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY.

SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES FOR CONSUMER PRODUCTS, SO THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO YOU.

THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH MAY VARY FROM STATE TO STATE.



*Personal Computer XT
Hardware Reference
Library*

Technical Reference

FEDERAL COMMUNICATIONS COMMISSION RADIO FREQUENCY INTERFERENCE STATEMENT

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

Notice: As sold by the manufacturer, the IBM Prototype Card does not require certification under the FCC's rules for Class B devices. The user is responsible for any interference to radio or TV reception which may be caused by a user-modified prototype card.

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

Revised Edition (April 1983)

Changes are periodically made to the information herein; these changes will be incorporated in new editions of this publication.

Products are not stocked at the address below. Requests for copies of this product and for technical information about the system should be made to your authorized IBM Personal Computer dealer.

A Reader's Comment Form is provided at the back of this publication. If this form has been removed, address comments to: IBM Corp., Personal Computer, P.O. Box 1328-C, Boca Raton, Florida 33432. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligations whatever.

© Copyright International Business Machines Corporation, 1981, 1982, 1983

PREFACE

The IBM Personal Computer XT Technical Reference manual describes the hardware design and provides interface information for the IBM Personal Computer XT. 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 XT, 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 seven 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
- Appendix G: Switch Settings

A glossary and bibliography are included.

Prerequisite Publication:

Guide to Operations for the IBM Personal Computer XT
Part Number 6936810

Suggested Reading:

BASIC for the IBM Personal Computer
Part Number 6025010

Disk Operating System (DOS) for the IBM Personal Computer
Part Number 6024061

Hardware Maintenance and Service for the IBM Personal
Computer XT
Part Number 6936809

MACRO Assembler for the IBM Personal Computer
Part Number 6024002

Related publications are listed in the bibliography.

TABLE OF CONTENTS

Section 1: Hardware

IBM Personal Computer XT System Unit	1-3
IBM Personal Computer Math Coprocessor	1-25
IBM Keyboard	1-65
IBM Expansion Unit	1-71
IBM 80 CPS Printers	1-81
IBM Printer Adapter	1-107
IBM Monochrome Display and Printer Adapter	1-113
IBM Monochrome Display	1-121
IBM Color/Graphics Display Adapter	1-123
IBM Color Display	1-149
IBM 5-¼" Diskette Drive Adapter	1-151
IBM 5-¼" Diskette Drive	1-175
Diskettes	1-177
IBM Fixed Disk Drive Adapter	1-179
IBM 10MB Fixed Disk Drive	1-195
IBM Memory Expansion Options	1-197
IBM Game Control Adapter	1-203
IBM Prototype Card	1-209
IBM Asynchronous Communications Adapter	1-215
IBM Binary Synchronous Communications Adapter	1-245
IBM Synchronous Data Link Control (SDLC) Communication Adapter	1-265
IBM Communications Adapter Cable	1-295

Section 2: ROM BIOS and System Usage

ROM BIOS	2-2
Keyboard Encoding and Usage	2-11

Appendix A: ROM BIOS Listings

A-1

System BIOS	A-2
Fixed Disk BIOS	A-85

Appendix B: 8088 Assembly Instruction

Set Reference	B-1
---------------------	-----

Appendix C: Of Characters, Keystrokes, and Colors	C-1
Appendix D: Logic Diagrams	D-1
System Board	D-2
Type 1 Keyboard	D-12
Type 2 Keyboard	D-14
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
5-¼ Inch Diskette Drive Adapter	D-45
5-¼ Inch Diskette Drive – Type 1	D-49
5-¼ Inch Diskette Drive – Type 2	D-52
Fixed Disk Drive Adapter	D-54
Fixed Disk Drive – Type 1	D-60
Fixed Disk Drive – Type 2	D-63
32K Memory Expansion Option	D-66
64K Memory Expansion Option	D-69
64/256K Memory Expansion Option	D-72
Game Control Adapter	D-76
Prototype Card	D-77
Asynchronous Communications Adapter	D-78
Binary Synchronous Communications Adapter	D-79
SDLC Communications Adapter	D-81
Appendix E: Specifications	E-1
Appendix F: Communications	F-1
Appendix G: Switch Settings	G-1
Glossary	H-1
Index	I-1

INDEX TAB LISTING

Section 1: Hardware

Hardware

Section 2: ROM BIOS and System Usage

BIOS

Appendix A: ROM BIOS Listings

Appendix A

Appendix B: 8088 Assembly Instruction Set Reference

Appendix B

Appendix C: Of Characters, Keystrokes, and Color

Appendix C

Appendix D: Logic Diagrams

Appendix D

Notes:

Appendix E: Specifications

Appendix E

Appendix F: Communications

Appendix F

Appendix G: Switch Settings

Appendix G

Glossary

Glossary

Bibliography

Bibliography

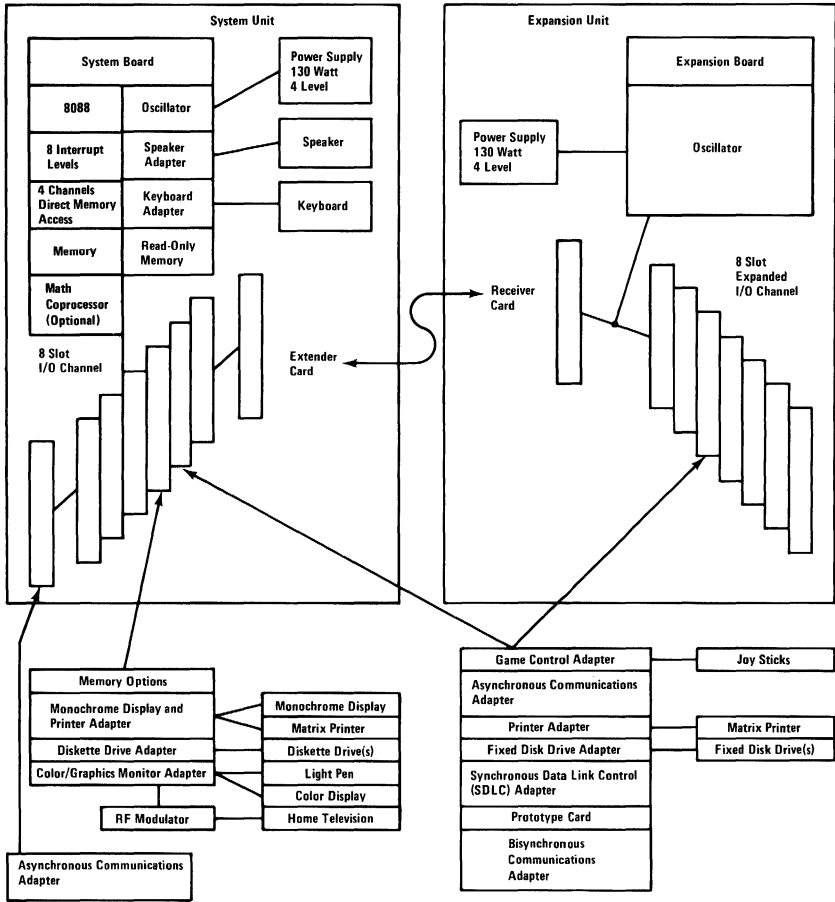
Index

Index

Notes:

SECTION 1: HARDWARE

- IBM Personal Computer XT System Unit 1-3
- IBM Personal Computer Math Coprocessor 1-25
- IBM Keyboard 1-65
- IBM Expansion Unit 1-71
- IBM 80 CPS Printers 1-81
- IBM Printer Adapter 1-107
- IBM Monochrome Display and Printer Adapter 1-113
- IBM Monochrome Display 1-121
- IBM Color/Graphics Display Adapter 1-123
- IBM Color Display 1-149
- IBM 5-1/4" Diskette Drive Adapter 1-151
- IBM 5-1/4" Diskette Drive 1-175
- Diskettes 1-177
- IBM Fixed Disk Drive Adapter 1-179
- IBM 10MB Fixed Disk Drive 1-195
- IBM Memory Expansion Options 1-197
- IBM Game Control Adapter 1-203
- IBM Prototype Card 1-209
- IBM Asynchronous Communications Adapter 1-215
- IBM Binary Synchronous Communications Adapter 1-245
- IBM Synchronous Data Link Control (SDLC)
Communication Adapter 1-265
- IBM Communications Adapter Cable 1-295



System Block Diagram

IBM Personal Computer XT System Unit

The system unit is the center of your IBM Personal Computer XT system. The system unit contains the system board, which features eight expansion slots, the 8088 microprocessor, 40K of ROM (includes BASIC), 128K of base R/W memory, and an audio speaker. A power supply is located in the system unit to supply dc voltages to the system board and internal drives.

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 and speaker. Eight 62-pin card edge-sockets are also mounted on the board. The I/O channel is bussed across these eight I/O slots. Slot J8 is slightly different from the others in that any card placed in it is expected to respond with a 'card selected' signal whenever the card is selected.

A dual-in-line package (DIP) switch (one eight-switch pack) is mounted on the board and can be read under program control. The DIP switch provides 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 64K by 8 of ROM or EPROM. Two module sockets are provided, each of which can accept a 32K or 8K device. One socket has 32K by 8 of ROM, the other 8K by 8 bytes. This ROM contains the 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 28-pin modules and has an access time and a cycle time of 250 ns each.

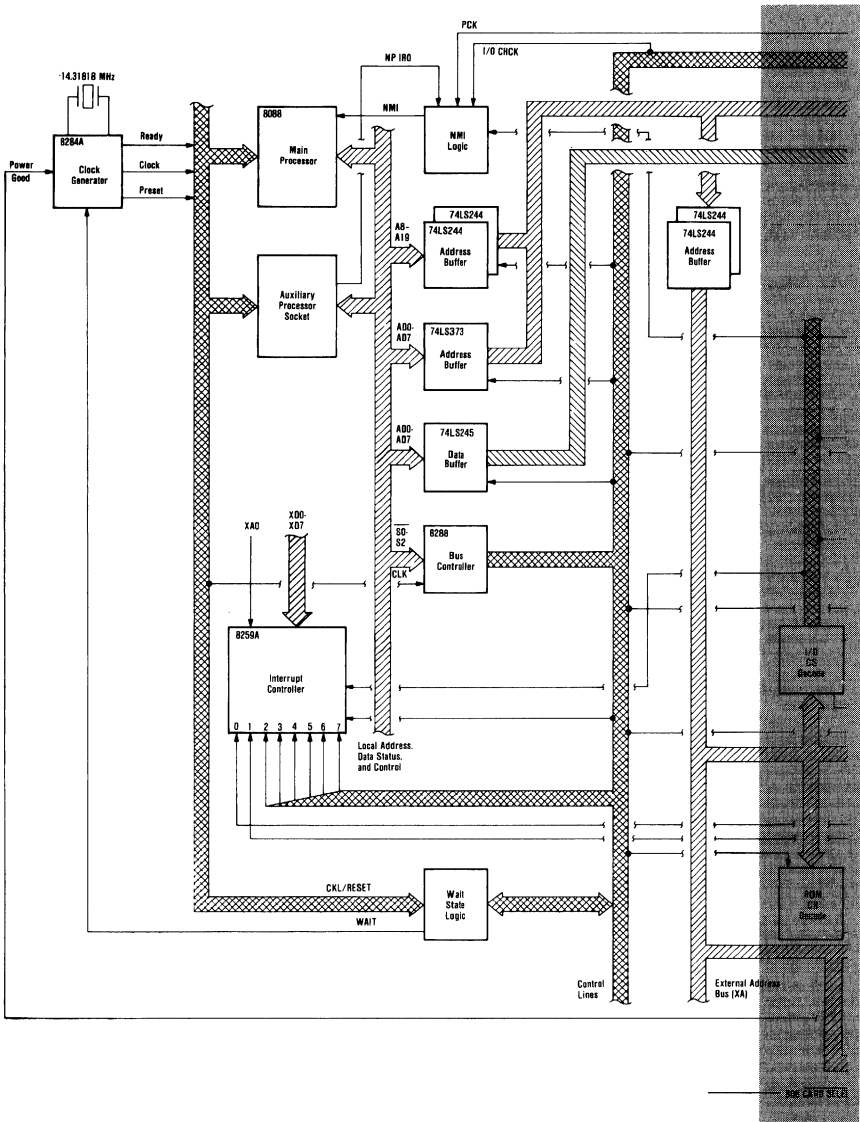
The system board also has from 128K by 9 to 256K by 9 of R/W memory. A minimum system would have 128K of memory, with module sockets for an additional 128K. Memory greater than the system board's maximum of 256K is obtained by adding memory cards in the expansion slots. The memory consists of dynamic 64K by 1 chips with an access time of 200 ns and a cycle time of 345 ns. All R/W memory is parity checked.

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.

The keyboard interface is a 5-pin DIN connector on the system board that extends through the rear panel of the system unit.

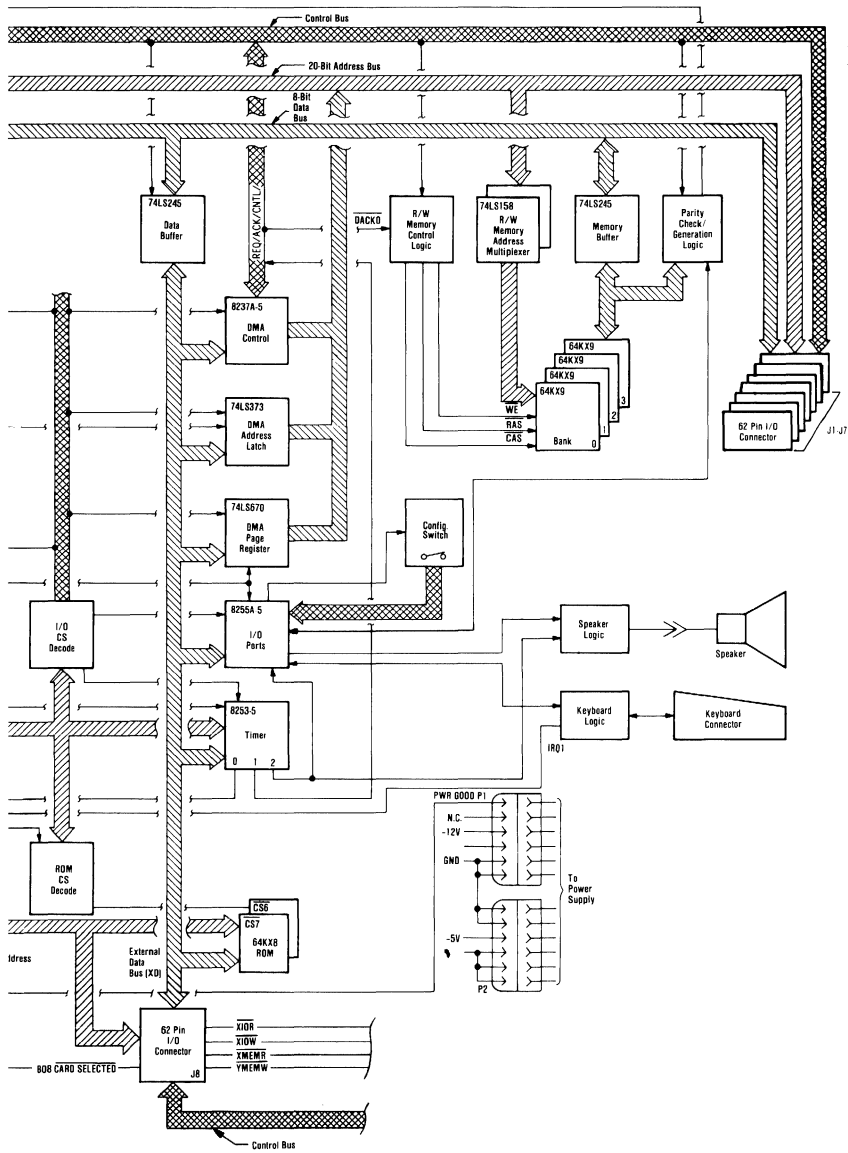
The system unit has a 2-1/4 inch audio speaker. The speaker's control circuits and driver are on the system board. The speaker connects through a 2-wire interface that attaches to a 3-pin connector on the system board.

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 Data Flow (Part 1 of 2)

1-6 System Unit



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
0AX*	NMI Mask Register
0CX	Reserved
0EX	Reserved
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	Printer
380-38C**	SDLC Communications
380-389**	Binary Synchronous Communications (Secondary)
3A0-3A9	Binary Synchronous Communications (Primary)
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 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)

** SDLC Communications and Secondary Binary Synchronous Communications cannot be used together because their hex addresses overlap.

I/O Address Map

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

8088 Hardware Interrupt Listing

Hex Port Number 0060	PA0 I N P U T	1	+Keyboard Scan Code	0	Or	Diagnostic Outputs	0																
		2		1			1																
		3		2			2																
		4		3			3																
		5		4			4																
		6		5			5																
		7		6			6																
0061	O U T P U T	1	+Timer 2 Gate Speaker																				
		2	+Speaker Data																				
		3	Spare																				
		4	Read High Switches Or Read Low Switches																				
		5	-Enable RAM Parity Check																				
		6	-Enable I/O Channel Check																				
		7	-Hold Keyboard Clock Low -(Enable Keyboard) Or + (Clear Keyboard)																				
0062	I N P U T	1	Loop on POST	Sw-1	Or	Display 0	**Sw-5																
		2	+Co-Processor Installed	Sw-2			Display 1	**Sw-6															
		3	+Planar RAM Size 0	*Sw-3			#5-1/4 Drives 0	***Sw-7															
		4	+Planar RAM Size 1	*Sw-4			#5-1/4 Drives 1	***Sw-8															
		5	Spare																				
		6	+Timer Channel 2 Out																				
		7	+I/O Channel Check +RAM Parity Check																				
0063	Command/Mode Register		Hex 99																				
	Mode Register Value		<table border="1"> <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																

*	Sw-4	Sw-3	Amount of Memory On System Board	
	0	0		64K
	0	1		128K
	1	0		192K
	1	1		256K
**	Sw-6	Sw-5	Display at Power-Up Mode	
	0	0		Reserved
	0	1		Color 40 X 25 (BW Mode)
	1	0		Color 80 X 25 (BW Mode)
	1	1		IBM Monochrome 80 X 25
***	Sw-8	Sw-7	Number of 5-1/4" Drives In System	
	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

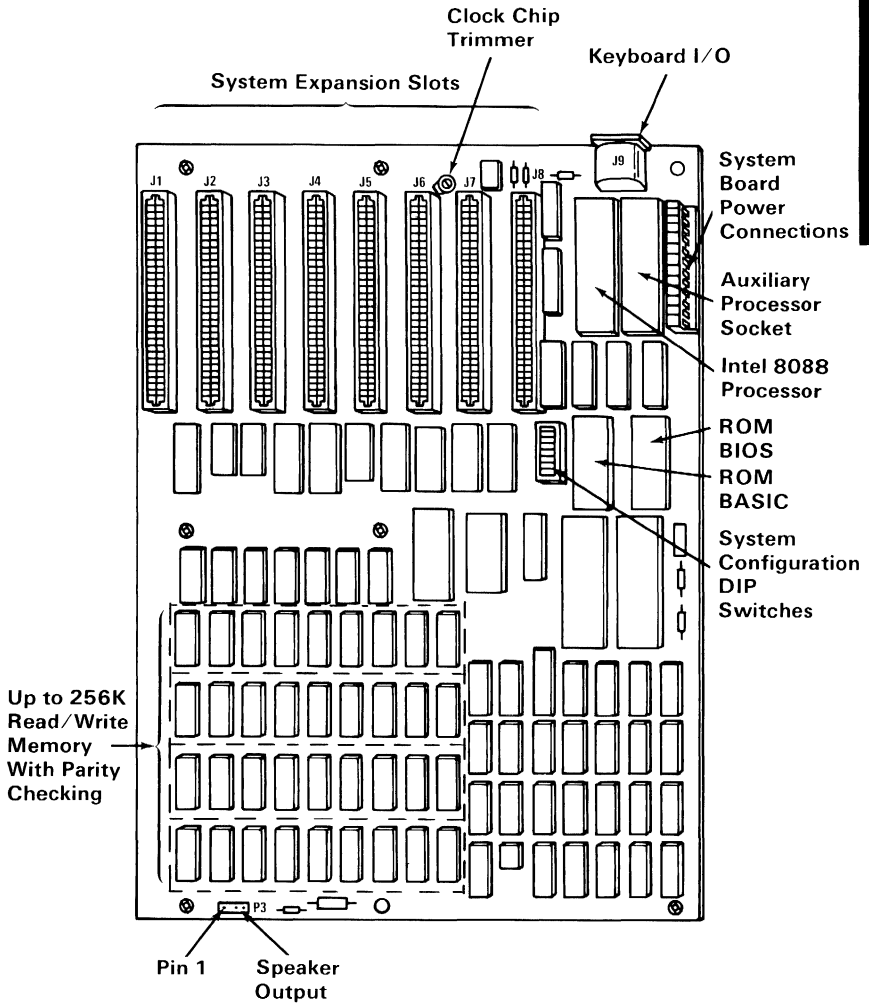
1-10 System Unit

Start Address		Function
Decimal	Hex	
0	00000	128-256K 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	384K R/W Memory Expansion 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 (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	
784K	C4000	
800K	C8000	Fixed Disk Control
816K	CC000	192K Read Only Memory Expansion and Control
832K	D0000	
848K	D4000	
864K	D8000	
880K	DC000	
896K	E0000	
912K	E4000	
928K	E8000	
944K	EC000	
960K	F0000	64K Base System ROM BIOS and BASIC
976K	F4000	
992K	F8000	
1008K	FC000	

System Memory Map (Part 2 of 2)



System Board Component Diagram

System Board Switch Settings

All system board switch settings for total system memory, number of diskette drives, and type of display are located in “Appendix G: 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.

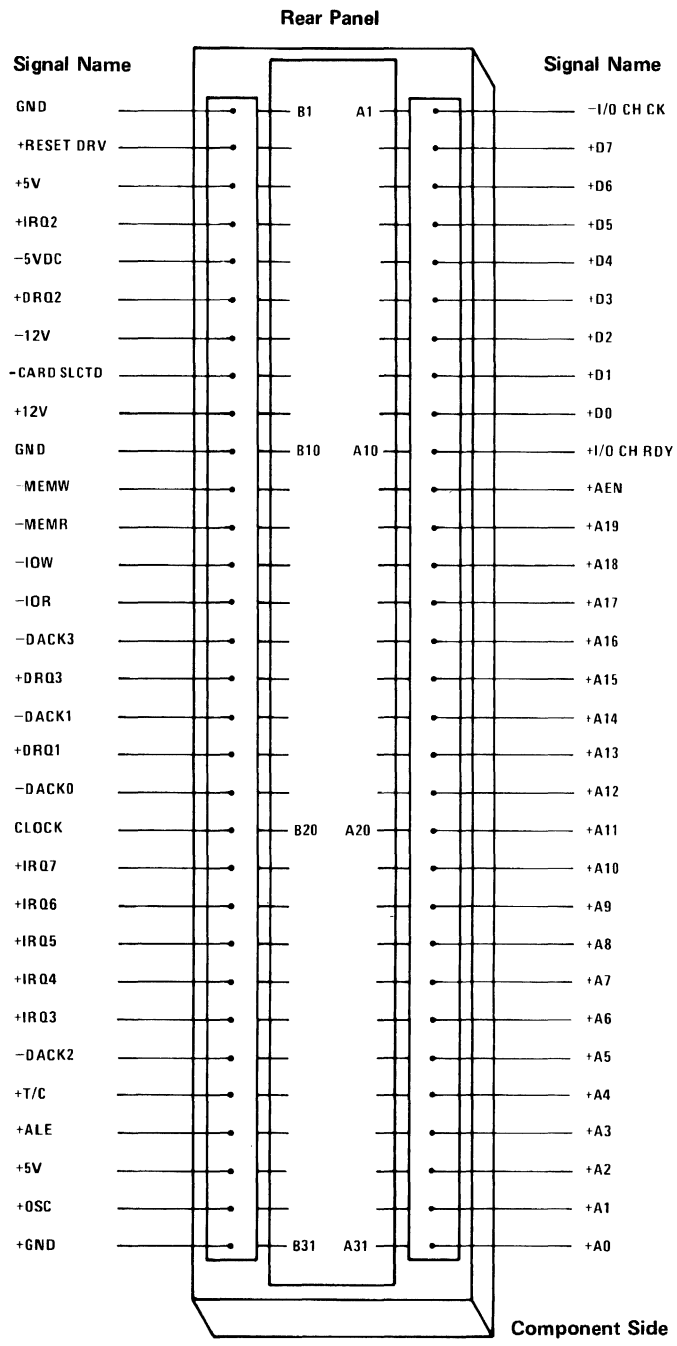
A 'ready' line is available on the I/O channel to allow operation with slow I/O or memory devices. If the channel's ready line is not activated by an addressed device, all processor-generated memory read and write cycles take four 210-ns clock or 840-ns/byte. All processor-generated I/O read and write 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 768 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 eight (J1 through J8) expansion slots, assuming two low-power Schottky (LS) loads per slot. The IBM I/O adapters typically use only one load.

Timing requirements on slot J8 are much stricter than those on slots J1 through J7. Slot J8 also requires the card to provide a signal designating when the card is selected. 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 XT 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.

Signal	I/O	Description
$\overline{\text{I/O CH CK}}$	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.
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.

Signal	I/O	Description
<u>MEMR</u>	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.
<u>MEMW</u>	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.
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- 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.

Signal	I/O	Description
<u>CARD SLCTD</u>	I	-Card Selected: This line is activated by cards in expansion slot J8. It signals the system board that the card has been selected and that appropriate drivers on the system board should be directed to either read from, or write to, expansion slot J8. Connectors J1 through J8 are tied together at this pin, but the system board does not use their signal. This line should be driven by an open collector device.

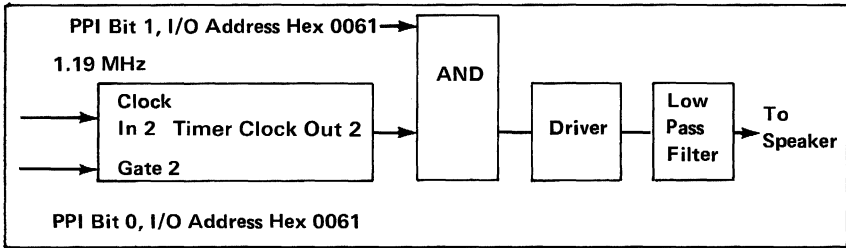
The following voltages are available on the system board I/O channel:

- +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

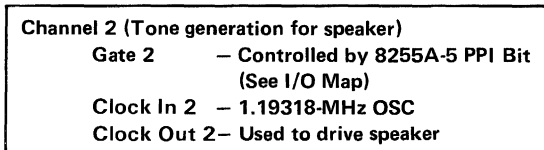
Speaker Interface

The sound system has a small, permanent-magnet, 2-¼ 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 dc power supply is a 130-watt, 4 voltage level switching regulator. It is integrated into the system unit and supplies power for the system unit, its options, and the keyboard. 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 system board takes approximately 2 to 4 A of +5 Vdc, thus allowing approximately 11 A of +5 Vdc for the adapters in the system expansion slots. The +12 Vdc power level is designed to power the internal 5-1/4 inch diskette drive and the 10 M fixed disk drive. The -5 Vdc level is used for analog circuits in the diskette adapter phase lock loop. The +12 Vdc and -12 Vdc are used for powering the EIA drivers for the communications adapters. All four power levels are bussed across the eight system expansion slots.

The IBM Monochrome Display has its own power supply, receiving its ac power from the system unit power system. The ac output for the display is switched on and off with the power switch and is a nonstandard connector, so only the IBM Monochrome Display can be connected.

Operating Characteristics

The power supply is located at the right rear area of the system unit. It supplies operating voltages to the system board, and IBM Monochrome Display, and provides two separate connections for power to the 5-1/4 inch diskette drive and the fixed disk drive. The nominal power requirements and output voltages are listed in the following tables:

Voltage @ 50/60 Hz		
Nominal Vac	Minimum Vac	Maximum Vac
110	90	137

Input Requirements

Frequency: 50/60 Hz +/- 3 Hz

Current: 4.1 A max @ 90 Vac

Voltage (Vdc)	Current (Amps)		Regulation (Tolerance)	
	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

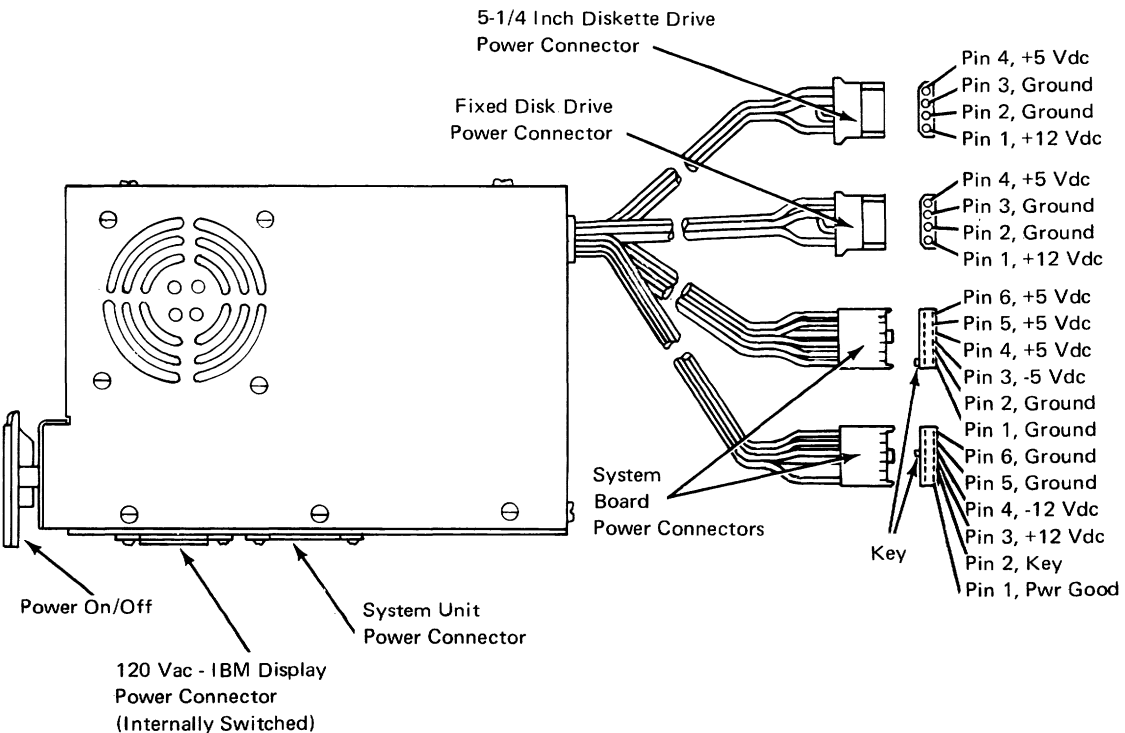
Vdc Output

Voltage (Vac)	Current (Amps)		Voltage Limits (Vac)	
	Minimum	Maximum	Minimum	Maximum
120	0.0	1.0	88	137

Vac Output

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:



Over-Voltage/Over-Current Protection

Voltage Nominal Vac	Type Protection	Rating Amps
110	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

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 (QSO 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).

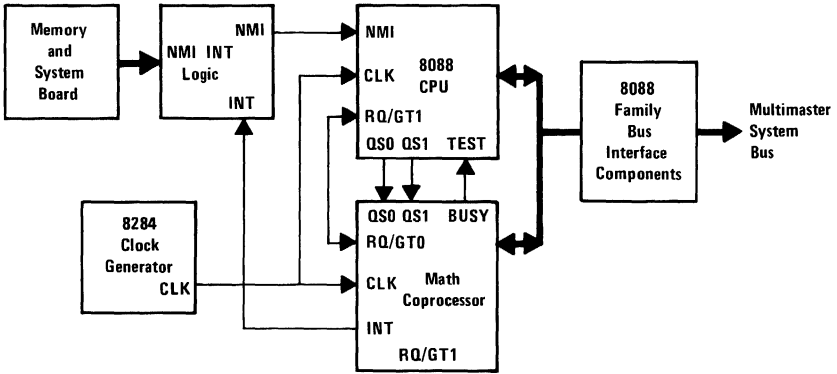
When an incorrect instruction is sent to the coprocessor (for example; divide by zero or load a full register), the coprocessor can signal the processor with an interrupt. There are three conditions that will disable the coprocessor interrupt to the processor:

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.

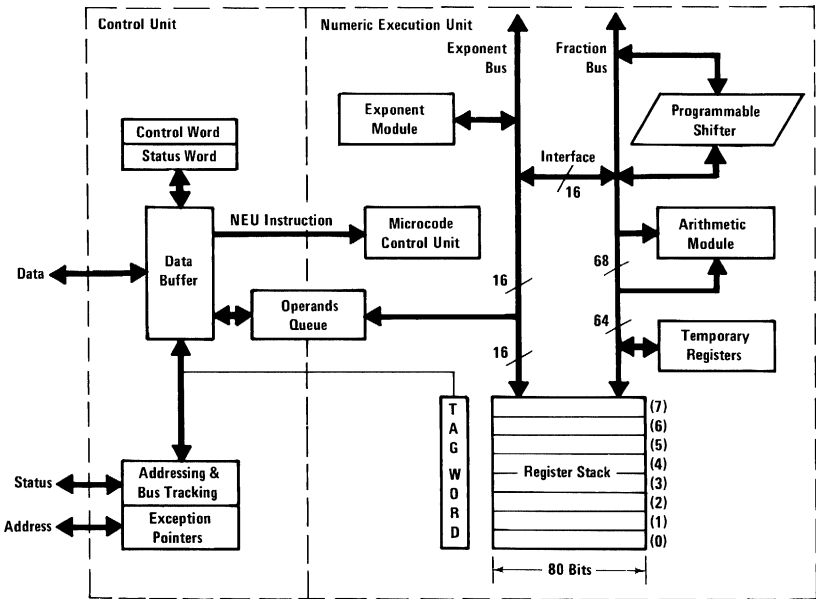
The NMI Mask REG and the coprocessors interrupt are tied to the NMI line through the NMI interrupt logic. Minor conversions of software designed for use with an 8087 must be made before existing software will be compatible with the IBM Personal Computer Math Coprocessor.



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

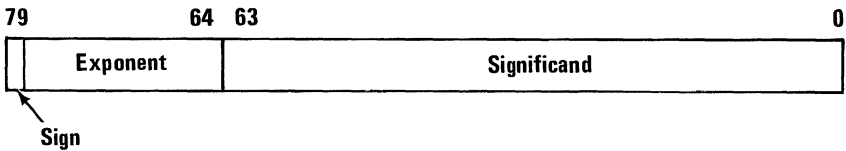
Register Stack

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 ith 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.



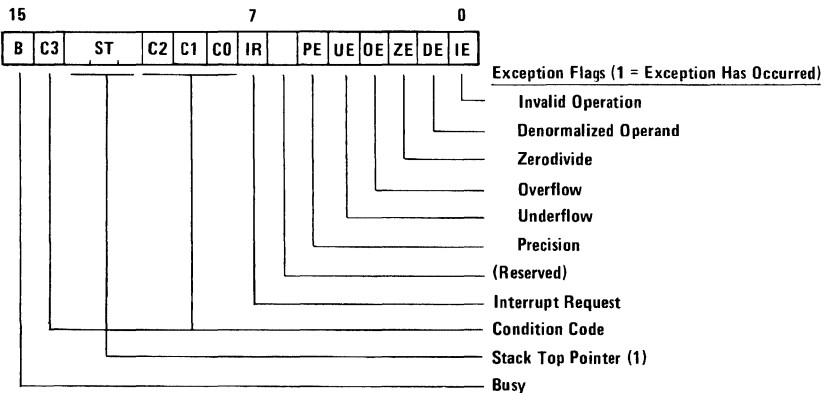
Register Structure

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.

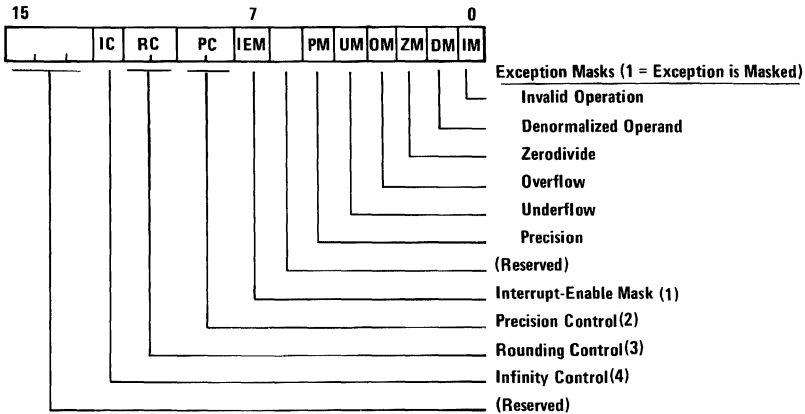


- (1) ST values:
- 000 = register 0 is stack top
 - 001 = register 1 is stack top
 - .
 - .
 - .
 - 111 = register 7 is stack top

Status Word Format

Control Word

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

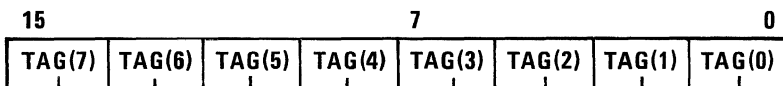


- (1) Interrupt-Enable Mask:
 - 0 = Interrupts Enabled
 - 1 = Interrupts Disabled (Masked)
- (2) Precision Control:
 - 00 = 24 bits
 - 01 = (reserved)
 - 10 = 53 bits
 - 11 = 64 bits
- (3) Rounding Control:
 - 00 = Round to Nearest or Even
 - 01 = Round Down (toward $-\infty$)
 - 10 = Round Up (toward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



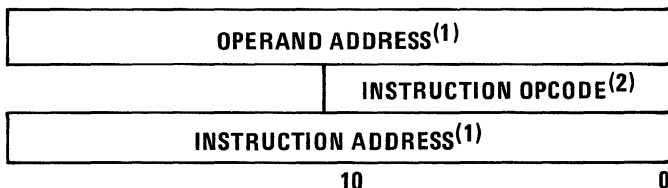
Tag values:

- 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)

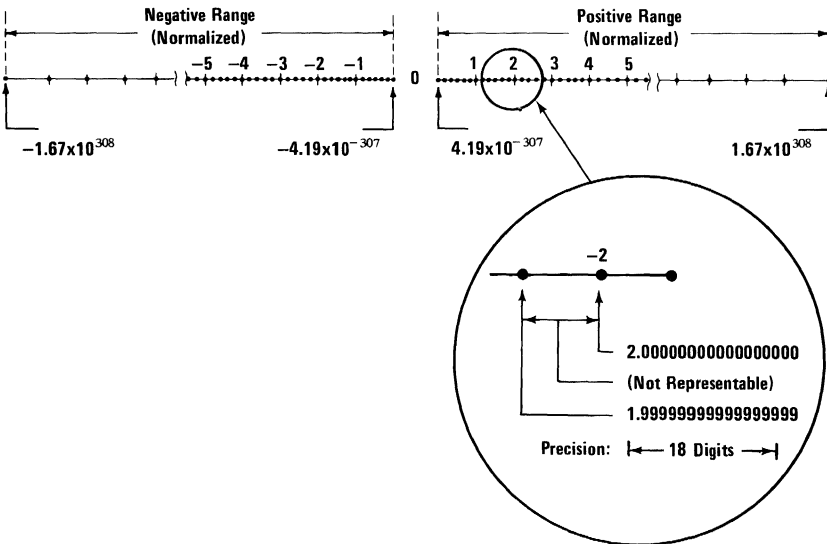
Exception Pointers Format

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 $\text{LOG}_e 2$ 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: I		
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: I		
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: I		
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: I		
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: I		
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		Transfers 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		Transfers 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		Transfers 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 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 $Y/X = \text{TAN}(\theta)$. 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.

$$\text{LOG}_n 2 \cdot \text{LOG}_2 X$$

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.

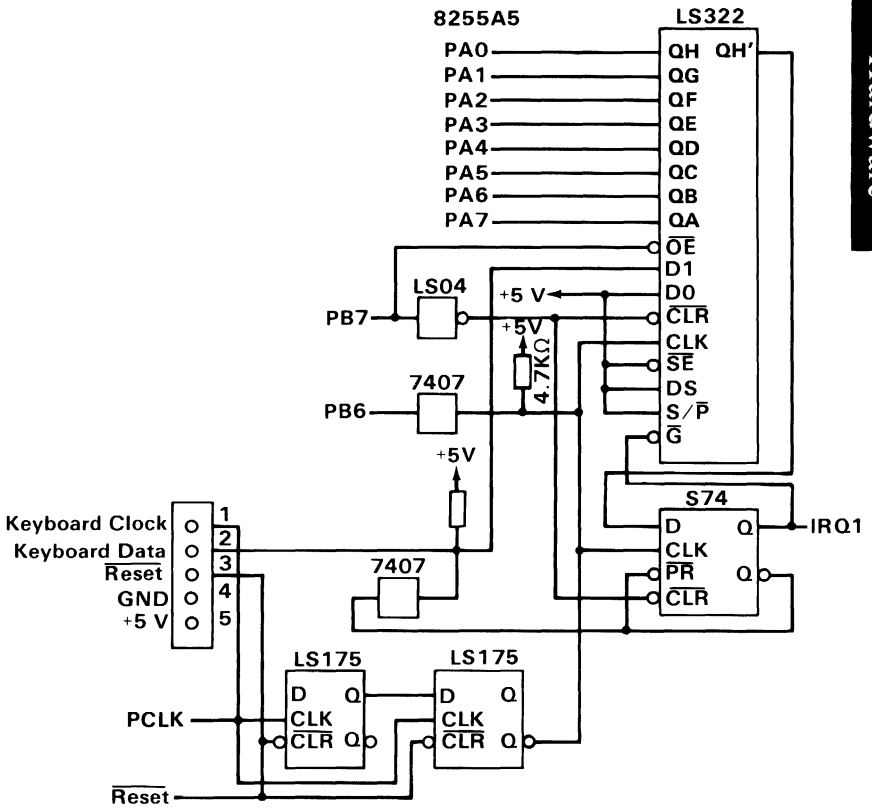
The keyboard uses a capacitive technology with a microcomputer (Intel 8048) performing the keyboard scan function. The keyboard has three tilt positions for operator comfort (5-, 7-, or 15-degree tilt orientations).

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.

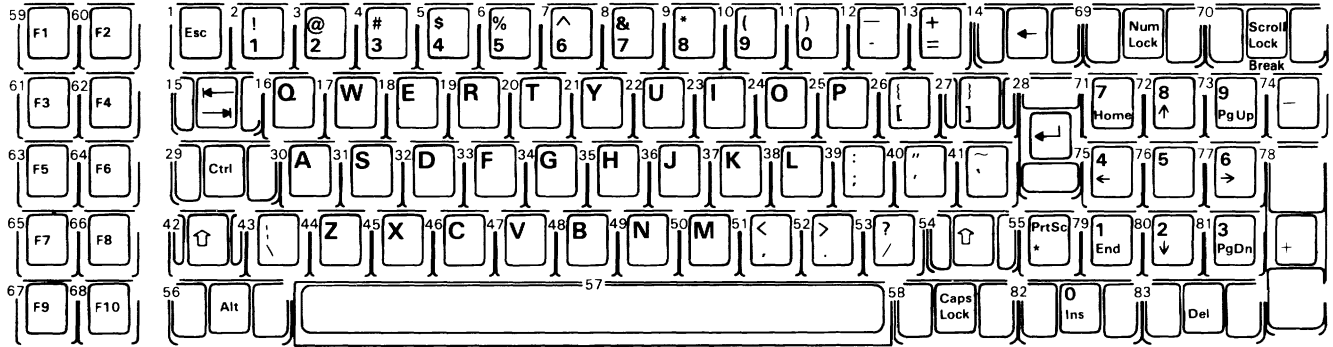
The microcomputer (Intel 8048) in the keyboard performs several functions, including a power-on self-test when requested by the system unit. This test checks the microcomputer ROM, tests memory, and checks for stuck keys. Additional functions are: keyboard scanning, buffering of up to 16 key scan codes, 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

Keyboard Diagram

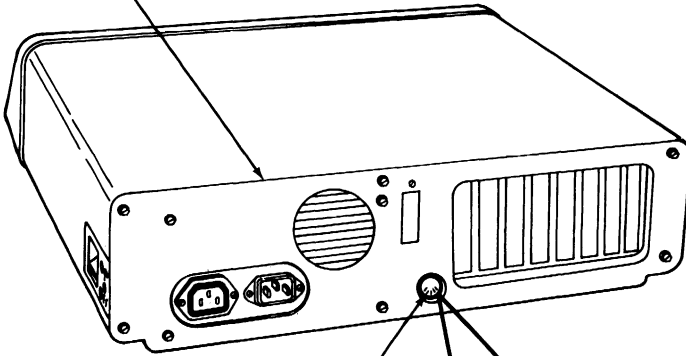


Note: Nomenclature is on both the top and front face of the keybutton as shown. The number to the upper left designates the button position.

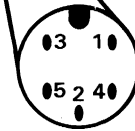
Key Position	Scan Code in Hex	Key Position	Scan Code in Hex
1	01	43	2B
2	02	44	2C
3	03	45	2D
4	04	46	2E
5	05	47	2F
6	06	48	30
7	07	49	31
8	08	50	32
9	09	51	33
10	0A	52	34
11	0B	53	35
12	0C	54	36
13	0D	55	37
14	0E	56	38
15	0F	57	39
16	10	58	3A
17	11	59	3B
18	12	60	3C
19	13	61	3D
20	14	62	3E
21	15	63	3F
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

Rear Panel



Keyboard Connector



5-Pin DIN Connector

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 XT 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.

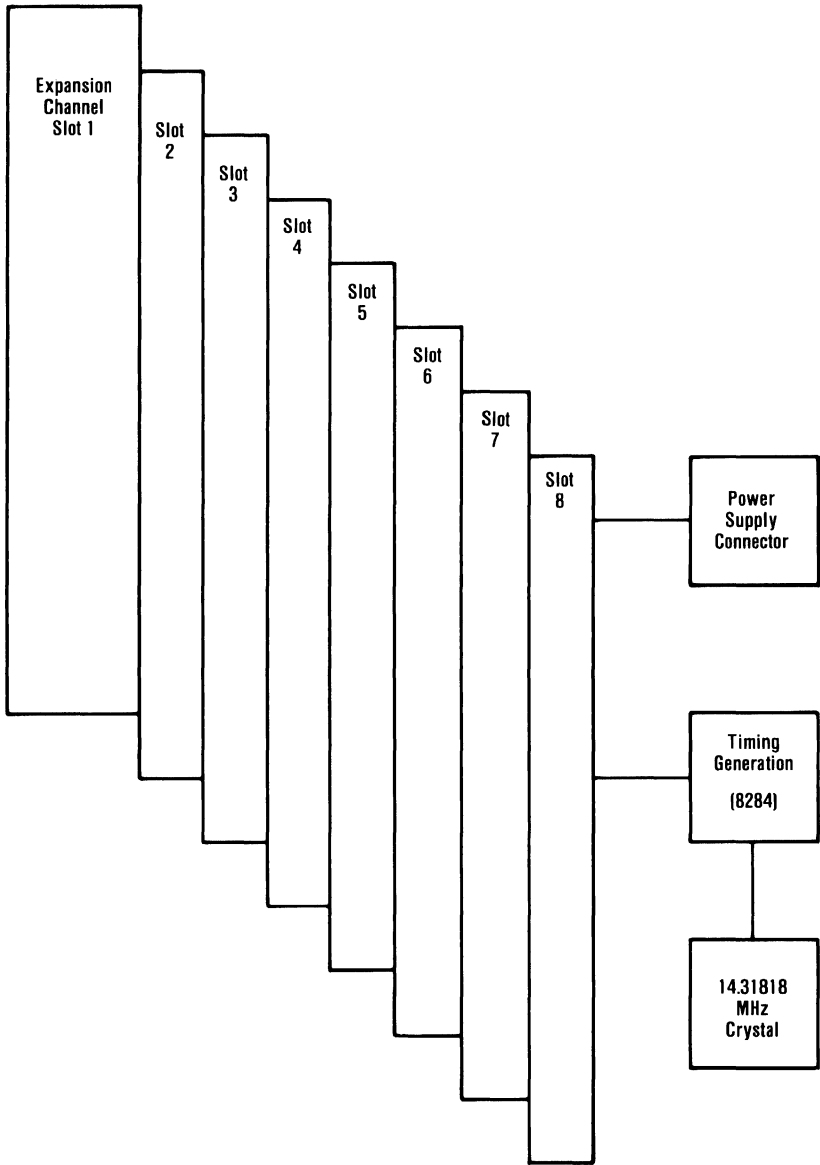
Power Supply

The expansion unit power supply provides +5, -5, +12, and -12 Vdc to the expansion board. The expansion unit power supply has the same specifications as the system unit power supply.

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.

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 settings are located in "Appendix G: Switch Settings." Switch positions 1 through 4 correspond to address bits hex A19 to hex A16, respectively.

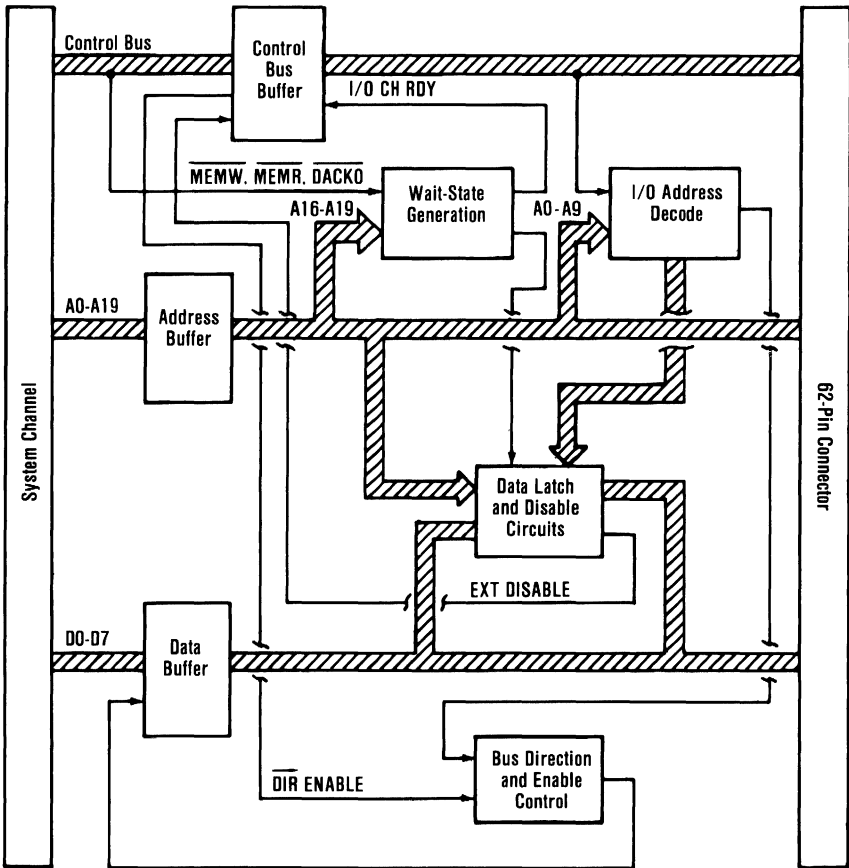
The switch settings determine which address segments have a wait state inserted during 'memory read' and 'memory write' operations. Wait states are required for any memory, including ROM on option adapters, in the expansion unit. Wait states are not inserted in the highest segment, hex addresses F0000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

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 (ED0-ED7)
Port 210	Read to verify expansion bus data (ED0-ED7)
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

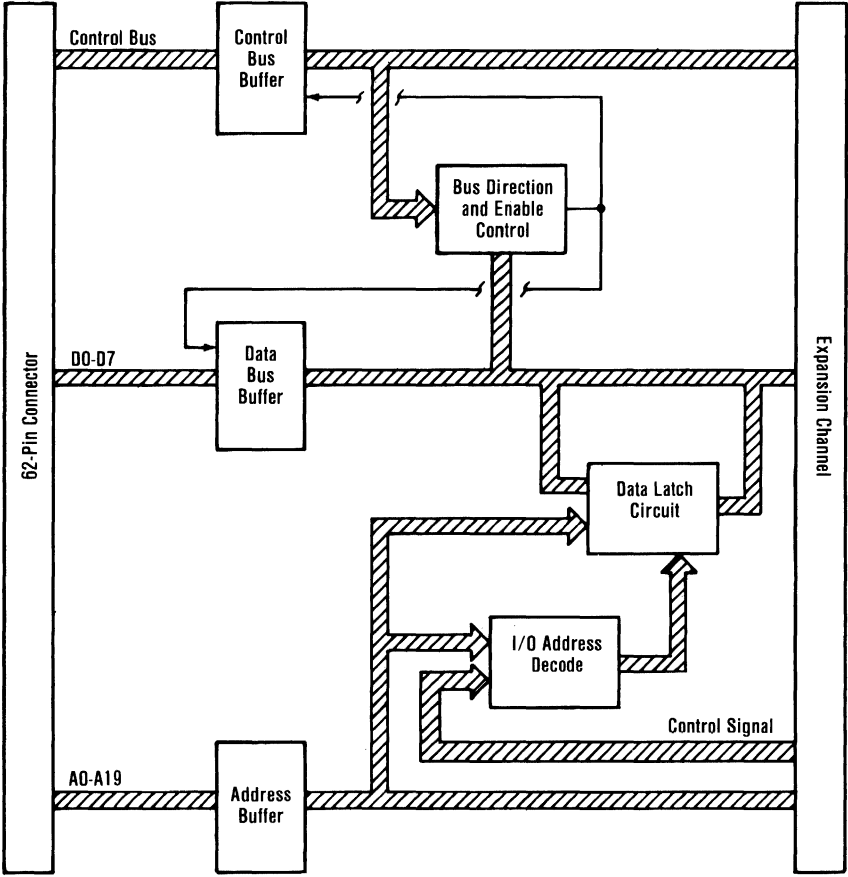
The receiver card is a four-plane card that fits in expansion slot 8 of the expansion unit. The receiver card redrives the I/O channel to provide sufficient power for additional options and to avoid capacitive effects. Directional control logic is contained on the receiver card to resolve contention 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 215	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 Printers

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:	1/16 inch (4.23 mm) or programmable	
(6)	Printing Characteristics		
	Matrix:	9 x 9	
	Character Set:	Full 96-character ASCII with descenders plus 9 international characters/symbols. See "Additional Printer Specifications"	
(7)	Graphic Character:		
	Printing Sizes	Characters per inch	Maximum characters per inch
	Normal:	10	80
	Double Width:	5	40
	Compressed:	16.5	132
	Double Width-Compressed:	8.25	66
(8)	Media Handling:		
	Paper Feed:	Adjustable sprocket pin feed	
	Paper Width Range:	4 inch (101.6 mm) to 10 inch (254 mm)	
	Copies:	One original plus two carbon copies (total thickness not to exceed 0.012 inch (0.3 mm)). Minimum paper thickness is 0.0025 inch (0.064 mm).	
	Paper Path:	Rear	
(9)	Interfaces:		
	Standard:	Parallel 8-bit Data and Control Lines	
(10)	Inked Ribbon:		
	Color:	Black	
	Type:	Cartridge	
	Life Expectancy:	3 million characters	
(11)	Environmental Conditions		
	Operating Temperature Range:	41 to 95°F (5 to 35°C)	
	Operating Humidity:	10 to 80% non-condensing	
(12)	Power Requirement:		
	Voltage:	120 Vac, 60 Hz	
	Current:	1 A maximum	
	Power Consumption:	100 VA maximum	
(13)	Physical Characteristics:		
	Height:	4.2 inches (107 mm)	
	Width:	14.7 inches (374 mm)	
	Depth:	12.0 inches (305 mm)	
	Weight:	12 pounds (5.5 kg)	

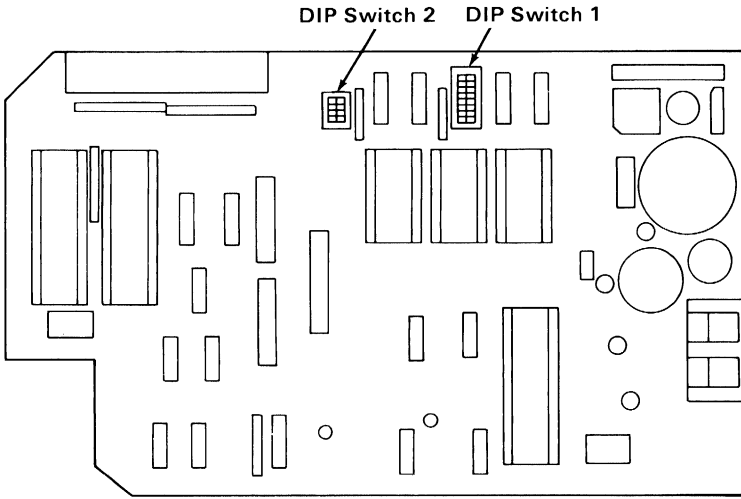
Printer Specifications

(6) Printing Characteristics:		
IBM 80 CPS Matrix Printer		
Graphics		64 block characters.
IBM 80 CPS Graphics Printer		
(6) Printing Characteristics:		
Extra Character Set.		Set 1
		Additional ASCII numbers 160 to 175 contain European characters. Numbers 176 to 223 contain graphic characters. Numbers 224 to 239 contain selected Greek characters. Numbers 240 to 255 contain math and extra symbols.
		Set 2
		The difference in set 2 are ASCII numbers 3, 4, 5, 6, and 21. ASCII numbers 128 to 175 contain European characters.
Graphics		There are 20 block characters and programmable graphics.
(7) Printing Sizes:		
	Characters	Maximum
	per inch	characters
		per line
Subscript:	10	80
Superscript:	10	80

Additional Printer Specifications

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	Delete Code	Invalid	Valid	On
1-6	Error	Sounds	Does Not Sound	On
1-7	Character Generator (Graphic Pattern Select)	N.A.	Graphic Patterns Select	Off
1-8	SLCT IN Signal Fixed Internally	Fixed	Not Fixed	On

Functions and Conditions of DIP Switch 1 (Matrix)

Switch Number	Function	On	Off	Factory-Set Condition
2-1	Not Applicable	—	—	On
2-2	Not Applicable	—	—	On
2-3	Auto Feed XT Signal	Fixed Internally	Not Fixed Internally	Off
2-4	Coding Table Select	N.A.	Standard	Off

Functions and Conditions of DIP Switch 2 (Matrix)

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	Off
1-8	SLCT IN Signal	Fixed Internally	Not Fixed Internally	On

Functions and Conditions of DIP Switch 1 (Graphics)

Switch Number	Function	On	Off	Factory-Set Condition
2-1	Form Length	12 Inches	11 Inches	Off
2-2	Line Spacing	1/8 Inch	1/6 Inch	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 (Graphics)

Parallel Interface Description

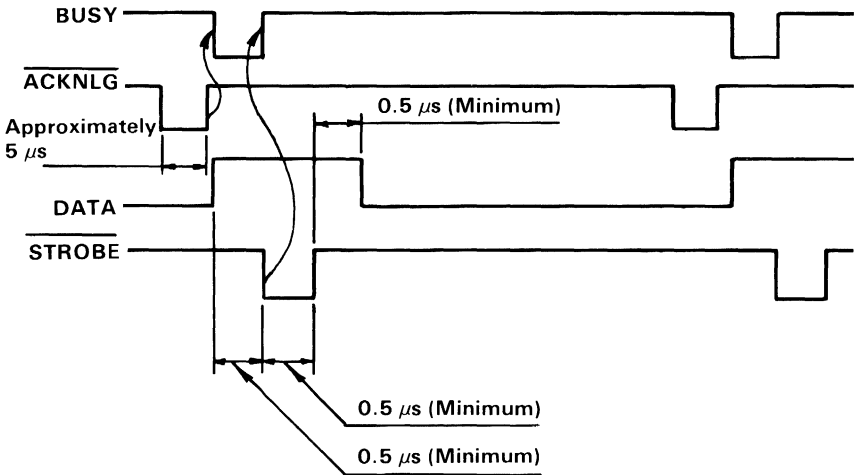
Specifications:

- Data transfer rate: 1000 cps (maximum)
- Synchronization: By externally-supplied $\overline{\text{STROBE}}$ pulses.
- Handshaking $\overline{\text{ACKNLG}}$ 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	—	<u>AUTO</u> 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	—	INT	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:**
- "Direction" refers to the direction of signal flow as viewed from the printer.
 - "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.
 - 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.
 - 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 Printers

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 IBM 80 CPS Matrix Printer cannot use the Subscript, Superscript, or Underline print modes. The Double Width print mode will affect the entire line with the matrix printer.

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 characters, controls, and graphics. You may want to keep them handy for future reference. The printer codes are listed in ASCII decimal numeric order (from NUL which is 0 to DEL which is 127). The examples given in the Printer Function descriptions are written in the BASIC language. The “input” description is given when more information is needed for programming considerations.

ASCII decimal values for the printer control codes can be found under “Printer Character Sets.”

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

Printer Code	Printer Function
NUL	<p>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: LPRINT CHR\$(0);</p>
BEL	<p>Bell Sounds the printer buzzer for 1 second. Example: LPRINT CHR\$(7);</p>
HT	<p>Horizontal Tab Tabs to the next horizontal tab stop. Tab stops are set with ESC D. No tab stops are set when the printer is powered on. (Graphics Printer sets a tab stop every 8 columns when powered on.) Example: LPRINT CHR\$(9);</p>
LF	<p>Line Feed Spaces the paper up one line. Line spacing is 1/6-inch unless reset by ESC A, ESC O, ESC 1, ESC 2 or ESC 3. Example: LPRINT CHR\$(10);</p>
VT	<p>Vertical Tab Spaces the paper to the next vertical tab position. (Graphics Printer does not allow vertical tabs to be set; therefore, the VT code is treated as LF.) Example: LPRINT CHR\$(11);</p>
FF	<p>Form Feed Advances the paper to the top of the next page. Note: The location of the paper, when the printer is powered on, determines the top of the page. The next top of page is 11 inches from that position. ESC C can be used to change the page length. Example: LPRINT CHR\$(12);</p>
CR	<p>Carriage Return Ends the line that the printer is on and prints the data remaining in the printer buffer. (No Line Feed operation takes place.) Note: IBM Personal Computer BASIC adds a Line Feed unless 128 is added [for example, CHR\$(141)]. Example: LPRINT CHR\$(13);</p>

Printer Code	Printer Function
SO	<p>Shift Out (Double Width) Changes the printer to the Double Width print mode. Note: A Carriage Return, Line Feed or DC4 cancels Double Width print mode. Example: LPRINT CHR\$(14);</p>
SI	<p>Shift In (Compressed) Changes the printer to the Compressed Character print mode. Example: LPRINT CHR\$(15);</p>
DC1	<p>Device Control 1 (Printer Selected) (Graphics Printer ignores DC1) Printer accepts data from the system unit. Printer DIP switch 1-8 must be set to the Off position. Example: LPRINT CHR\$(17);</p>
DC2	<p>Device Control 2 (Compressed Off) Stops printing in the Compressed print mode. Example: LPRINT CHR(18);</p>
DC3	<p>Device Control 3 (Printer Deselected) (Graphics Printer ignores DC3) Printer does not accept data from the system unit. The system unit must have the printer select line low, and DIP switch 1-8 must be in the Off position. Example: LPRINT CHR\$(19);</p>
DC4	<p>Device Control 4 (Double Width Off) Stops printing in the Double Width print mode. Example: LPRINT CHR\$(20);</p>
CAN	<p>Cancel Clears the printer buffer. Control codes, except SO, remain in effect. Example: LPRINT CHR\$(24);</p>
ESC	<p>Escape Lets the printer know that the next data sent is a printer command. (See the following list of commands.) Example: LPRINT CHR\$(27);</p>

Printer Code	Printer Function
ESC -	<p>Escape Minus (Underline) Format: ESC -;n; (Graphics Printer only) ESC - followed by a 1, prints all of the following data with an underline. ESC - followed by a 0 (zero), cancels the Underline print mode. Example: LPRINT CHR\$(27);CHR\$(45);CHR\$(1);</p>
ESC 0	<p>Escape Zero (1/8-Inch Line Feeding) Changes paper feeding to 1/8 inch. Example: LPRINT CHR\$(27);CHR\$(48);</p>
ESC 1	<p>Escape One (7/72-Inch Line Feeding) Changes paper feed to 7/72 inch. Example: LPRINT CHR\$(27);CHR\$(49);</p>
ESC 2	<p>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 returns to 1/6-inch. Example: LPRINT CHR\$(27);CHR\$(50);</p>
ESC 3	<p>Escape Three (Variable Line Feeding) Format: ESC 3;n; (Graphics Printer only) Changes the paper feeding to n/216-inch. The example below sets the paper feeding to 54/216 (1/4) inch. The value of n must be between 1 and 255. Example: LPRINT CHR\$(27);CHR\$(51);CHR\$(54);</p>
ESC 6	<p>Escape Six (Select Character Set 2) (Graphics Printer only) Selects character set 2. (See "Printer Character Set 2.") Example: LPRINT CHR\$(27);CHR\$(54);</p>
ESC 7	<p>Escape Seven (Select Character Set 1.) (Graphics Printer only) Selects character set 1. (See "Printer Character Set 1.") Character set 1 is selected when the printer is powered on or reset. Example: LPRINT CHR\$(27);CHR\$(55);</p>
ESC 8	<p>Escape Eight (Ignore Paper End) Allows the printer to print to the end of the paper. The printer ignores the Paper End switch. Example: LPRINT CHR\$(27);CHR\$(56);</p>

Printer Code	Printer Function
ESC 9	<p>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);</p>
ESC <	<p>Escape Less Than (Home Head) (Graphics Printer only) 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);</p>
ESC A	<p>Escape A (Sets Variable Line Feeding) Format: ESC A;n; Escape A sets the line-feed to n/72-inch. The example below tells the printer to set line feeding to 24/72-inch. ESC 2 must be sent to the printer before the line feeding will change. For example, ESC A;24 (text) ESC 2 (text). The text following ESC A;24 will space at the previously set line-feed increments. The text following ESC 2 will be printed with new line-feed increments of 24/72-inch. Any increment between 1/72 and 85/72 may be used. Example: LPRINT CHR\$(27);CHR\$(65);CHR\$(24);CHR\$(27);CHR\$(50);</p>
ESC B	<p>Escape B (Set Vertical Tabs) Format: ESC B;n₁;n₂;...n_k;NUL; (Graphics Printer ignores ESC B) Sets vertical tab stop positions. Up to 64 vertical tab stop positions are recognized by the printer. The n's, in the format above, are used to indicate tab stop positions. Tab stop numbers must be received in ascending numeric order. The tab stop numbers will not become valid until the NUL code is entered. Once vertical tab stops are established, they will be valid until new tab stops are specified. (If the printer is reset or powered Off, set tab stops are cleared.) If no tab stop is set, the Vertical Tab command behaves as a Line Feed command. ESC B followed only by NUL will cancel tab stops. The form length must be set by the ESC C command prior to setting tabs. Example: LPRINT CHR\$(27);CHR\$(66);CHR\$(10);CHR\$(20);CHR\$(40);CHR\$(0);</p>

Printer Code	Printer Function
ESC C	<p>Escape C (Set Lines per Page) Format: ESC C;n; Sets the page length. The ESC C command must have a value following it to specify the length of page desired. (Maximum form length for the printer is 127 lines.) 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);</p> <p>Escape C (Set Inches per Page) Format: ESC C;n;m; (Graphics Printer only) Escape C sets the length of the page in inches. 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);</p>
ESC D	<p>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 also be in ascending numeric order as shown. Tab stops can be set between 1 and 80. When in the Compressed print mode, tab stops can be set up to 132. The maximum number of tabs that can be set is 112. The Graphics Printer can have a maximum of 28 tab stops. The HT (CHR\$(9)) is used to execute a tab operation. Example: LPRINT CHR\$(27);CHR\$(68);CHR\$(10)CHR\$(20)CHR\$(40);CHR\$(0);</p>
ESC E	<p>Escape E (Emphasized) Changes the printer to the Emphasized print mode. The speed of the printer is reduced to half speed during the Emphasized print mode. Example: LPRINT CHR\$(27);CHR\$(69);</p>
ESC F	<p>Escape F (Emphasized Off) Stops printing in the Emphasized print mode. Example: LPRINT CHR\$(27);CHR\$(70);</p>
ESC G	<p>Escape G (Double Strike) Changes the printer to the Double Strike print mode. The paper is spaced 1/216 of an inch before the second pass of the print head. Example: LPRINT CHR\$(27);CHR\$(71);</p>

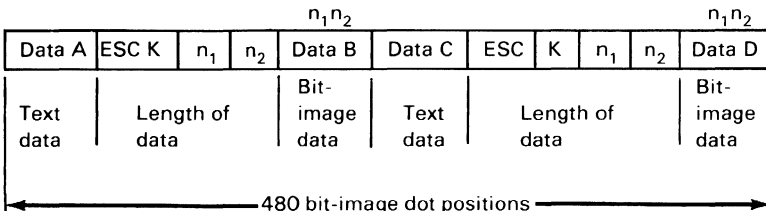
Printer Code	Printer Function																																																																																				
ESC H	<p>Escape H (Double Strike Off) Stops printing in the Double Strike mode. Example: LPRINT CHR\$(27);CHR\$(72);</p>																																																																																				
ESC J	<p>Escape J (Set Variable Line Feeding) Format: ESC J;n; (Graphics Printer only) When ESC J is sent to the printer, the paper will feed in increments of $n/216$ of an inch. The value of n must be between 1 and 255. The example below gives a line feed of 50/216-inch. ESC J is canceled after the line feed takes place. Example: LPRINT CHR\$(27);CHR\$(74);CHR\$(50);</p>																																																																																				
ESC K	<p>Escape K (480 Bit-Image Graphics Mode) Format ESC K;n₁;n₂;v₁;v₂;...v_k; (Graphics Printer only) Changes from the Text mode to the Bit-Image Graphics mode. n₁ and n₂ are one byte, which specify the number of bit-image data bytes to be transferred. v₁ through v_k are the bytes of the bit-image data. The number of bit-image data bytes (k) is equal to $n_1 + 256n_2$ and cannot exceed 480 bytes. 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.</p> <p>Note: Assign values to n₁ and n₂ as follows: n₁ represents values from 0 - 255. n₂ represents values from 0 - 1 x 256.</p> <p>MSB is most significant bit and LSB is least significant bit.</p> <div style="text-align: center;"> <p>n₂</p> <table border="1" style="margin: auto;"> <tr> <td style="width: 100px;">MSB</td> <td colspan="16"></td> <td style="width: 100px;">LSB</td> </tr> <tr> <td></td> <td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> <td></td> </tr> <tr> <td></td> <td>2</td><td>2</td><td>2</td><td>2</td><td>2</td><td>2</td><td>2</td><td>2</td> <td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td> <td></td> </tr> </table> <p>n₁</p> <table border="1" style="margin: auto;"> <tr> <td style="width: 100px;">MSB</td> <td colspan="8"></td> <td style="width: 100px;">LSB</td> </tr> <tr> <td></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></td> <td>2</td><td>2</td><td>2</td><td>2</td><td>2</td><td>2</td><td>2</td><td>2</td> </tr> </table> </div>	MSB																	LSB		15	14	13	12	11	10	9	8												2	2	2	2	2	2	2	2											MSB									LSB		7	6	5	4	3	2	1	0		2	2	2	2	2	2	2	2
MSB																	LSB																																																																				
	15	14	13	12	11	10	9	8																																																																													
	2	2	2	2	2	2	2	2																																																																													
MSB									LSB																																																																												
	7	6	5	4	3	2	1	0																																																																													
	2	2	2	2	2	2	2	2																																																																													

Data sent to the printer.

Text (20 characters)	ESC	K	n=360	Bit-image data	Next data
----------------------	-----	---	-------	----------------	-----------

In text mode, 20 characters in text mode correspond to 120 bit-image positions (20 x 6 = 120). The printable portion left in Bit-Image mode is 360 dot positions (480 - 120 = 360).

Data sent to the printer.



Example:

```

TYPE B:GRAPH.TXT
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$(02)+CHR$(04)+CHR$(08)
6 SLASH$=SLASH$+CHR$(16)+CHR$(32)+CHR$(64)+CHR$(128)+CHR$(0)
7 GAP$=CHR$(0)+CHR$(0)+CHR$(0)
8 NDOTS=480
9 'ESC K N1 N2
10 PRINT #1,CHR$(27);"K";CHR$(NDOTS MOD 256);CHR$(FIX (NDOTS/256));
11 ' SEND NDOTS NUMBER OF BIT IMAGE BYTES
12 FOR I=1 TO NDOTS/12 'NUMBER OF SLASHES TO PRINT USING
    GRAPHICS
13 PRINT #1,SLASH$;GAP$;
14 NEXT I
15 CLOSE
16 END
    
```

This example will give you a row of slashes printed in the 480 Bit-Image mode.

Printer Code	Printer Function
ESC L	<p>Escape L (960 Bit-Image Graphics Mode) Format: ESC L;n_1;n_2;v_1;v_2;...v_k, (Graphics Printer only) 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 Graphics mode, but can produce a denser graphic image. The number of bytes of bit-image Data (k) is $n_1 + 256n_2$ but cannot exceed 960. n_1 is in the range of 0 to 255.</p>
ESC N	<p>Escape N (Set Skip Perforation) Format ESC N;n; (Graphics Printer only) Sets the Skip Perforation function. The number following ESC N sets the value for the number of lines of Skip Perforation. The example shows a 12-line skip perforation. This will print 54 lines and feed the paper 12 lines. The value of n must be between 1 and 127. ESC N must be reset anytime the page length (ESC C) is changed. Example: CHR\$(27);CHR\$(78);CHR\$(12);</p>
ESC O	<p>Escape O (Cancel Skip Perforation) (Graphics Printer only) Cancels the Skip Perforation function. Example: LPRINT CHR\$(27);CHR\$(79);</p>
ESC S	<p>Escape S (Subscript/Superscript) Format: ESC S;n; (Graphics Printer only) 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);</p>
ESC T	<p>Escape T (Subscript/Superscript Off) (Graphics Printer only) The printer stops printing in the Subscript or Superscript print mode. Example: LPRINT CHR\$(27);CHR\$(84);</p>
ESC U	<p>Escape U (Unidirectional Printing) Format: ESC U;n; (Graphics Printer only) The printer will print 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) ensures a more accurate print-start position for better print quality. Example: LPRINT CHR\$(27);CHR\$(85);CHR\$(1);</p>

Printer Code	Printer Function
ESC W	<p>Escape W (Double Width) Format: ESC W;n; (Graphics Printer only) Changes the printer to the Double Width print mode when ESC W is followed by a 1. This mode is not canceled by a line-feed operation and must be canceled with ESC W followed by a 0 (zero). Example: LPRINT CHR\$(27);CHR\$(87);CHR\$(1);</p>
ESC Y	<p>Escape Y (960 Bit-Image Graphics Mode Normal Speed) Format: ESC Y n₁;n₂;v₁;v₂;...v_k; (Graphics Printer only) 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.</p>
ESC Z	<p>Escape Z (1920 Bit-Image Graphics Mode) Format: ESC Z;n₁;n₂;v₁;v₂;...v_k; (Graphics Printer only) Changes from the Text mode to the 1920 Bit-Image Graphics mode. The input is similar to the other Bit-Image Graphics modes. ESC Z can print only every third dot position.</p>
DEL	<p>Delete (Clear Printer Buffer) (Graphics Printer ignores DEL) Clears the printer buffer. Control codes, except SO, still remain in effect. DIP switch 1-5 must be in the Off position. Example: LPRINT CHR\$(127);</p>

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		DC1	DC2	DC3
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	{		}	~	DEL	NUL	

Matrix Printer Character Set (Part 1 of 2)

130	131	132	133	134	135	136	137	138	139
					BEL		HT	LF	
140	141	142	143	144	145	146	147	148	149
FF	CR	SO	SI		DC1	DC2	DC3	DC4	
150	151	152	153	154	155	156	157	158	159
		CAN			ESC				
160	161	162	163	164	165	166	167	168	169
170	171	172	173	174	175	176	177	178	179
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						

Matrix Printer Character Set (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	{		}	~		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
▒	▒	▒	▒	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 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	£	¥	℞	f
160	161	162	163	164	165	166	167	168	169
á	í	ó	ú	ñ	Ñ	á	ó	¿	⌞
170	171	172	173	174	175	176	177	178	179
⌞	½	¼	ï	<<	>>	■	■	■	■
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)

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.

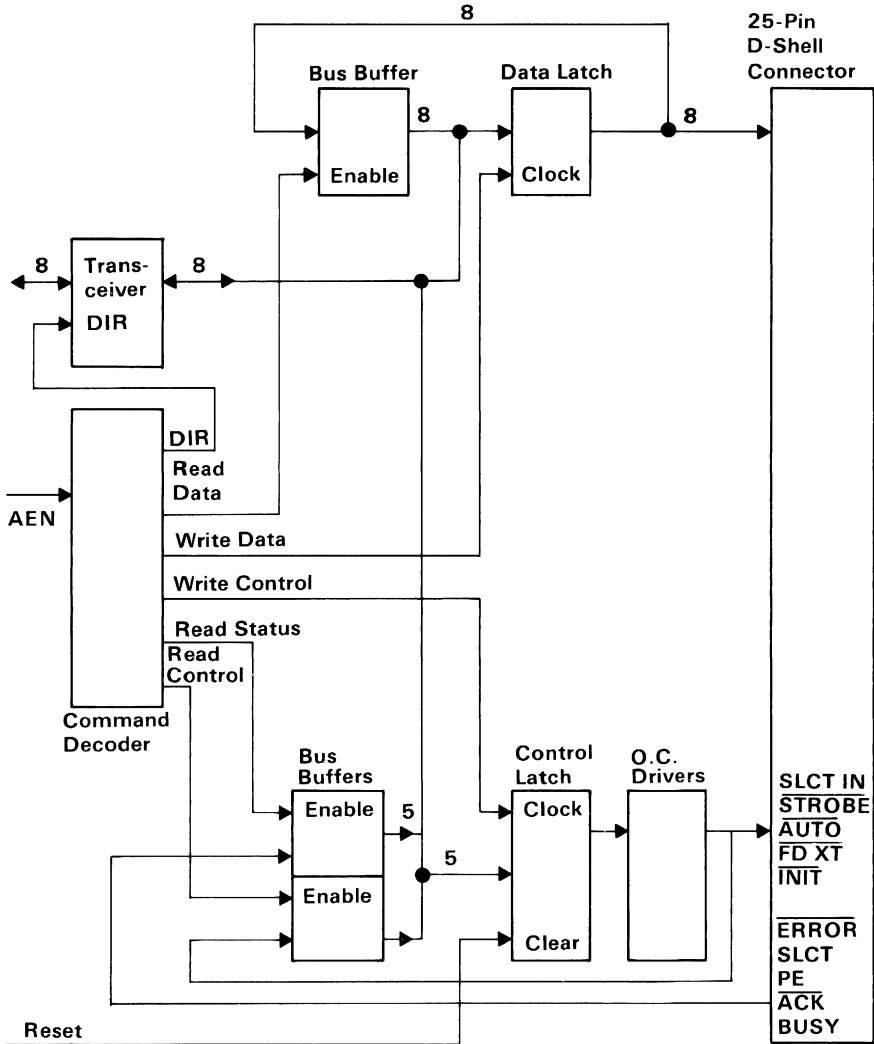
In addition, one input can also be used to create a processor interrupt. This interrupt can be enabled and disabled under program control. Reset from the power-on circuit is also ORed with a program output point, allowing a device to receive a power-on reset when the processor is reset.

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

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 IRQ Enable	$\overline{\text{Bit 3}}$ Pin 17	Bit 2 Pin 16	$\overline{\text{Bit 1}}$ Pin 14	$\overline{\text{Bit 0}}$ 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.

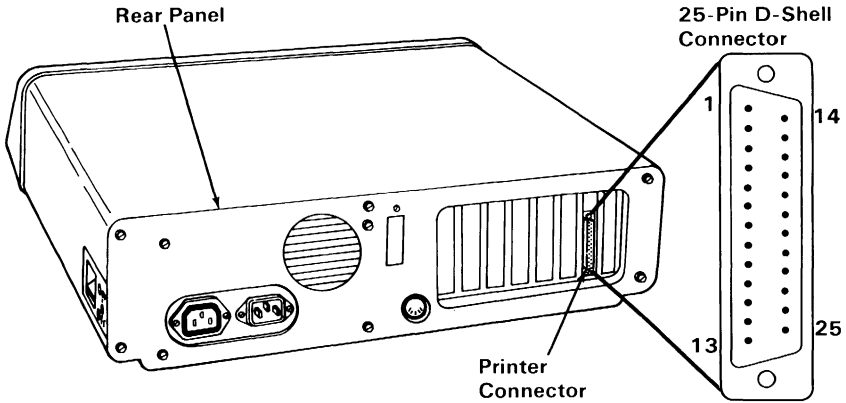
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Pin 11	Pin 10	Pin 12	Pin 13	Pin 15	-	-	-

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

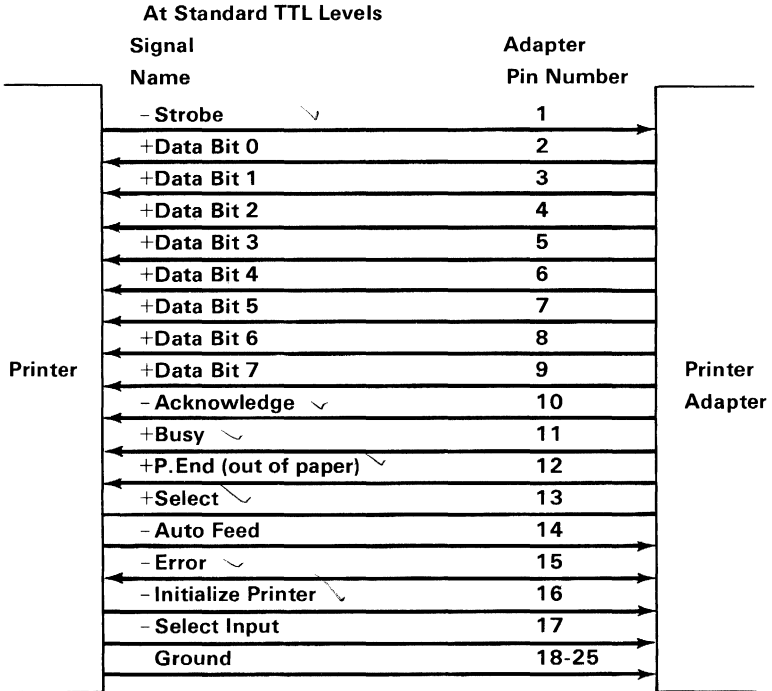
This instruction causes the data present on pins 1, 14, 15, 17, and the IRQ bit to read by the processor. In the absence of external drive applied to these pins, data read by the processor will exactly match data last written to hex 3BE in the same bit positions. Note that data bits 0-2 are not included. 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	<u>Pin 17</u>	Pin 16	<u>Pin 14</u>	<u>Pin 1</u>
			Por=0	Por=1	Por=0	Por=1	Por=1

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



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



Connector Specifications

IBM Monochrome Display and Printer Adapter

This chapter has two functions. The first is to provide the interface to the IBM Monochrome Display. The second provides a parallel interface for the IBM 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.

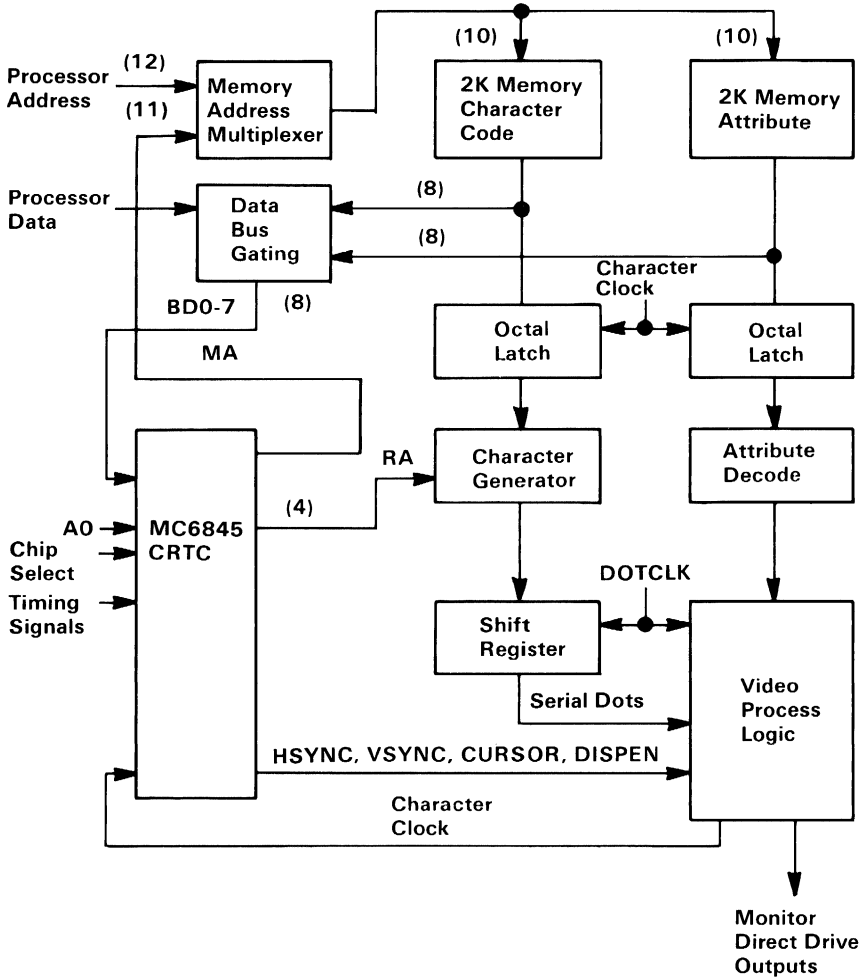
The monitor adapter supports 256 different character codes. An 8K-byte character generator contains the fonts for the character codes. The characters, values, and screen characteristics are given in “Appendix C: Of Characters, Keystrokes, and Color.”

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



IBM Monochrome Adapter Block Diagram

Programming Considerations

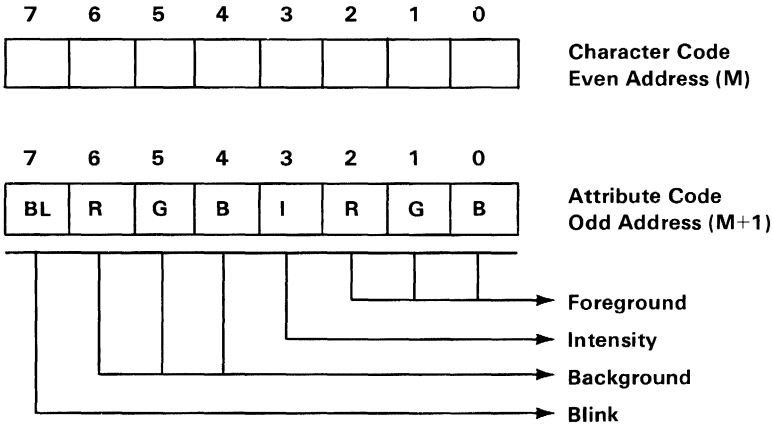
The following table summarizes the 6845 internal data registers, their functions, and their parameters. For the IBM Monochrome Display, the values must be programmed into the 6845 to ensure proper initialization of the device.

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	Interface 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				Foreground				Function
R	G	B	R	G	B			
0	0	0	0	0	0	0	Non-Display	
0	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.

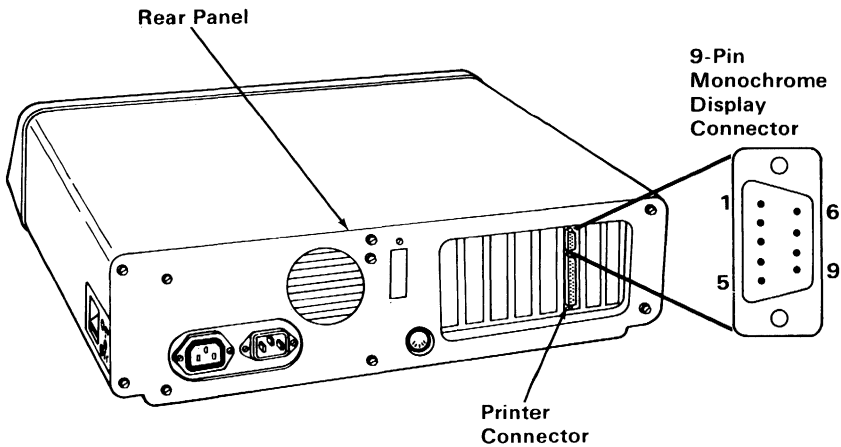
I/O Address and Bit Map

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

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 11-½ 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.

IBM Color/Graphics Monitor Adapter

The IBM Color/Graphics Monitor Adapter is designed to attach to the IBM Color Display, to a variety of television-frequency monitors, or to 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.

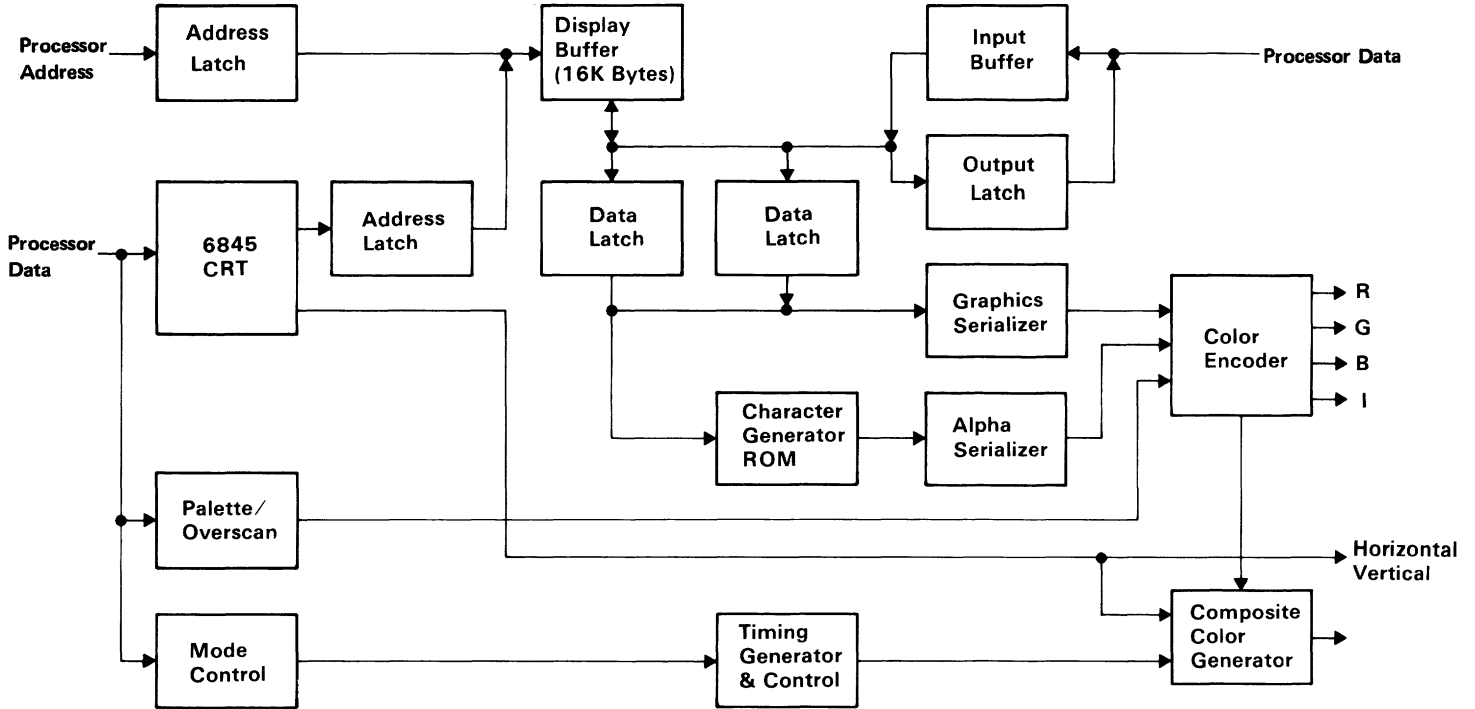
In the A/N mode, characters are formed from a ROM character generator. The character generator contains dot patterns for 256 different characters. The character set contains the following major groupings of characters:

- 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.

A block diagram of the color/graphics adapter is on the following page.



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 the 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

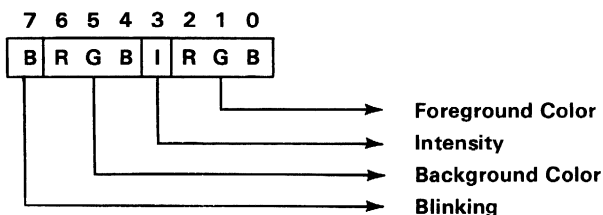
The functions of the attribute byte are defined by the following table:

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 mode 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	Background Color	Character Color	Background Color	Character Color
B R G B				I R G B							
FG Background				Foreground							
B	0	0	0		1	1	1	Black	White	Black	White
B	1	1	1		0	0	0	White	Black	White	Black
B	0	0	0		0	0	0	Black	Black	Black	Black
B	1	1	1		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.

	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 colors are:

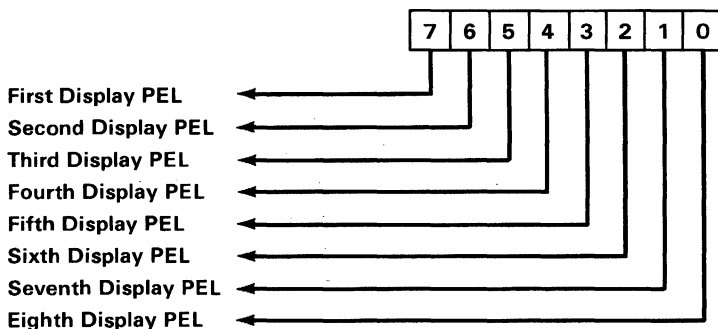
Color Set 1	Color Set 2
Color 1 is Green	Color 1 is Cyan
Color 2 is Red	Color 2 is Magenta
Color 3 is Brown	Color 3 is White

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

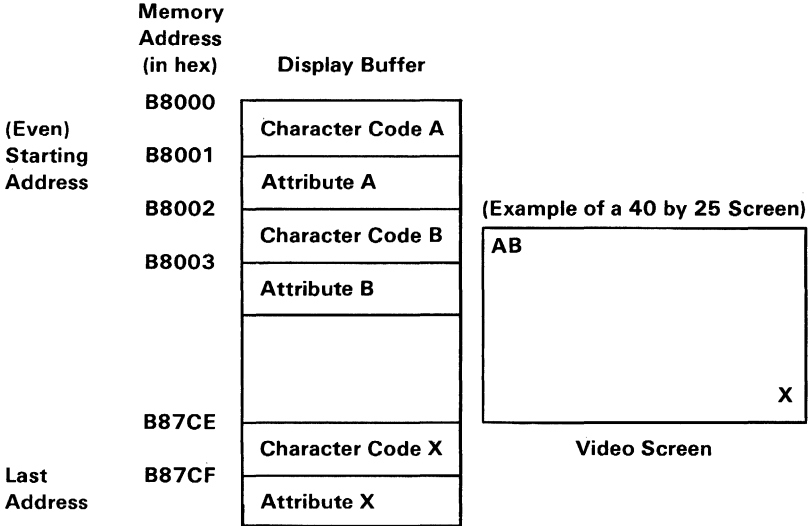
The high-resolution mode supports color monitors. This mode has the following features:

- 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

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.

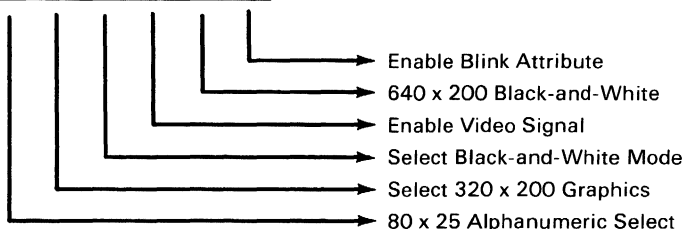
The following is a description of the register's functions:

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 by 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	40 x 25 Alphanumeric Black-and-White
0	0	0	1	0	1	40 x 25 Alphanumeric Color
1	0	1	1	0	1	80 x 25 Alphanumeric Black-and-White
1	0	0	1	0	1	80 x 25 Alphanumeric Color
0	1	1	1	0	z	320 x 200 Black-and-White Graphics
0	1	0	1	0	z	320 x 200 Color Graphics
0	1	1	1	1	z	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

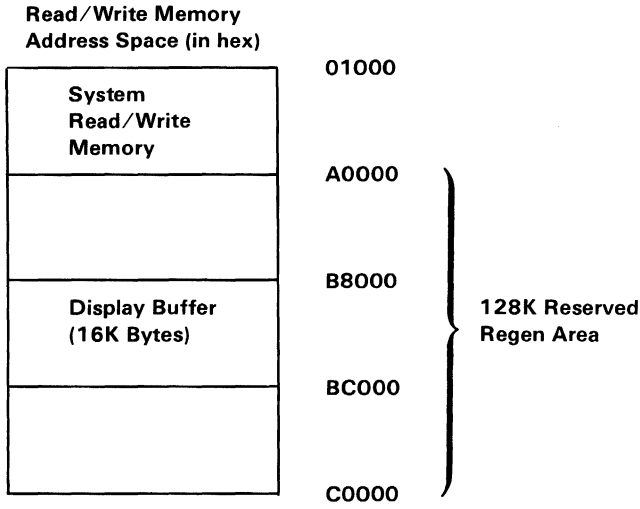
- Bit 0 This bit, when active, indicates that a regen buffer memory access can be made without interfering with the display.
- Bit 1 This bit, when active, indicates that a positive-going edge from the light-pen has set the light pen's trigger. This trigger is reset upon power-on and may also be cleared 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.

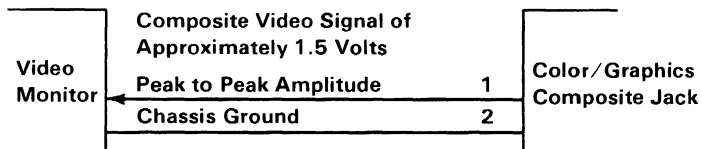
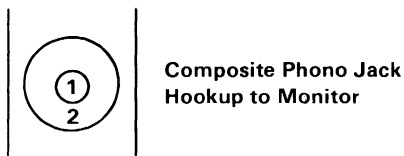
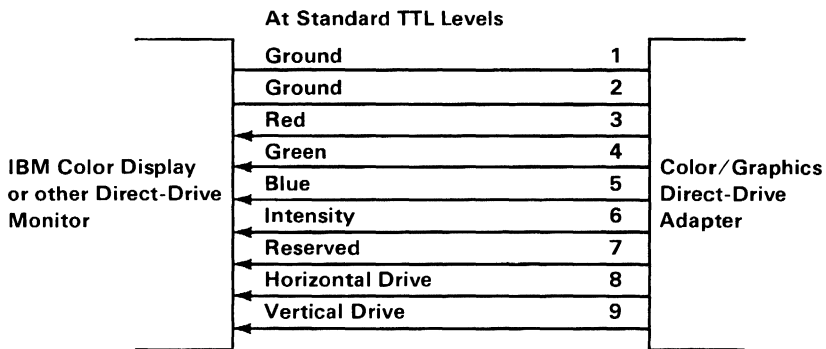
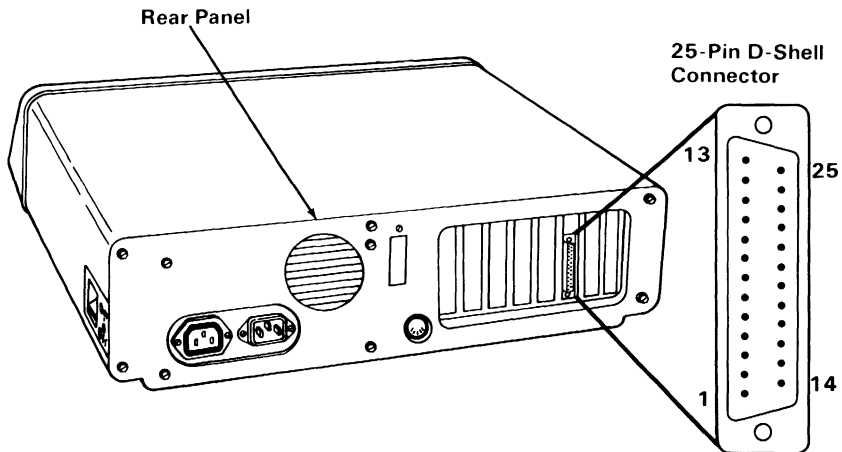
Sequence of Events for Changing Modes

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.

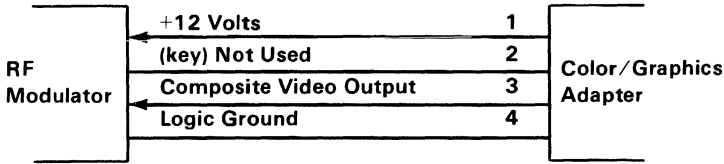
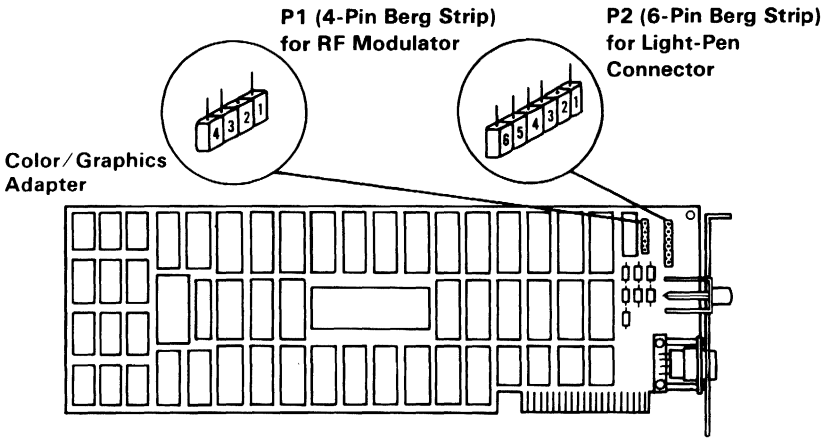
Memory Requirements

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.

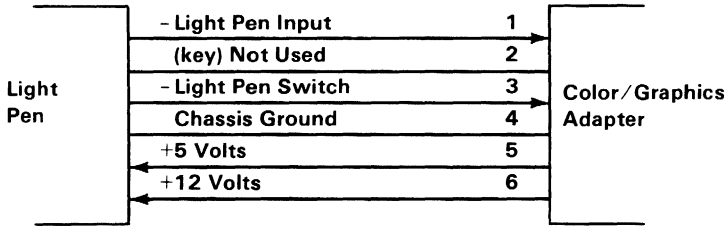




Connector Specifications (Part 1 of 2)



RF Modulator Interface



Light Pen Interface

Connector Specifications (Part 2 of 2)

Notes:

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

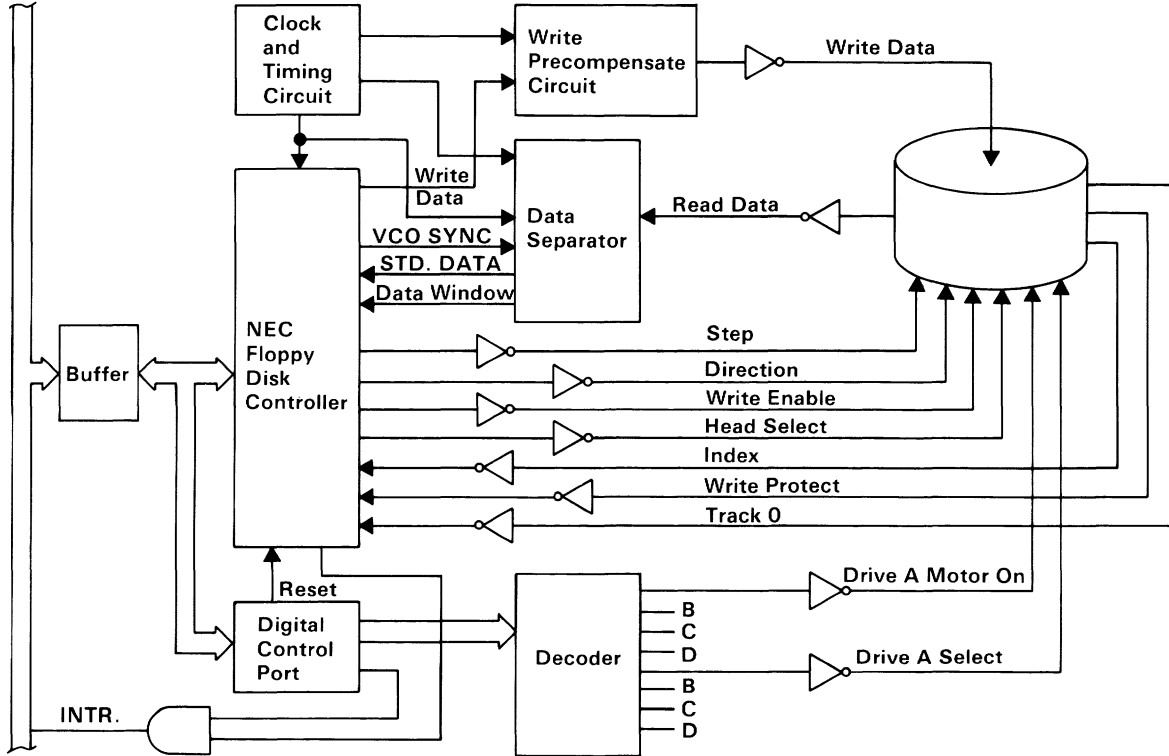
- Positive-level, TTL-compatibility, at a frequency of 15.75 kHz.

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 the 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 AO =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										Sector ID information prior to command execution.
	W								C		
	W								H		
	W								R		
	W								N		
	W								EOT		
W								GPL			
W								DTL			
Execution										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			
Command	W	MT	MF	SK	0	1	1	0	0	Command Codes	
	W	X	X	X	X	X	HD	US1	US0		
	W										Sector ID information prior to command execution.
	W								C		
	W								H		
	W								R		
	W								N		
	W								EOT		
W								GPL			
W								DTL			
Execution										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			

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										Sector ID information to command execution.
	W								C		
	W								H		
	W								R		
	W								N		
	W								EOT		
W								GPL			
W								DTL			
Execution										Data transfer between the main system and FDD.	
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	MT	MF	0	0	1	0	0	1	Command Codes	
	W	X	X	X	X	X	HD	US1	US0		
	W										Sector ID information prior to command execution.
	W								C		
	W								H		
	W								R		
	W								N		
	W								EOT		
W								GPL			
W								DTL			
Execution										Data transfer between FDD and main system.	
Result	R								ST 0	Status ID information after command execution. Sector 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	Read a Track								Command Codes
		0	MF	SK	0	0	0	1	0	
		X	X	X	X	X	HD	US1	US0	
						C				
						H				
						R				
						N				
						EOT				
Execution	W									Data transfer between the FDD and main system. FDC reads all of cylinder's contents from index hole to EOT.
Result	R									Status information after command execution. Sector ID information after command execution.
Command	W	Read ID								Command Codes
		0	MF	0	0	1	0	1	0	
Execution	W	X	X	X	X	X	HD	US1	US0	The first correct ID information on the cylinder is stored in data register.
Result	R									Status information after command execution. Sector ID information during execution phase.

Phase	R/W	Data Bus								Remarks	
		D7	D6	D5	D4	D3	D2	D1	D0		
Format a Track											
Command	W	0	MF	0	0	1	1	0	0	Command Codes	
	W	X	X	X	X	X	HD	US1	US0		
	W										Bytes/Sector
	W										Sector/Track
	W										Gap 3
Execution	W									filler byte.	
										FDC formats an entire cylinder.	
										Status information after command execution.	
										In this case, the ID information has no meaning.	
Result	R									ST 0	
	R									ST 1	
	R									ST 2	
	R									C	
	R									H	
	R									R	
	R									N	
Scan Equal											
Command	W	MT	MF	SK	1	0	0	0	1	Command Codes	
	W	X	X	X	X	X	HD	US1	US0		
	W										Sector ID information prior to command execution.
	W										
	W										
	W										
	W										
Execution	W										
Result	R									ST 0	
	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		
Scan Low or Equal											
Command	W	MT	MF	SK	1	1	0	0	1	Command Codes	
	W	X	X	X	X	X	HD	US1	US0		
	W										Sector ID information prior to command execution.
	W								C		
	W								H		
	W								R		
	W								N		
	W								EOT		
W								GPL			
W								STP			
Execution										Data compared between the FDD and main system. Status information after command execution.	
Result	R								ST 0		
	R								ST 1		
	R								ST 2		
	R								C		
	R								H		
	R								R		
	R								N		
Scan High or Equal											
Command	W	MT	MF	SK	1	1	1	0	1	Command Codes	
	W	X	X	X	X	X	HD	US1	US0		
	W										Sector ID information prior to command execution.
	W								C		
	W								H		
	W								R		
	W								N		
	W								EOT		
W								GPL			
W								STP			
Execution										Data compared between the FDD and main system. Status information after command execution.	
Result	R								ST 0		
	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 Execution No Result Phase	W	Recalibrate								Command Codes Head retracted to track 0
	W	0	0	0	0	0	1	1	1	
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	
Command No Result Phase	W	Specify								Command Codes
	W	0	0	0	0	0	0	1	1	
Command Result	W	Sense Drive Status								Command Codes Status information about FDD.
	R	0	0	0	0	0	1	0	0	
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	
Command Result	W	Invalid								Invalid command codes (NoOp - FDC goes into standy state). ST 0 = 80.
	R	Invalid Codes ST 0								

Bit			Description
No.	Name	Symbol	
D7	Interrupt Code	IC	D7 = 0 and D6 = 0 Normal termination of command (NT). Command was completed and properly executed.
D6			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	Unit Select 1	US 1	These flags are used to indicate a drive unit number at interrupt.
D0	Unit Select 0	US 0	

Command Status Register 0

No.	Bit		Description
	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	T0	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 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 de-gates 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

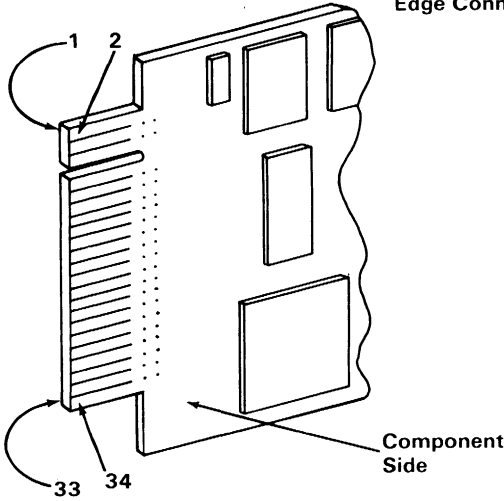
- Drive Select A and B (Driver: 7438)
These two lines are used by drives A and B to degate all drivers to the adapter and receivers from the attachment (except motor enable) when the line associated with a drive is inactive.

- 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.
- Write Enable (Driver: 7438)
The drive disables write current in the head unless this line is active.

Adapter Inputs

- 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.

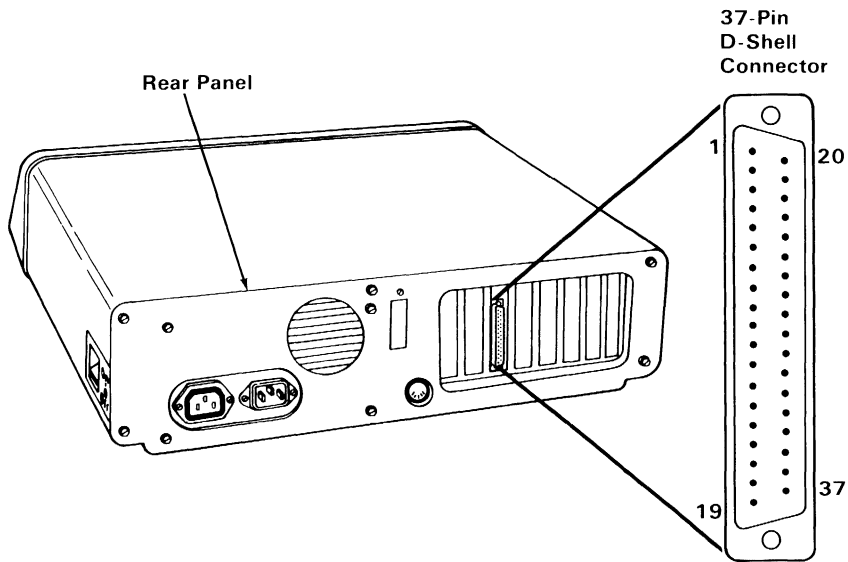
**34-Pin Keyed
Edge Connector**



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	←
	Step Pulse	20	←
	Write Data	22	←
	Write Enable	24	←
	Track 0	26	←
	Write Protect	28	→
	Read Data	30	→
	Select Head 1	32	→
	Unused	34	←

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	

External Drives

Drive Adapter

Connector Specifications (Part 2 of 2)

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.

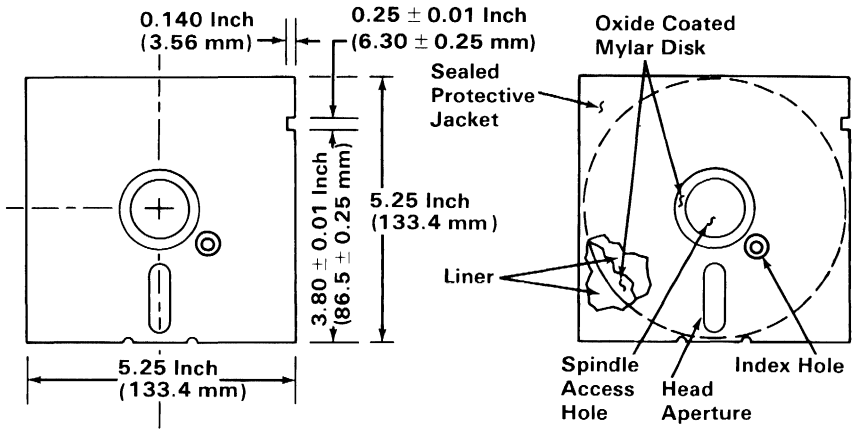
For interface information, refer to "IBM 5-1/4" Diskette Drive Adapter" earlier in this section.

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.

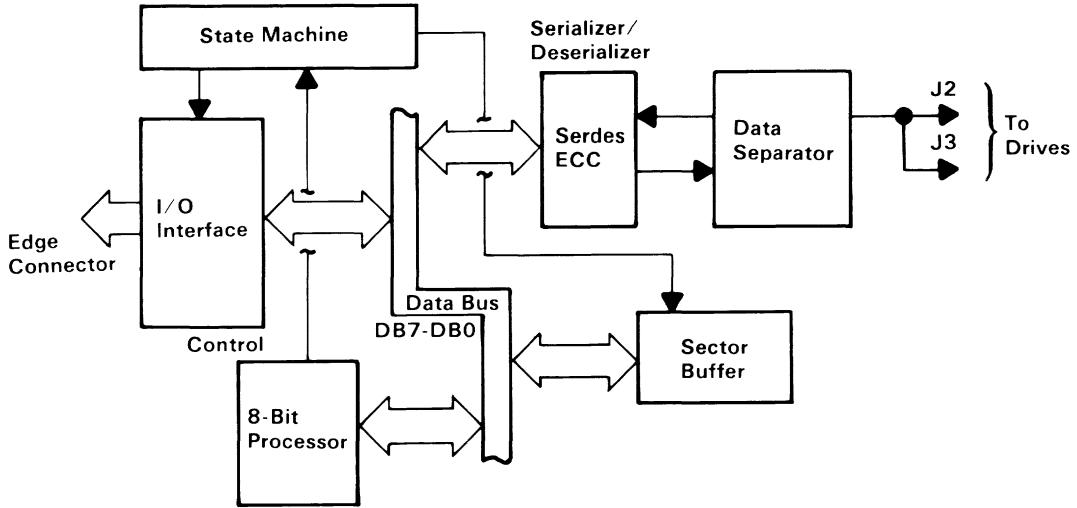
The fixed disk drive adapter provides automatic 11-bit burst error detection and correction in the form of 32-bit error checking and correction (ECC).

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	1	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 tables list the error types and error codes found in byte 0:

	Error Type	Error Code	
Bits	5 4	3 2 1 0	Description
	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.

Bits	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)	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)	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)	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)	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
	Byte 3	Cylinder Low
	Byte 4	0 0 0 Interleave
	Byte 5	r 0 0 0 0 s s s
Ready Verify (Class 0, Opcode 05)	Bit	7 6 5 4 3 2 1 0
	Byte 0	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)	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
	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)	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
	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)	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)	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)	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
	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 border="1"> <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 border="1"> <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 border="1"> <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 border="1"> <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 border="1"> <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 = (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.

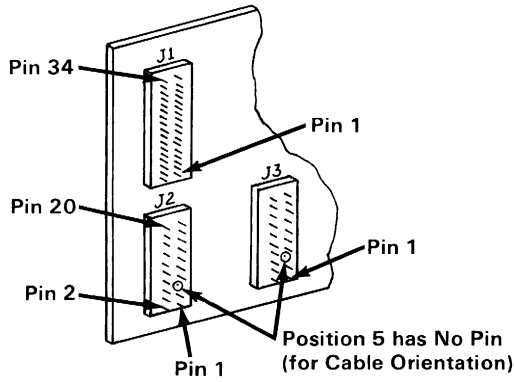
The following figure is a table of the four read/write ports:

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 storage (ROS) 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 ($-\text{IOR}$) or I/O Write ($-\text{IOW}$) 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 ($-\text{DACK 3}$) 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

Disk Drive Connector J1 Disk Drive Connector J1

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

Disk Drive Connector J2 or J3 Disk Adapter Connector J2 or J3

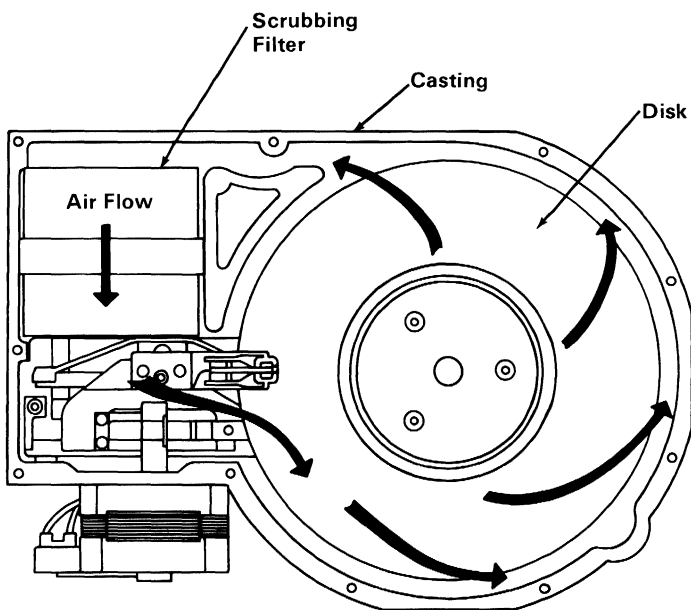
Fixed Disk Adapter Interface Specifications

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 ±1%
Transfer Rate	5.0 M bits/sec
Recording Mode	MFM
Power	+12 Vdc ± 5% 1.8 A (4.5 A maximum) +5 Vdc ± 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 and a memory module kit are available for the IBM Personal Computer XT. They are the 32KB, 64KB, and 64/256KB Memory Expansion Options and the 64KB Memory Module Kit. The base system has a standard 128K of RAM on the system board. One or two memory module kits can be added, providing the system board with 192K or 256K of RAM. The base 64/256K option has a standard 64K of RAM. One, two, or three 64K memory module kits may be added, providing the 64/256K option with 128K, 192K, or 256K of RAM. A maximum of 256K of RAM can be installed on the system board as modules without using any of the system unit expansion slots or expansion options. The system board must be populated to the maximum 256K of RAM 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 DIP switches on the option.

The 32K and 64K options both use 16K by 1 bit memory modules, while the 64/256K option uses 64K by 1 bit memory modules. On the 32K and 64/256K options, 16-pin industry-standard parts are used. On the 64K option, stacked modules are used resulting in a 32K by 1 bit, 18-pin module. This allows the 32K and 64K options to have approximately the same physical size.

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 “Appendix G: Switch Settings” for the proper memory expansion option switch settings.

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)	32K by 1 Bit Module (used on 64K option)	64K by 1 Bit Module (used on 64/256K option)
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: A19=0; OFF: A19=1	ON: A19=0; OFF: A19=1
2	ON: A18=0; OFF: A18=1	ON: A18=0; OFF: A18=1
3	ON: A17=0; OFF: A17=1	ON: A17=0; OFF: A17=1
4	ON: A16=0; OFF: A16=1	ON: A16=0; OFF: A16=1
5	ON: A15=0; OFF: A15=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 A15 (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

Memory Option Switch Settings

Switch settings for all memory expansion options are located in “Appendix G: Switch Settings.”

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

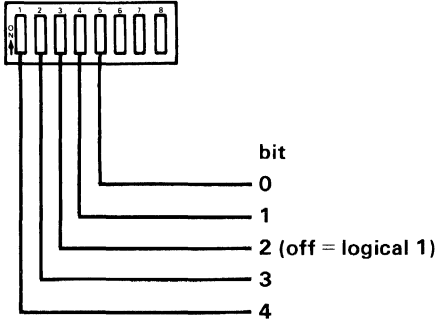
Starting Address = xxxK

32K xxxK =Decimal value

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.

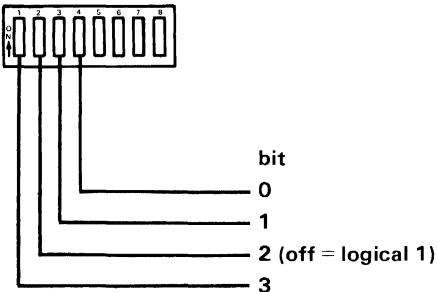
Starting Address = xxxK

64K xxxK =Decimal value

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.

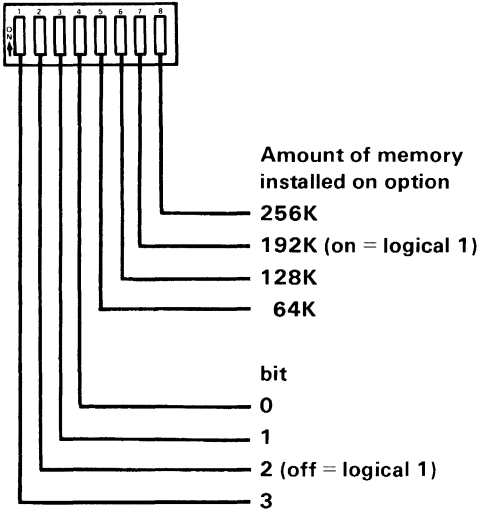
Starting Address = xxxK

64K xxxK
 =Decimal value

Convert decimal value to binary

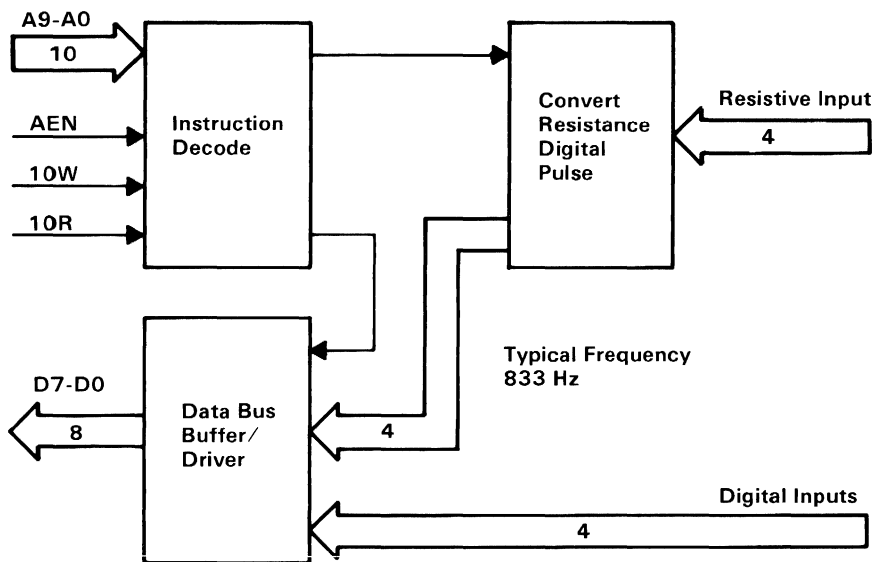
Bit.....3 2 1 0
Bit value...8 4 2 1

Switch



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 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 CHCK:	Unused.
I/O CH RDY:	Unused.
RESET DRV:	Unused.
-5 Vdc, +12 Vdc, -12 Vdc:	Unused.

Interface Description

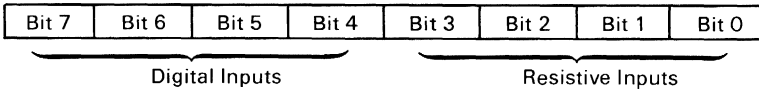
The game control adapter has eight input lines, four of which are digital inputs and 4 of which are resistive inputs. The inputs are read with one In from address hex 201.

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.



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:

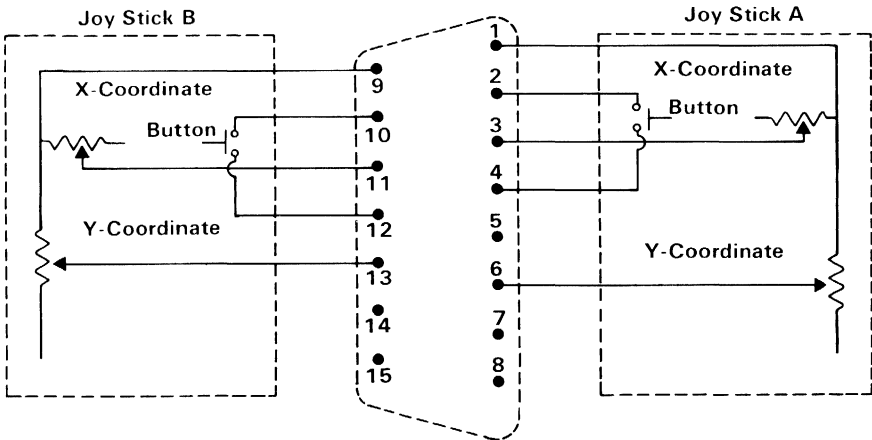
Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
B-#2 Button	B-#1 Button	A-#2 Button	A-#1 Button	B-Y Coordinate	B-X Coordinate	A-Y Coordinate	A-X Coordinate

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 of 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

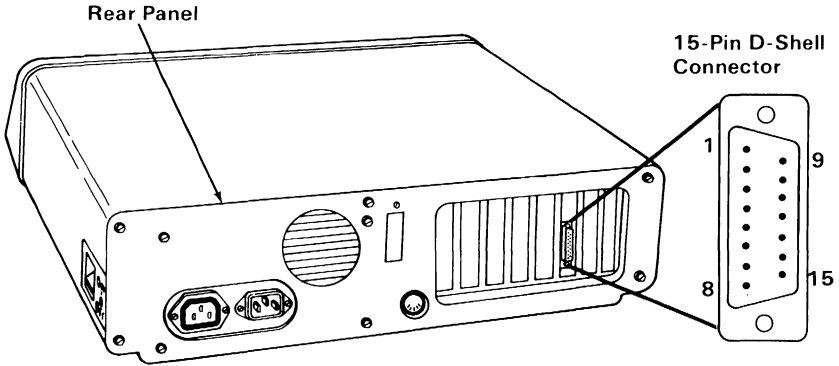
Refer to “Joy Stick Schematic Diagram” for attaching game controllers.

15-Pin Male D-Shell Connector

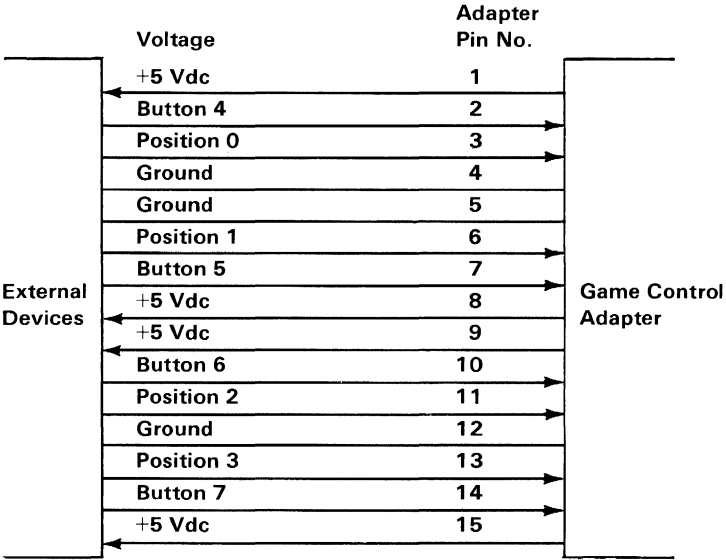


Note: Potentiometer for X- and Y-Coordinates has a range of 0 to 100 k-ohms. Button is normally open; closed when pressed.

Joy Stick Schematic Diagram



At Standard TTL Levels



Connector Specifications

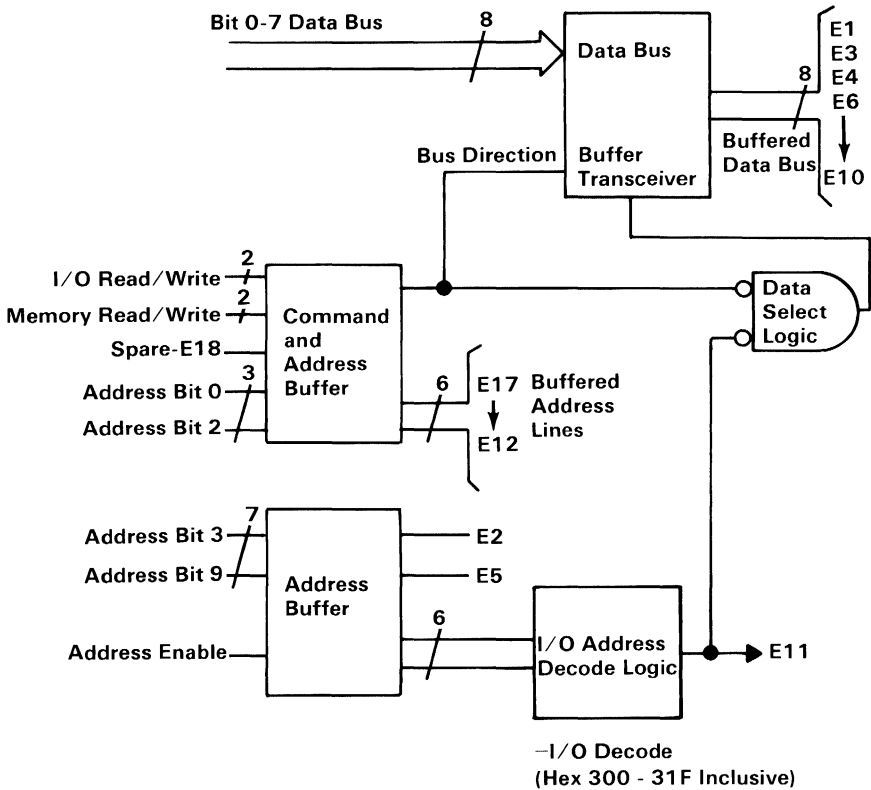
IBM Prototype Card

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.



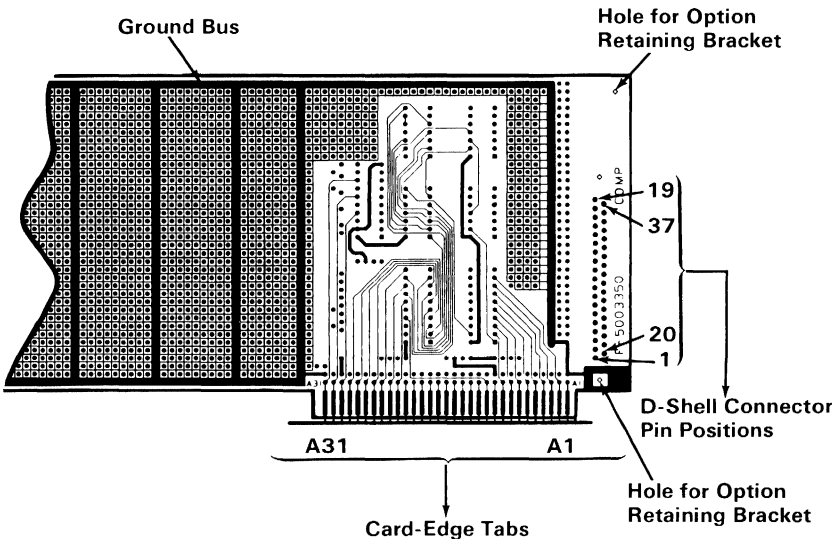
Prototype Card Block Diagram

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).

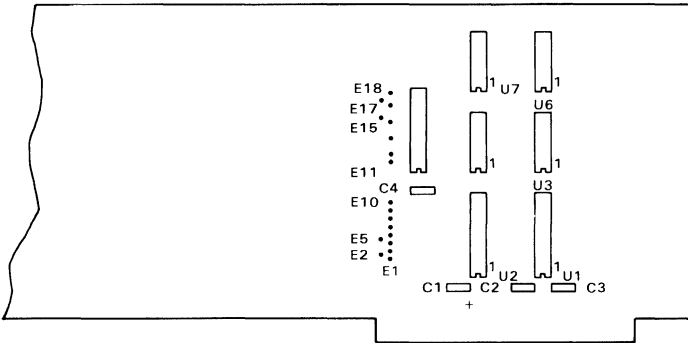
Prototype Card Layout

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.



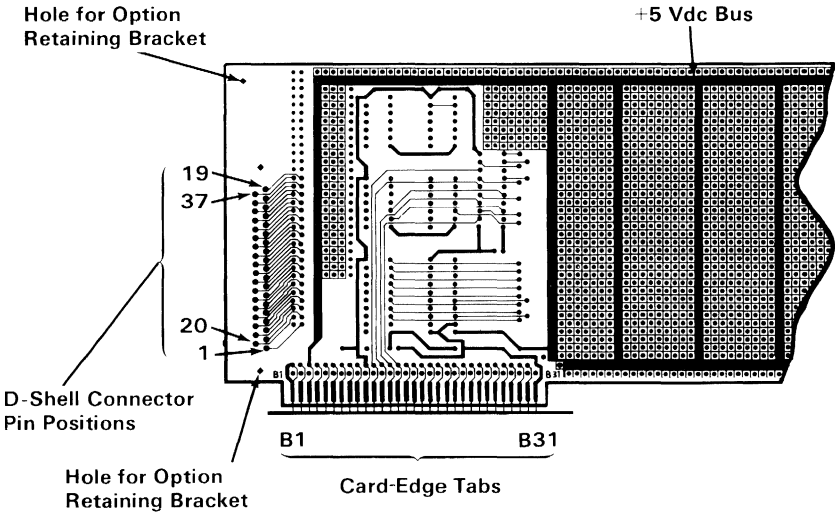
Component Side

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-edged 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.

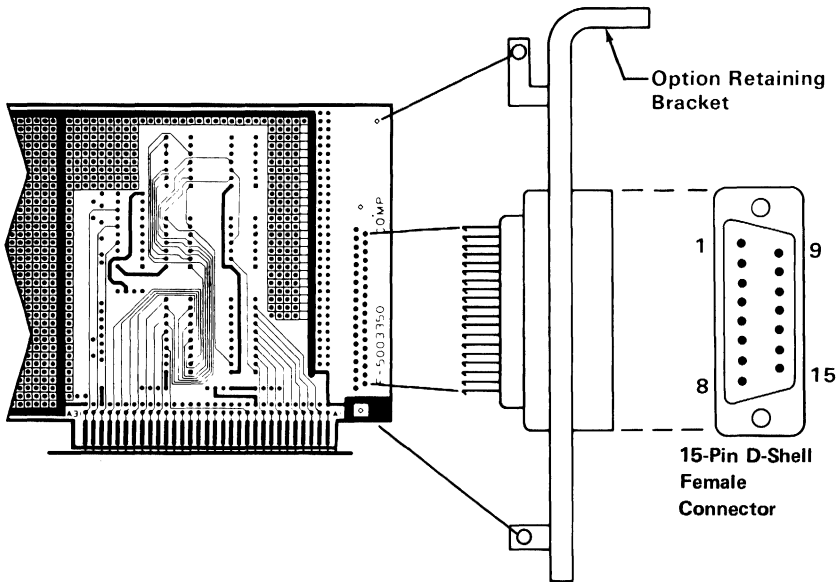
Refer to the power supply information in this manual for the power limitations to be observed.

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

The following example shows a 15-pin, D-shell, female connector attached to a prototype card.



Component Side

IBM Asynchronous Communications Adapter

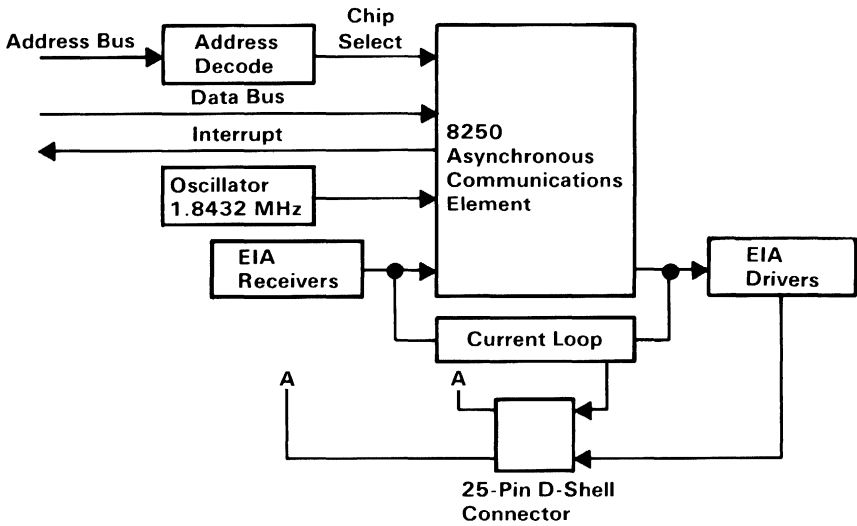
The asynchronous communications adapter system control signals and voltage requirements are provided through a 2 by 31 position card edge tab. Two jumper modules are provided on the adapter. One jumper module selects either RS-232C or current-loop operation. The other jumper module selects one of two 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	3FA	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											
A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	DLAB	Register
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).

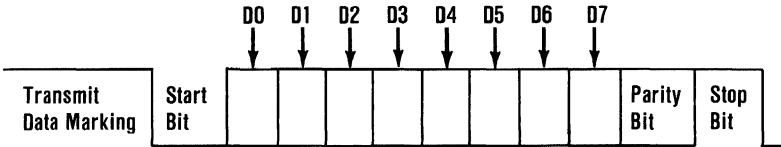
A2, A1 and A0 bits are "don't cares" and are used to select the different register of the communications chip.

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:



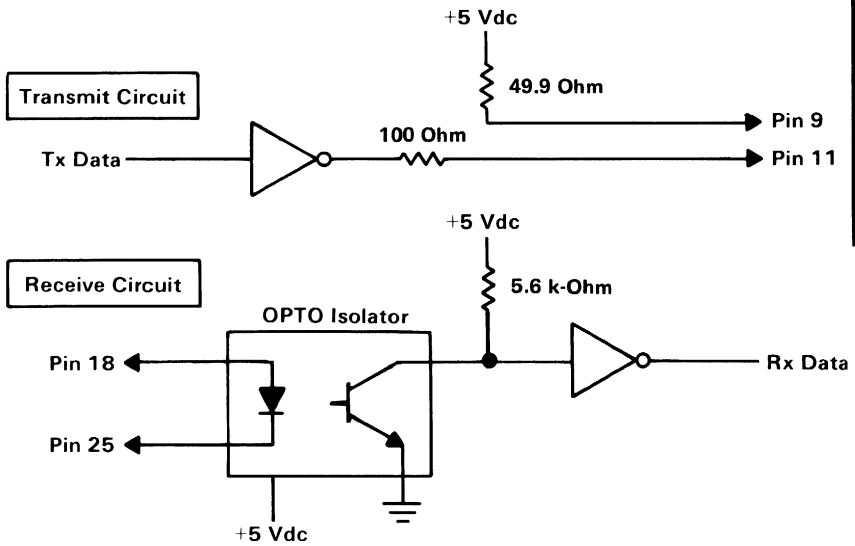
Data bit 0 is the first bit to be transmitted or received. The adapter automatically inserts the start bit, the correct parity bit if programmed to do so, and the stop bit (1, 1-1/2, or 2 depending on the command in the line-control register).

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

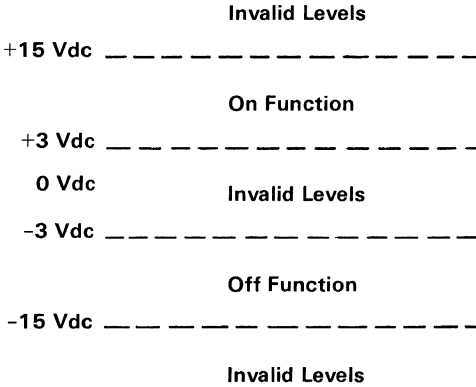
The voltage interface is a serial interface. It supports certain data and control signals, as listed below.

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 $+3\text{ Vdc}$ with respect to signal ground. The region between $+3\text{ Vdc}$ 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 $+15\text{ Vdc}$ 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 $+3\text{ Vdc}$ 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 (CS0, CS1, $\overline{\text{CS2}}$), Pins 12-14: When CS0 and CS1 are high and $\overline{\text{CS2}}$ is low, the chip is selected. Chip selection is complete when the decoded chip select signal is latched with an active (low) address strobe ($\overline{\text{ADS}}$) input. This enables communications between the INS8250 and the processor.

Data Input Strobe (DISTR, $\overline{\text{DISTR}}$) Pins 22 and 21: When DISTR is high or $\overline{\text{DISTR}}$ 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 DISTR or $\overline{\text{DISTR}}$ input is required to transfer data from the INS8250 during a read operation. Therefore, tie either the DISTR input permanently low or the $\overline{\text{DISTR}}$ input permanently high, if not used.

Data Output Strobe (DOSTR, $\overline{\text{DOSTR}}$), Pins 19 and 18: When DOSTR is high or $\overline{\text{DOSTR}}$ 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 DOSTR or $\overline{\text{DOSTR}}$ input is required to transfer data to the INS8250 during a write operation. Therefore, tie either the DOSTR input permanently low or the $\overline{\text{DOSTR}}$ input permanently high, if not used.

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

Note: An active $\overline{\text{ADS}}$ 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 $\overline{\text{ADS}}$ 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 from 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, affects 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 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 (SOUT, INTRPT, $\overline{\text{OUT 1}}$, $\overline{\text{OUT 2}}$, $\overline{\text{RTS}}$, $\overline{\text{DTR}}$) 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 ($\overline{\text{RI}}$), Pin 39: When low, indicates that a telephone ringing signal has been received by the modem or data set. The $\overline{\text{RI}}$ 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 $\overline{\text{RI}}$ 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 register interrupt is enabled.

VCC, Pin 40: +5 Vdc supply.

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

Output Signals

Data Terminal Ready ($\overline{\text{DTR}}$), 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 $\overline{\text{DTR}}$ signal is set high upon a master reset operation.

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

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

Output 2 ($\overline{\text{OUT 2}}$), Pin 31: User-designated output that can be set to an active low by programming bit 3 (OUT 2) of the modem control register to a high level. The $\overline{\text{OUT 2}}$ 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 $\overline{\text{CS2}}$ 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 ($\overline{\text{BAUDOUT}}$), 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 $\overline{\text{BAUDOUT}}$ 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 Bus (D7-D0), 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 through 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

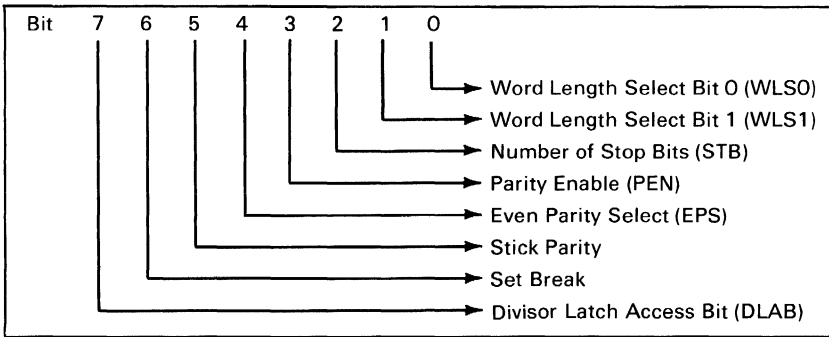
The INS8250 has a number of accessible registers. The system programmer may access or control any of the INS8250 registers through the processor. These registers are used to control INS8250 operations and to transmit and receive data. A table listing and description of the accessible registers follows.

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.

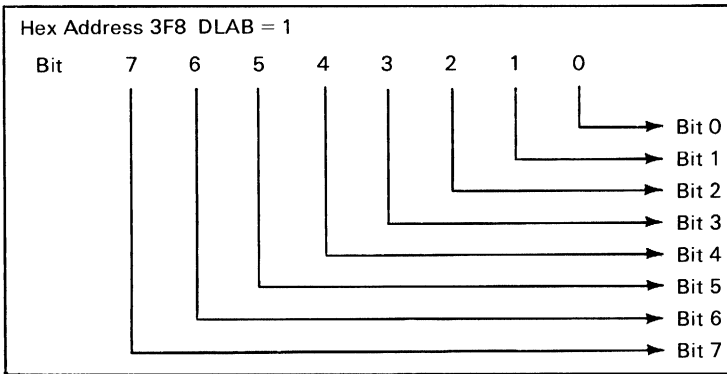
Bit 5: This bit is the stick parity bit. When bit 3 is a logical 1 and bit 5 is a logical 1, the parity bit is transmitted and then detected by the receiver as a logical 0 if bit 4 is a logical 1, or as a logical 1 if bit 4 is a logical 0.

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.

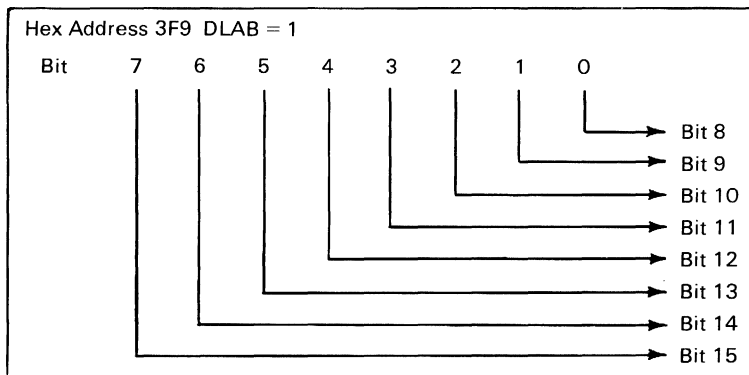
Bit 7: This bit is the divisor latch access bit (DLAB). It must be set high (logical 1) to access the divisor latches of the baud rate generator during a read or write operation. It must be set low (logical 0) to access the receiver buffer, the transmitter holding register, or the interrupt enable register.

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 x the baud rate [divisor # = (frequency input)/(baud rate x 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.

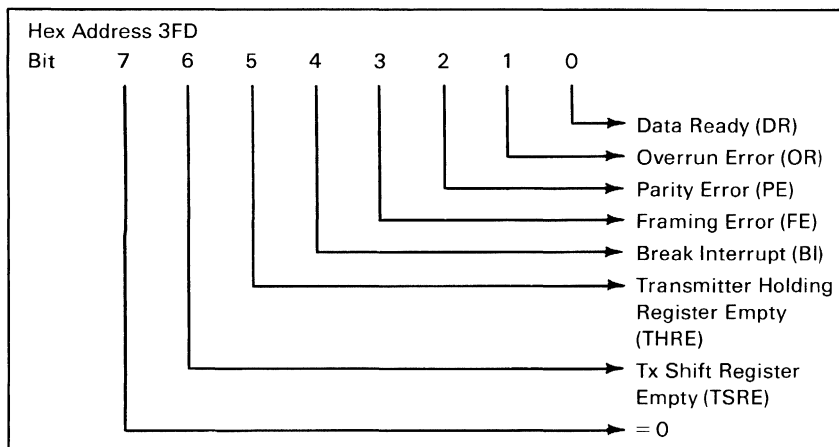
Note: The maximum operating frequency of the baud generator is 3.1 MHz. In no case should the data rate be greater than 9600 baud.

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.

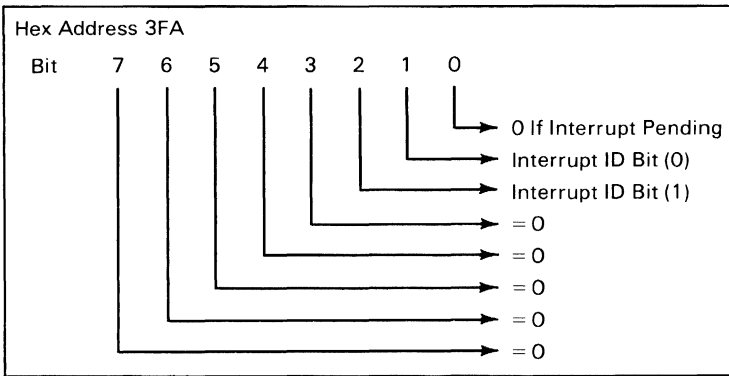
Bit 6: This bit is the transmitter shift register empty (TSRE) indicator. Bit 6 is set to a logical 1 whenever the transmitter shift register is idle. It is reset to logical 0 upon a data transfer from the transmitter holding register to the transmitter shift register. Bit 6 is a read-only bit.

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.

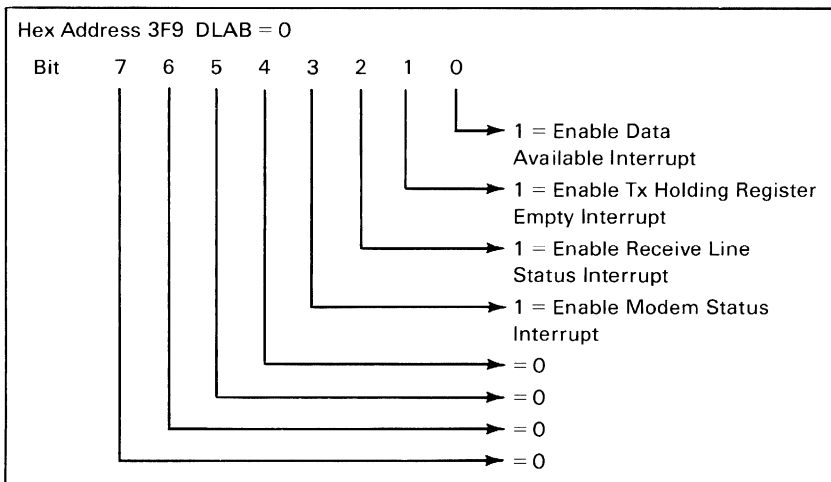
Bits 3 through 7: These five bits of the IIR are always logical 0.

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.

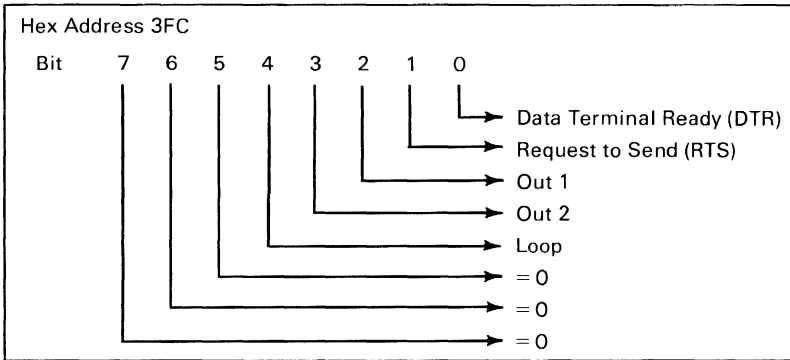
Bit 2: This bit enables the receiver line status interrupt when set to logical 1.

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 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 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 $\overline{\text{DIR}}$ 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 ($\overline{\text{CTS}}$, $\overline{\text{DRS}}$, $\overline{\text{RLSD}}$, 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.

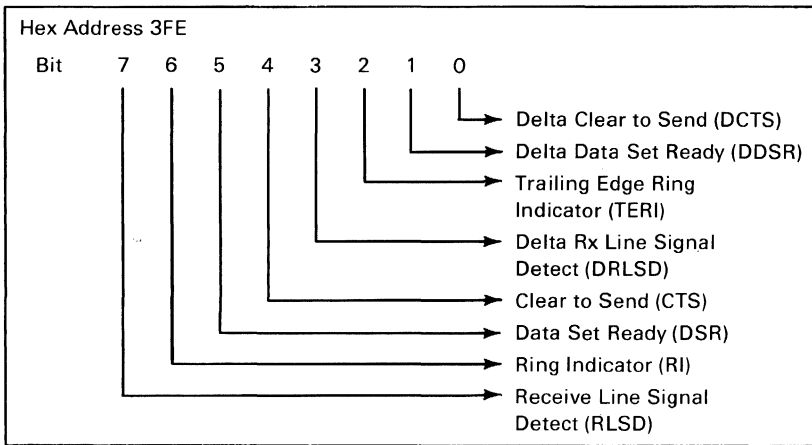
The INS8250 interrupt system can be tested by writing into the lower four bits of the modem status register. Setting any of these bits to a logical 1 generates the appropriate interrupt (if enabled). The resetting of these interrupts is the same as in normal INS8250 operation. To return to normal operation, 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 $\overline{\text{DRS}}$ input to the chip has changed 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.

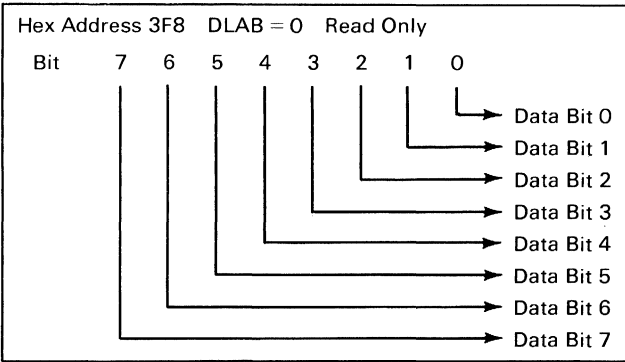
Bit 5: This bit is the complement of the data set ready ($\overline{\text{DSR}}$) input. If bit 4 of the MCR is set to a logical 1, this bit is equivalent to DTR in the MCR.

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:

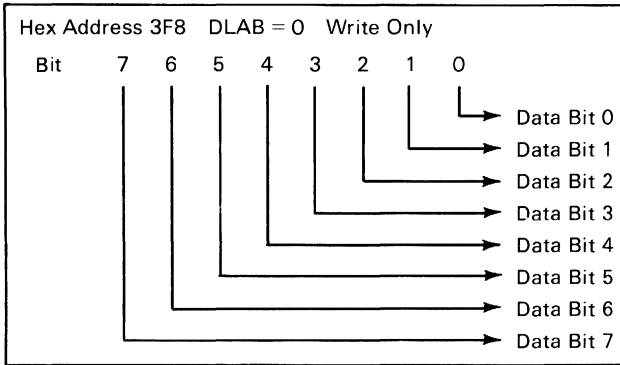


Receiver Buffer Register (RBR)

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)

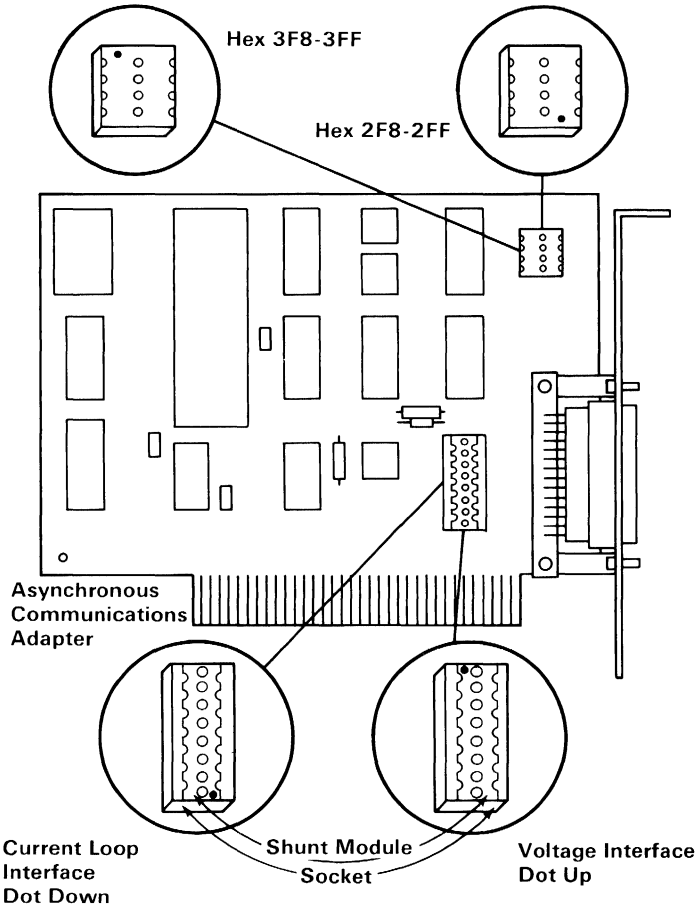
Bit 0 is the least significant bit and is the first bit serially transmitted.

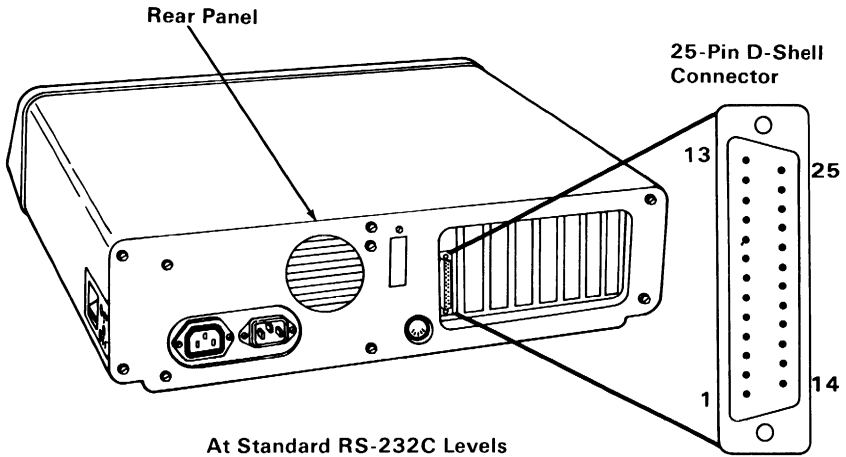
Selecting the Interface Format and Adapter Address

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)

	Description	Pin		
	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		
External Device	NC	13	Asynchronous Communications Adapter (RS-232C)	
	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 not be used to drive inductive devices, such as relay coils.

Connector Specifications

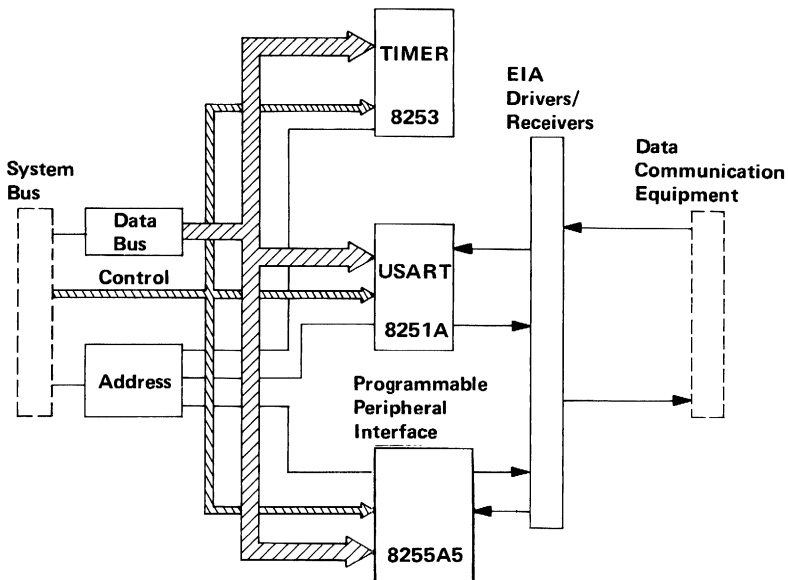
Notes:

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

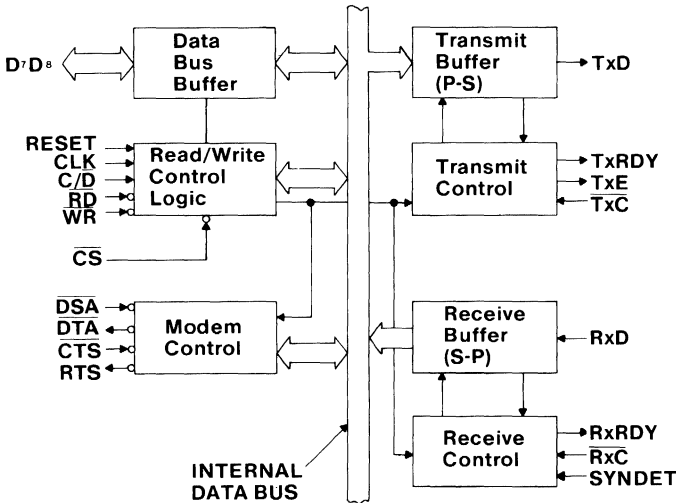
Functional Description

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 RxD (Receive Data) pin, and is clocked in on the rising edge of RxC (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.

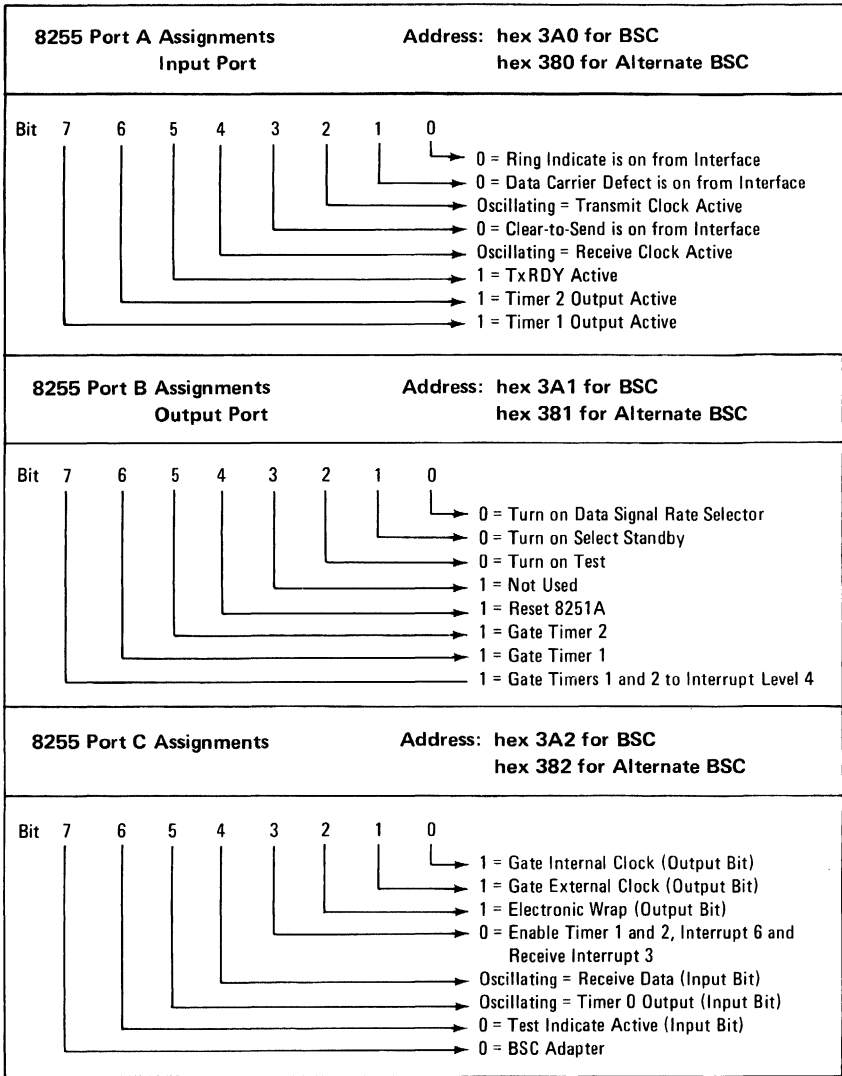
Programming Considerations

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.



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)

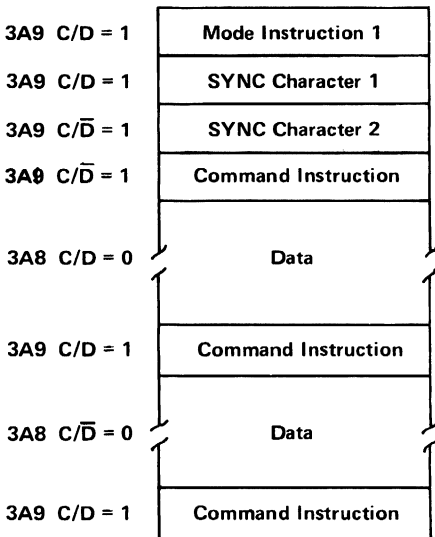
8253-5 Control Word Format

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.

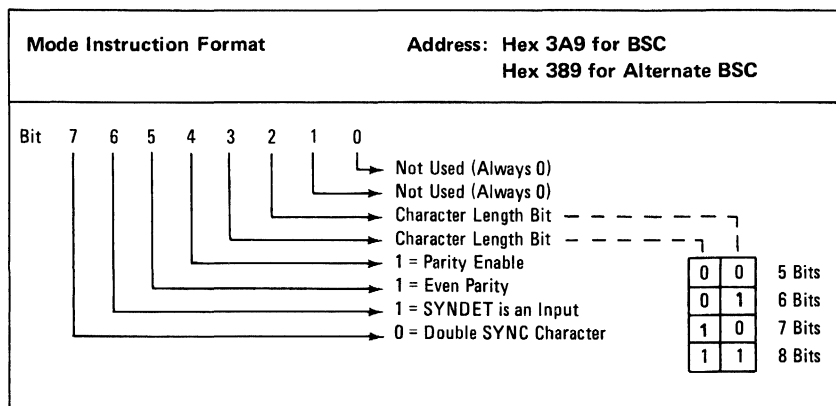


Typical Data Block

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.

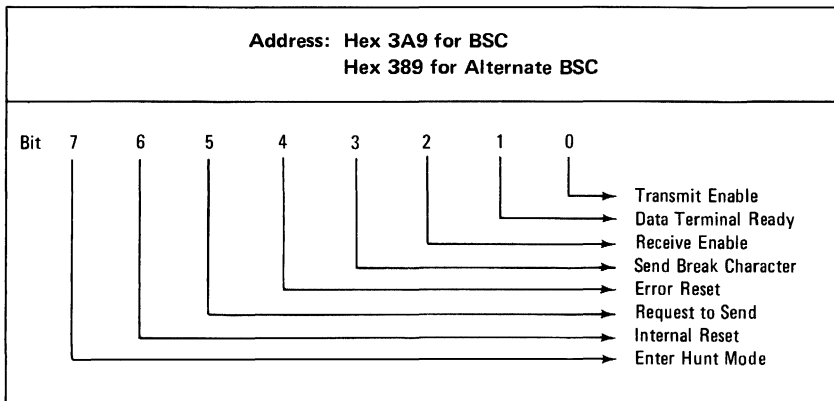
Bit 6 External synchronization is set by this bit. A 1 on this bit establishes synchronization detection as an input.

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)
<p>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.</p>	

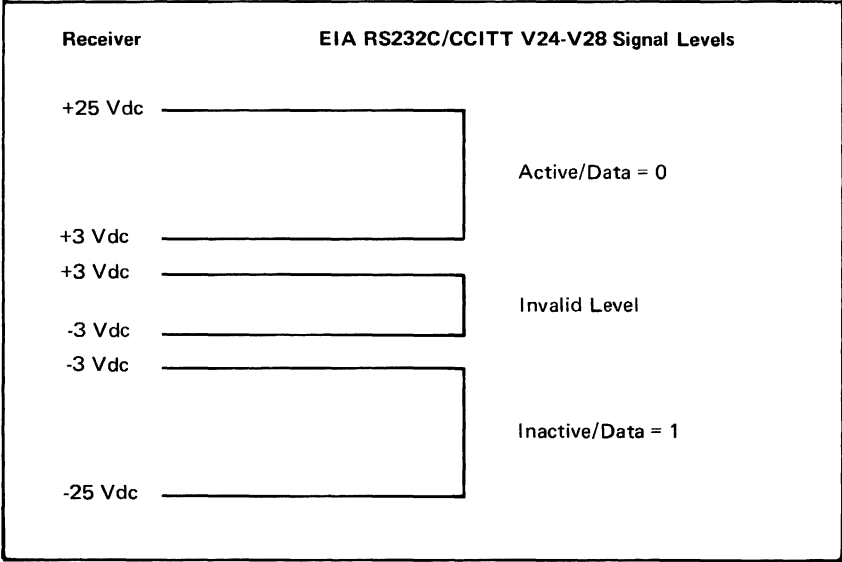
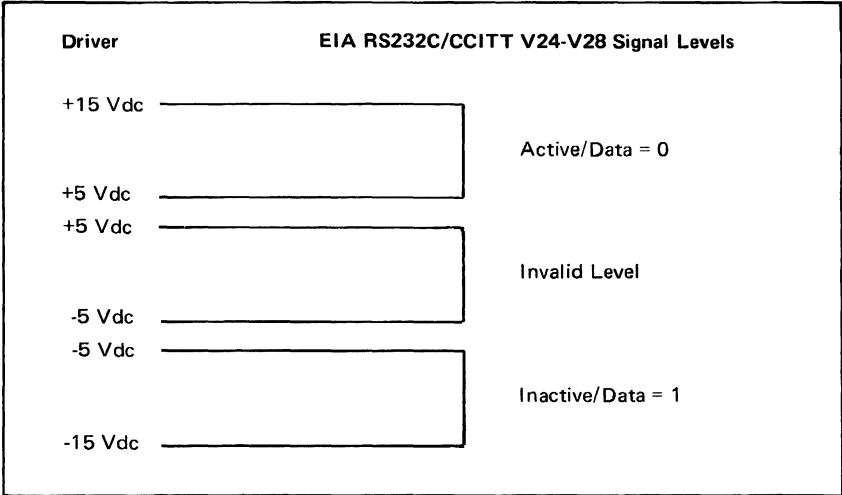
Status Read Format

- 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.



Interface Voltage Levels

Interrupt Information

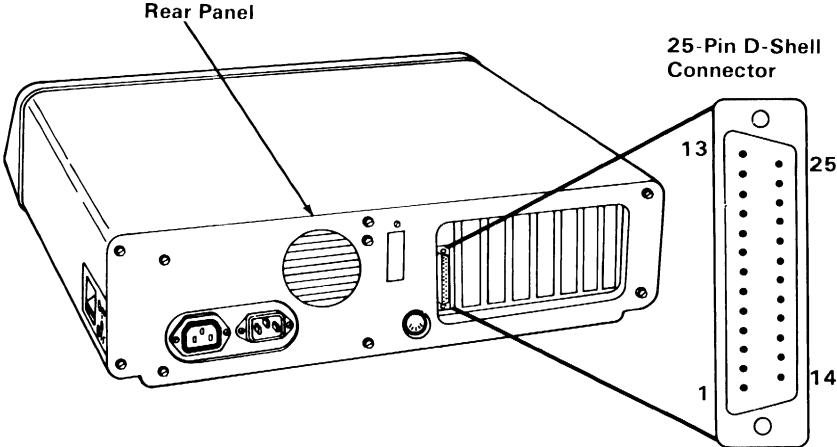
Interrupt Level 4: Transmitter Ready
 Counter 1
 Counter 2

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	
	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	→

*Not standardized by EIA (Electronics Industry Association).

Connector Specifications

Notes:

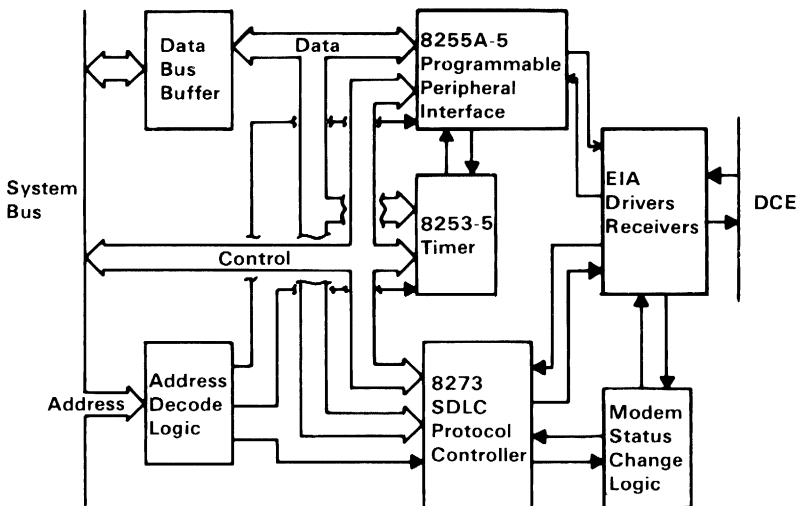
IBM Synchronous Data Link Control (SDLC) Communications Adapter

The SDLC communications adapter system control, voltage, and data signals are provided through a 2 by 31 position card edge tab. Modem interface is in the form of EIA drivers and receivers connecting to an RS232C 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.

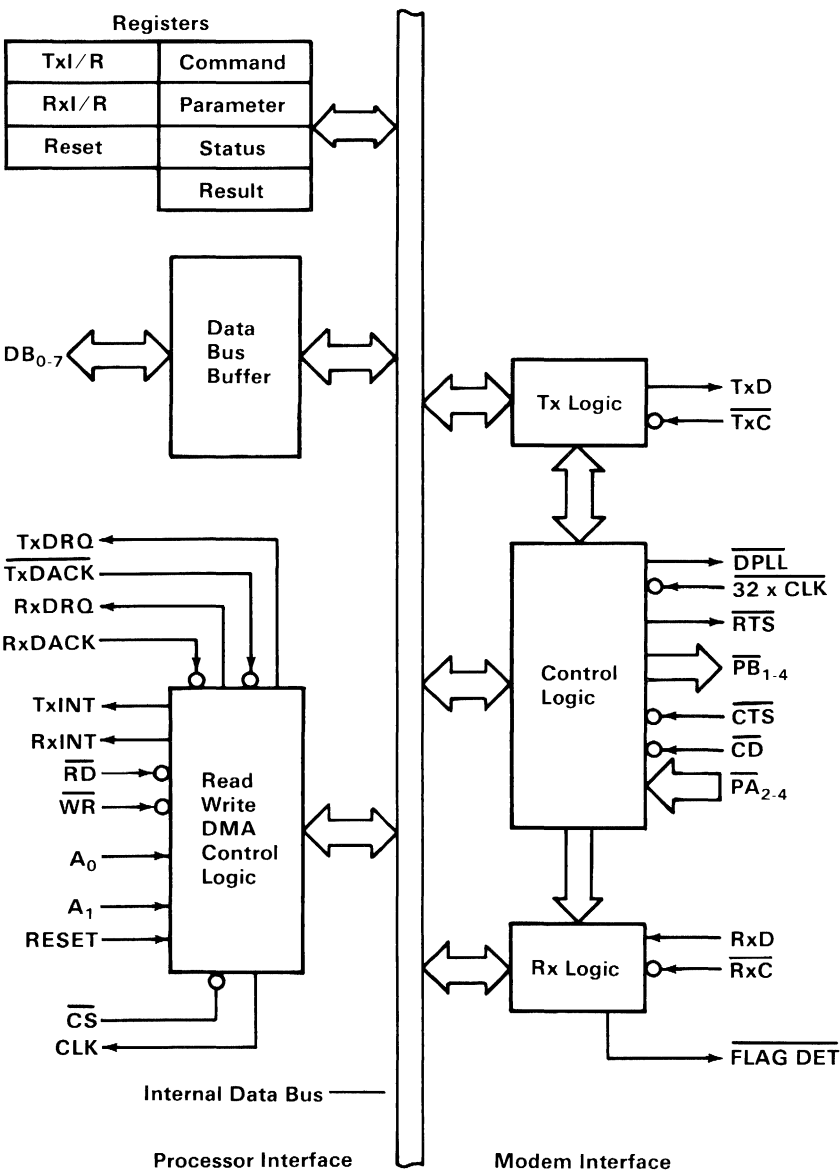
The 8273 SDLC protocol controller operations, whether transmission, reception, or port read, are each comprised of three phases:

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

- TxD_{DRQ}/RxD_{DRQ}** This line requests a DMA to or from memory and is asserted by the 8273.
- TxD_{DACK}/RxD_{DACK}** 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 TxD_{DACK}/RxD_{DACK} 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.

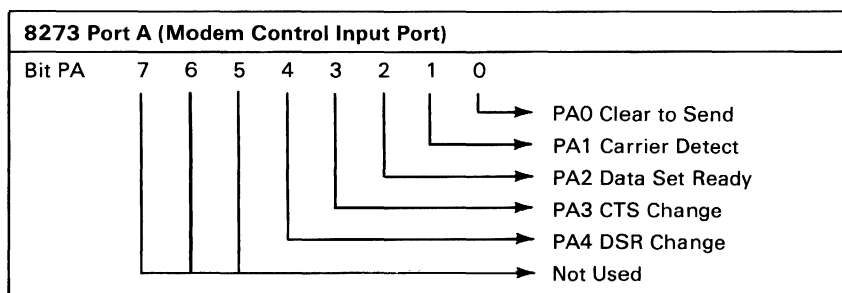
Modem Interface

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.



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.

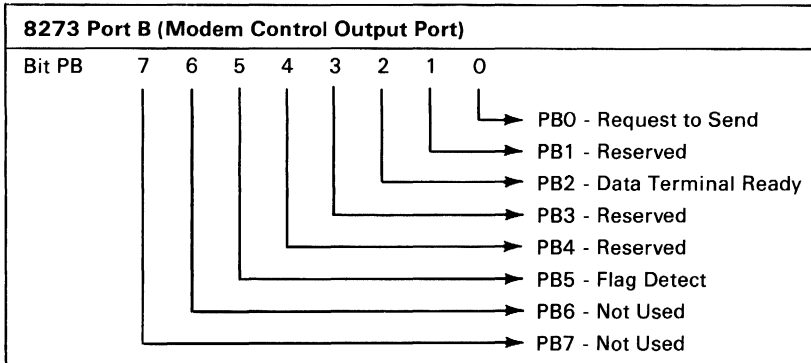
Bit PA2 This bit is a sense bit for data set ready (DSR).

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.

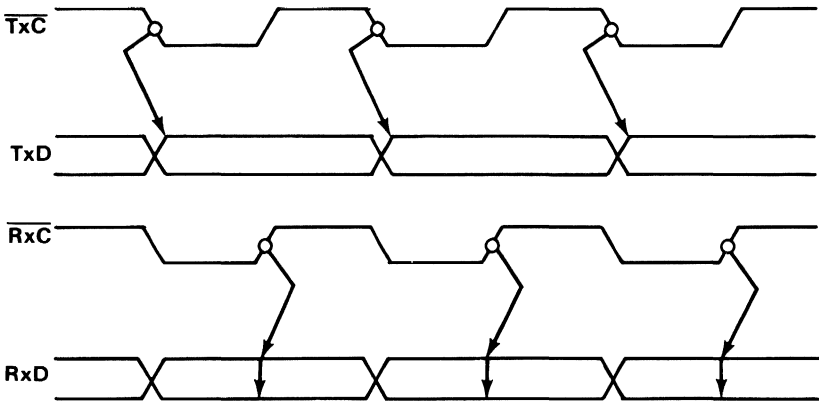
Bit PB6 Not used.

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 Tx_D (transmitted data output) and Rx_D (received data input), and the respective clocks. The leading edge of Tx_C generates new transmitted data and the trailing edge of Rx_C 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 eight 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	
		<ul style="list-style-type: none"> 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	
		<ul style="list-style-type: none"> 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

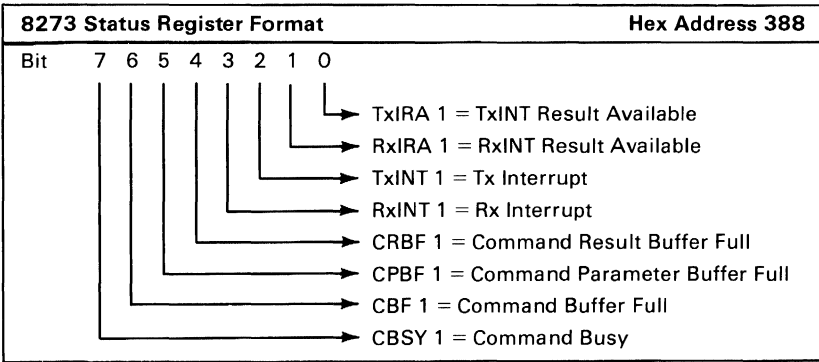
The 8253-5 is driven by a processor clock signal divided by two. It has the following output:

- 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 level 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

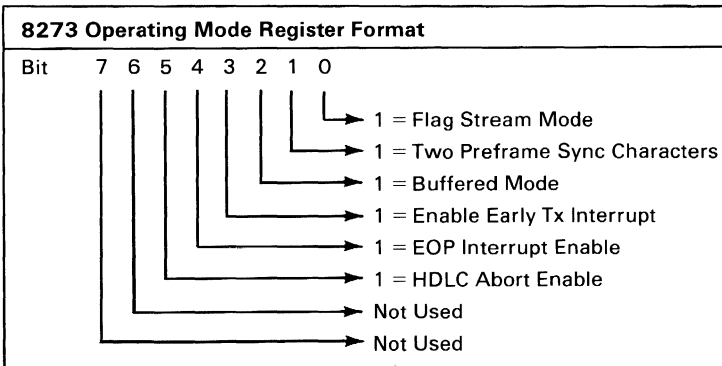
The initialization/configuration commands manipulate internal registers of the 8273, which define operating modes. After chip reset, the 8273 defaults to all 1's in the mode registers. The initialization/configuration commands either set or reset specified bits in the registers depending on the type 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.

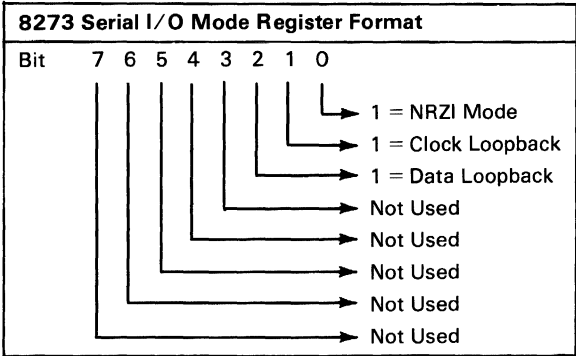
The following are descriptions of each bit of the operating, serial I/O, one-bit delay, and data transfer mode registers.

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

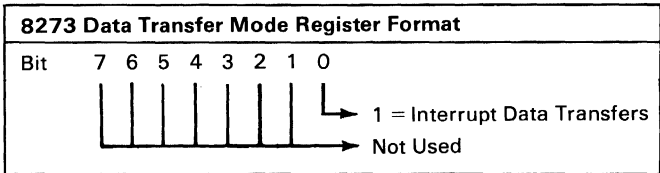


Bit 0 Set to 1, this bit specifies NRZI encoding and decoding. Resetting this bit specifies that transmit and receive data be treated as a normal positive-logic bit stream.

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.

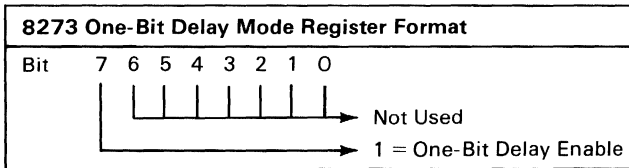
Bit 2 When bit 2 is set, the transmitted data is internally routed to the received data circuitry. The reset condition causes the transmitted and received data to be routed to their respective 8273 I/O pins.

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.

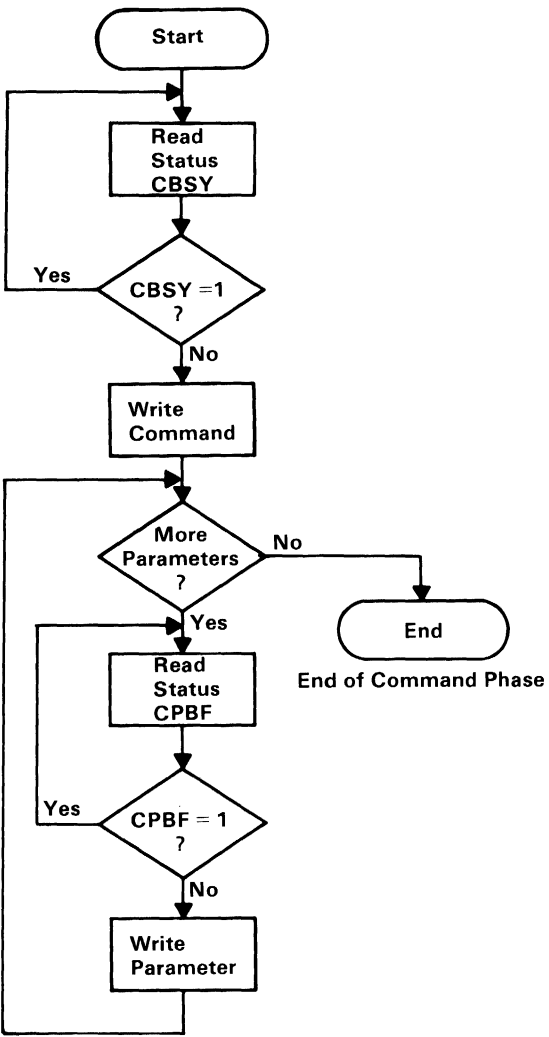
The system software starts the command phase by selecting the 8273 command register address and writing a command byte into the register. The following table lists command and parameter information for the 8273 protocol controller. If further information is required by the 8273 prior to execution of 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	B0,B1	RIC,R0,R1, A,C	RXI/R	Yes
Selective Receive	C1	B0, B1, 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.



8273 SDLC Protocol Controller Command Phase Flowchart

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

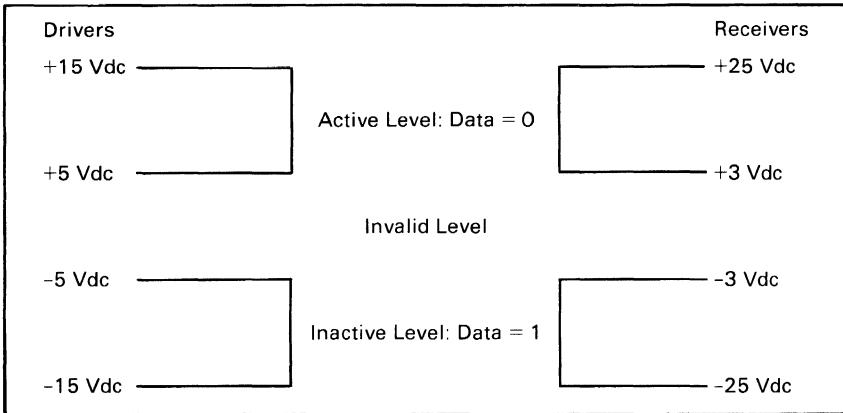
Interrupt Level 3	Transmit/Receive Interrupt
Interrupt Level 4	Timer 1 Interrupt Timer 2 Interrupt Clear to Send Changed Data Set Ready Changed
DMA Level One is used for Transmit and Receive	

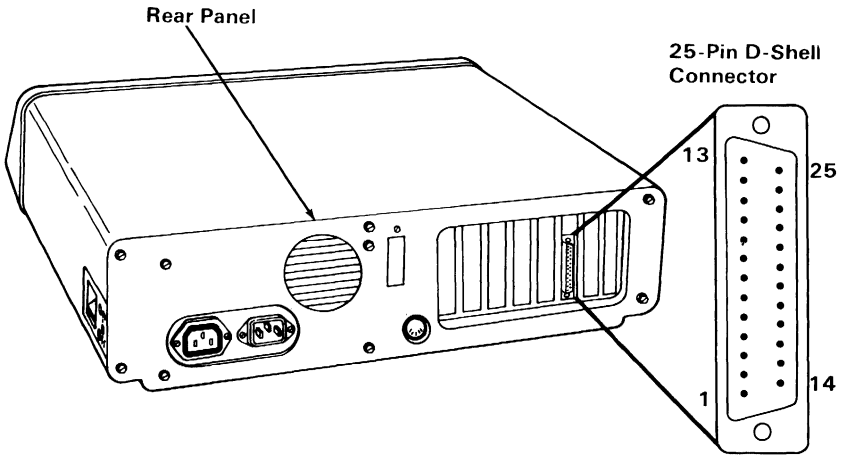
Interrupt Information

Interface Information

The SDLC communications adapter conforms to interface signal levels standardized by the Electronics Industries Association RC-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	
	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	→

*Not standardized by EIA (Electronics Industry Association).

Connector Specifications

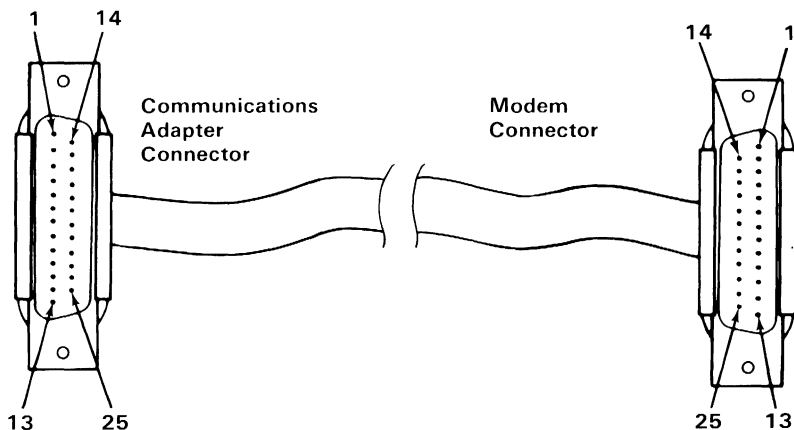
Notes:

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 RC-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

SECTION 2: ROM BIOS AND SYSTEM USAGE

ROM BIOS	2-2
Keyboard Encoding and Usage	2-11



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	D11
4-7	1	Single Step	D11
8-B	2	Nonmaskable	NMI_INT
C-F	3	Breakpoint	D11
10-13	4	Overflow	D11
14-17	5	Print Screen	PRINT_SCREEN
18-1B	6	Reserved	D11
1D-1F	7	Reserved	D11
20-23	8	Time of Day	TIMER_INT
24-27	9	Keyboard	KB_INT
28-2B	A	Reserved	D11
2C-2F	B	Communications	D11
30-33	C	Communications	D11
34-37	D	Disk	D11
38-3B	E	Diskette	DISK_INT
3C-3F	F	Printer	D11
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_PARMS
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	ROM BIOS	See BIOS Listing
490-4EF		Reserved
4F0-4FF		Reserved as Intra-Application Communication Area for any application
500-5FF	DOS	Reserved for DOS and BASIC
500		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
<p>Example</p> <pre>100 Print PEEK (&H2E) + 256*PEEK (&H2F)</pre> <p style="margin-left: 40px;"> L H </p> <div style="display: flex; align-items: center; margin-left: 40px;"> } <div style="border: 1px solid black; padding: 5px; display: flex; gap: 20px;"> <div style="border-right: 1px solid black; padding: 5px;">Hex 64</div> <div style="padding: 5px;">Hex 00</div> </div> </div>		

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 (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.”

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	!	-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	-1	-1	-1	-1
Caps Lock				
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	-1	-1	Break (Note 2)	-1
Scroll Lock				
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 (Upper Case 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

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

This 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 backwards 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 PrtSc (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 templates 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

DOS 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)	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 insert mode	Ins
Exit insert mode	Ins
Make new line the template	F5
String separator in REPLACE	F6
End of file in keyboard input	F6

BASIC Screen Editor Special Functions

APPENDIX A: ROM BIOS LISTINGS

	Page	Line Number
System ROM BIOS		
Equates	A-2	12
8088 Interrupt Locations	A-2	35
Stack	A-2	67
Data Areas	A-2	76
Power-On Self-Test	A-5	239
Boot Strap Loader	A-20	1408
I/O Support		
Asynchronous Communications		
(RS-232C)	A-21	1461
Keyboard	A-24	1706
Diskette	A-34	2303
Printer	A-44	3078
Display	A-46	3203
System Configuration Analysis		
Memory Size Determination	A-71	5052
Equipment Determination	A-71	5083
Graphics Character Generator	A-77	5496
Time of Day	A-79	5630
Print Screen	A-81	5821

Fixed Disk ROM BIOS

Fixed Disk I/O Interface	A-84	1
Boot Strap Loader	A-89	399

Appendix A

LOC OBJ

LINE SOURCE

```

1 $TITLE(BIOS FOR THE IBM PERSONAL COMPUTER XT)
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 ;-----
0060 15 PORT_A EQU 60H ; 8255 PORT A ADDR
0061 16 PORT_B EQU 61H ; 8255 PORT B ADDR
0062 17 PORT_C EQU 62H ; 8255 PORT C ADDR
0063 18 CMD_PORT EQU 63H
0020 19 INTA00 EQU 20H ; 8259 PORT
0021 20 INTA01 EQU 21H ; 8259 PORT
0020 21 EOI EQU 20H
0040 22 TIMER EQU 40H
0043 23 TIM_CTL EQU 43H ; 8253 TIMER CONTROL PORT ADDR
0040 24 TIMER0 EQU 40H ; 8253 TIMER/CENTER 0 PORT ADDR
0001 25 THINT EQU 01 ; TIMER 0 INTR RECDV MASK
0008 26 DMA08 EQU 08 ; DMA STATUS REG PORT ADDR
0000 27 DMA EQU 00 ; DMA CH.0 ADDR. REG PORT ADDR
0540 28 MAX_PERIOD EQU 540H
0410 29 MIN_PERIOD EQU 410H
0060 30 KBD_IN EQU 60H ; KEYBOARD DATA IN ADDR PORT
0002 31 KBDINT EQU 02 ; KEYBOARD INTR MASK
0060 32 KB_DATA EQU 60H ; KEYBOARD SCAN CODE PORT
0061 33 KB_CYL EQU 61H ; CONTROL BITS FOR KEYBOARD SENSE DATA
34
35 ;-----
36 ; 8088 INTERRUPT LOCATIONS :
37 ;-----
38
---- 39 ABS0 SEGMENT AT 0
0000 40 STG_LOCC LABEL BYTE
0008 41 ORG 2*4
0008 42 NMI_PTR LABEL WORD
0014 43 ORG 5*4
0014 44 INTS_PTR LABEL WORD
0020 45 ORG 8*4
0020 46 INT_ADDR LABEL WORD
0020 47 INT_PTR LABEL DWORD
0040 48 ORG 10H*4
0040 49 VIDEO_INT LABEL WORD
0074 50 ORG 10H*4
0074 51 PARM_PTR LABEL DWORD ; POINTER TO VIDEO PARMS
0060 52 ORG 18H*4
0060 53 BASIC_PTR LABEL WORD ; ENTRY POINT FOR CASSETTE BASIC
0078 54 ORG 01EH*4 ; INTERRUPT 1EH
0078 55 DISK_POINTER LABEL DWORD
007C 56 ORG 01FH*4 ; LOCATION OF POINTER
007C 57 EXT_PTR LABEL DWORD ; POINTER TO EXTENSION
0400 58 ORG 400H
0400 59 DATA_AREA LABEL BYTE ; ABSOLUTE LOCATION OF DATA SEGMENT
0400 60 DATA_WORD LABEL WORD
0500 61 ORG 0500H
0500 62 MFG_TEST_RTN LABEL FAR
7C00 63 ORG 7C00H
7C00 64 BOOT_LOCH LABEL FAR
---- 65 ABS0 ENDS
66
67 ;-----
68 ; STACK -- USED DURING INITIALIZATION ONLY :
69 ;-----
70
---- 71 STACK SEGMENT AT 30H
0000 (128 72 DW 128 DUP(?)
????
)
0100 73 TOS LABEL WORD
---- 74 STACK ENDS
75
76 ;-----
77 ; ROM BIOS DATA AREAS :

```

```

LOC OBJ          LINE  SOURCE

78  ;-----
79
----           80  DATA  SEGMENT AT 40H
0000 {4         81  RS232_BASE  DW    4 DUP(?) ; ADDRESSES OF RS232 ADAPTERS
      ????)
      }
0008 {4         82  PRINTER_BASE  DW    4 DUPL(?) ; ADDRESSES OF PRINTERS
      ????)
      }
0010 ?????     83  EQUIP_FLAG   DW    ? ; INSTALLED HARDWARE
0012 ??        84  MFG_TST      DB    ? ; INITIALIZATION FLAG
0013 ?????     85  MEMORY_SIZE  DW    ? ; MEMORY SIZE IN K BYTES
0015 ??        86  MFG_ERR_FLAG DB    ? ; SCRATCHPAD FOR MANUFACTURING
0016 ??        87  ?            DB    ? ; ERROR CODES
88
89  ;-----
90  ;          KEYBOARD DATA AREAS          ;
91  ;-----
92
0017 ??        93  KB_FLAG      DB    ?
94
95  ;----- SHIFT FLAG EQUATES WITHIN KB_FLAG
96
0080           97  INS_STATE    EQU    80H ; INSERT STATE IS ACTIVE
0040           98  CAPS_STATE   EQU    40H ; CAPS LOCK STATE HAS BEEN TOGGLED
0020           99  NUM_STATE    EQU    20H ; NUM LOCK STATE HAS BEEN TOGGLED
0010           100 SCROLL_STATE EQU    10H ; SCROLL LOCK STATE HAS BEEN TOGGLED
0008           101 ALT_SHIFT     EQU    08H ; ALTERNATE SHIFT KEY DEPRESSED
0004           102 CTL_SHIFT     EQU    04H ; CONTROL SHIFT KEY DEPRESSED
0002           103 LEFT_SHIFT    EQU    02H ; LEFT SHIFT KEY DEPRESSED
0001           104 RIGHT_SHIFT   EQU    01H ; RIGHT SHIFT KEY DEPRESSED
105
0018 ??        106 KB_FLAG_1     DB    ? ; SECOND BYTE OF KEYBOARD STATUS
107
0080           108 INS_SHIFT     EQU    80H ; INSERT KEY IS DEPRESSED
0040           109 CAPS_SHIFT    EQU    40H ; CAPS LOCK KEY IS DEPRESSED
0020           110 NUM_SHIFT     EQU    20H ; NUM LOCK KEY IS DEPRESSED
0010           111 SCROLL_SHIFT  EQU    10H ; SCROLL LOCK KEY IS DEPRESSED
0008           112 HOLD_STATE   EQU    08H ; SUSPEND KEY HAS BEEN TOGGLED
113
0019 ??        114 ALT_INPUT     DB    ? ; STORAGE FOR ALTERNATE KEYPAD ENTRY
001A ?????     115 BUFFER_HEAD  DW    ? ; POINTER TO HEAD OF KEYBOARD BUFFER
001C ?????     116 BUFFER_TAIL  DW    ? ; POINTER TO TAIL OF KEYBOARD BUFFER
001E {16       117 KB_BUFFER     DW    16 DUP(?) ; ROOM FOR 15 ENTRIES
      ?????)
003E           118 KB_BUFFER_END LABEL  WORD
119
120 ;----- HEAD = TAIL INDICATES THAT THE BUFFER IS EMPTY
121
0045           122 NUM_KEY      EQU    69 ; SCAN CODE FOR NUMBER LOCK
0046           123 SCROLL_KEY   EQU    70 ; SCROLL LOCK KEY
0038           124 ALT_KEY      EQU    56 ; ALTERNATE SHIFT KEY SCAN CODE
001D           125 CTL_KEY      EQU    29 ; SCAN CODE FOR CONTROL KEY
003A           126 CAPS_KEY     EQU    58 ; SCAN CODE FOR SHIFT LOCK
002A           127 LEFT_KEY     EQU    42 ; SCAN CODE FOR LEFT SHIFT
0036           128 RIGHT_KEY    EQU    54 ; SCAN CODE FOR RIGHT SHIFT
0052           129 INS_KEY      EQU    82 ; SCAN CODE FOR INSERT KEY
0053           130 DEL_KEY      EQU    83 ; SCAN CODE FOR DELETE KEY
131
132 ;-----
133 ;          DISKETTE DATA AREAS          ;
134 ;-----
003E ??        135 SEEK_STATUS  DB    ? ; DRIVE RECALIBRATION STATUS
136 ; BIT 3-0 = DRIVE 3-0 NEEDS RECAL
137 ; BEFORE NEXT SEEK IF BIT IS = 0
138
0080           139 INT_FLAG     EQU    080H ; INTERRUPT OCCURRENCE FLAG
003F ??        140 MOTOR_STATUS DB    ? ; MOTOR STATUS
141 ; BIT 3-0 = DRIVE 3-0 IS CURRENTLY
142 ; RUNNING
143 ; BIT 7 = CURRENT OPERATION IS A WRITE,
144 ; REQUIRES DELAY
145
0040 ??        146 MOTOR_COUNT  DB    ? ; TIME OUT COUNTER FOR DRIVE TURN OFF
0025           147 MOTOR_WAIT   EQU    37 ; 2 SECS OF COUNTS FOR MOTOR TURN OFF
148

```

```

LOC OBJ          LINE    SOURCE
0041 ??         149  DISKETTE_STATUS DB    ?           ; RETURN CODE STATUS BYTE
0080           150  TIME_OUT      EQU   80H          ; ATTACHMENT FAILED TO RESPOND
0040           151  BAD_SEEK     EQU   40H          ; SEEK OPERATION FAILED
0020           152  BAD_NECC    EQU   20H          ; NEC CONTROLLER HAS FAILED
0010           153  BAD_CRC     EQU   10H          ; BAD CRC ON DISKETTE READ
0009           154  DMA_BOUNDARY EQU   09H          ; ATTEMPT TO DMA ACROSS 64K BOUNDARY
0008           155  BAD_DMA     EQU   08H          ; DMA OVERRUN ON OPERATION
0004           156  RECORD_NOT_FND EQU   04H          ; REQUESTED SECTOR NOT FOUND
0003           157  WRITE_PROTECT EQU   03H          ; WRITE ATTEMPTED ON WRITE PROT DISK
0002           158  BAD_ADDR_MARK EQU   02H          ; ADDRESS MARK NOT FOUND
0001           159  BAD_CMD     EQU   01H          ; BAD COMMAND PASSED TO DISKETTE I/O
                160
0042 (7         161  NEC_STATUS   DB    7 DUP(?)      ; STATUS BYTES FROM NEC
??
)
                162
                163  ;-----
                164  ;   VIDEO DISPLAY DATA AREA   :
                165  ;-----
0049 ??         166  CRT_MODE    DB    ?           ; CURRENT CRT MODE
004A ????       167  CRT_COLS   DW    ?           ; NUMBER OF COLUMNS ON SCREEN
004C ????       168  CRT_LEN    DW    ?           ; LENGTH OF REGEN IN BYTES
004E ????       169  CRT_START  DW    ?           ; STARTING ADDRESS IN REGEN BUFFER
0050 (8         170  CURSOR_POSN DW    8 DUP(?)      ; CURSOR FOR EACH OF UP TO 8 PAGES
????
)
0060 ????       171  CURSOR_MODE DW    ?           ; CURRENT CURSOR MODE SETTING
0062 ??         172  ACTIVE_PAGE DB    ?           ; CURRENT PAGE BEING DISPLAYED
0063 ????       173  ADDR_6845 DW    ?           ; BASE ADDRESS FOR ACTIVE DISPLAY CARD
0065 ??         174  CRT_MODE_SET DB    ?           ; CURRENT SETTING OF THE 3x8 REGISTER
0066 ??         175  CRT_PALETTE DB    ?           ; CURRENT PALETTE SETTING COLOR CARD
                176
                177  ;-----
                178  ;   POST DATA AREA           :
                179  ;-----
0067 ????       180  IO_ROM_INIT DW    ?           ; PNTR TO OPTIONAL I/O ROM INIT ROUTINE
0069 ????       181  IO_ROM_SEG DW    ?           ; POINTER TO IO ROM SEGMENT
006B ??         182  INTR_FLAG DB    ?           ; FLAG TO INDICATE AN INTERRUPT HAPPEND
                183
                184  ;-----
                185  ;   TIMER DATA AREA         :
                186  ;-----
006C ????       187  TIMER_LOW  DW    ?           ; LOW WORD OF TIMER COUNT
006E ????       188  TIMER_HIGH DW    ?           ; HIGH WORD OF TIMER COUNT
0070 ??         189  TIMER_OFL  DB    ?           ; TIMER HAS ROLLED OVER SINCE LAST READ
                190  ; COUNTS_SEC    EQU   18
                191  ; COUNTS_MIN    EQU  1092
                192  ; COUNTS_HOUR   EQU  65543
                193  ; COUNTS_DAY    EQU  1573040 = 180080H
                194
                195  ;-----
                196  ;   SYSTEM DATA AREA       :
                197  ;-----
0071 ??         198  BIOS_BREAK  DB    ?           ; BIT 7=1 IF BREAK KEY HAS BEEN HIT
0072 ????       199  RESET_FLAG  DW    ?           ; WORD=1234H IF KEYBOARD RESET UNDERWAY
                200  ;-----
                201  ;   FIXED DISK DATA AREAS :
                202  ;-----
0074 ????       203  ; DW    ?
0076 ????       204  ; DW    ?
                205  ;-----
                206  ;   PRINTER AND RS232 TIME-OUT VARIABLES :
                207  ;-----
0078 (4         208  PRINT_TIM_OUT DB    4 DUP(?)
??
)
007C (4         209  RS232_TIM_OUT DB    4 DUP(?)
??
)
                210  ;-----
                211  ;   ADDITIONAL KEYBOARD DATA AREA :
                212  ;-----
0080 ????       213  BUFFER_START DW    ?
0082 ????       214  BUFFER_END  DW    ?
                215  DATA ENDS
                216  ;-----
                217  ;   EXTRA DATA AREA       :
                218  ;-----

```



```

LOC OBJ          LINE  SOURCE

-----
0000 ??         219  XXDATA  SEGMENT AT 50H
                220  STATUS_BYTE  DB      ?
                221  XXDATA  ENDS
                222  ;-----
                223  ;          VIDEO DISPLAY BUFFER          :
                224  ;-----
0000           225  VIDEO_RAM  SEGMENT AT 0B800H
0000           226  REGENH  LABEL  BYTE
0000           227  REGENH  LABEL  WORD
0000 (16384     228  DB      16384 DUP(?)
    ??
    )

-----
                229  VIDEO_RAM  ENDS
                230  ;-----
                231  ;          ROM RESIDENT CODE          :
                232  ;-----
0000 (57344     233  CODE    SEGMENT AT 0F000H
    ??         234  DB      57344 DUP(?)          ; FILL LOWEST 56K
    )

E000 31353031353132 235
      20434F50522E20 236          DB      '1501512 CPR. IBM 1981'          ; COPYRIGHT NOTICE
      49424D20313938
      32

                237
                238
                239  ;-----
                240  ;          INITIAL RELIABILITY TESTS -- PHASE 1      :
                241  ;-----
                242
                243  ASSUME   CS:CODE,SS:CODE,ES:ABS0,DS:DATA
                244
                245  ;-----
                246  ;          DATA DEFINITIONS          :
                247  ;-----
                248
E016 D7E0       249  C1    DW      C11          ; RETURN ADDRESS
E018 7EE1       250  C2    DW      C24          ; RETURN ADDRESS FOR DUMMY STACK
                251
E01A 204842204F4B 252  F3B   DB      ' KB OK ',13          ; KB FOR MEMORY SIZE
E020 0D

                253
                254  ;-----
                255  ;          LOAD A BLOCK OF TEST CODE THROUGH THE KEYBOARD PORT      :
                256  ;          FOR MANUFACTURING TEST.          :
                257  ;          THIS ROUTINE WILL LOAD A TEST (MAX LENGTH=FAFFH) THROUGH :
                258  ;          THE KEYBOARD PORT. CODE WILL BE LOADED AT LOCATION      :
                259  ;          0000:0500. AFTER LOADING, CONTROL WILL BE TRANSFERRED :
                260  ;          TO LOCATION 0000:0500. STACK WILL BE LOCATED JUST BELOW :
                261  ;          THE TEST CODE. THIS ROUTINE ASSUMES THAT THE FRIST 2      :
                262  ;          BYTES TRANSFERED CONTAIN THE COUNT OF BYTES TO BE LOADED :
                263  ;          (BYTE 1=COUNT LOW, BYTE 2=COUNT HI.)          :
                264  ;-----
                265
                266  ;----- FIRST, GET THE COUNT
                267
E021           268  MFG_BOOT:
E021 E8131A     269  CALL   SP_TEST          ; GET COUNT LOW
E024 8AFB       270  MOV    BH,BL          ; SAVE IT
E026 E80E1A     271  CALL   SP_TEST          ; GET COUNT HI
E029 8AEB       272  MOV    CH,BL
E02B 8ACF       273  MOV    CL,BH          ; CX NOW HAS COUNT
E02D FC         274  CLD                    ; SET DIR. FLAG TO INCRMENT
E02E FA         275  CLI
E02F BF0005     276  MOV    DI,0500H        ; SET TARGET OFFSET (DS=0000)
E032 B0FD       277  MOV    AL,0FDH         ; UNMASK K/B INTERRUPT
E034 E621       278  OUT   INTA01,AL
E036 B00A       279  MOV    AL,0AH          ; SEND R/W INT. REQUEST REG. CMD
E038 E620       280  OUT   INTA00,AL
E03A BA6100     281  MOV    DX,61H          ; SET UP PORT B ADDRESS
E03D BBCC4C     282  MOV    BX,4CCCCH       ; CONTROL BITS FOR PORT B
E040 B402       283  MOV    AH,02H          ; K/B REQUEST PENDING MASK
E042           284  TST:
E042 8AC3       285  MOV    AL,BL
E044 EE         286  OUT   DX,AL          ; TOGGLE K/B CLOCK

```

```

LOC OBJ          LINE    SOURCE

E045 8AC7        287      MOV    AL,BH
E047 EE          288      OUT    DX,AL
E048 4A          289      DEC    DX                      ; POINT DX AT ADDR. 60 (KB DATA)
E049             290      TST1:
E049 E420        291      IN     AL,INTA00              ; GET IRR REG
E04B 22C4        292      AND    AL,AH                  ; KB REQUEST PENDING?
E04D 74FA        293      JZ     TST1                    ; LOOP TILL DATA PRESENT
E04F EC          294      IN     AL,DX                   ; GET DATA
E050 AA          295      STOSB                          ; STORE IT
E051 42          296      INC    DX                      ; POINT DX BACK AT PORT B (61)
E052 E2EE        297      LOOP   TST                      ; LOOP TILL ALL BYTES READ
E054 EA00050000  299      JMP    HFG_TEST_RTN           ; FAR JUMP TO CODE THAT WAS JUST
300                                     ; LOADED
301
302      ;-----
303      ; 8086 PROCESSOR TEST          :
304      ; DESCRIPTION                  :
305      ; VERIFY 8086 FLAGS, REGISTERS :
306      ; AND CONDITIONAL JUMPS      :
307      ;-----
308      ASSUME CS:CODE,DS:NOTHING,ES:NOTHING,SS:NOTHING
E05B             309      ORG    0E05BH
E05B             310      RESET LABEL FAR
E05B FA          311      START: CLI                    ; DISABLE INTERRUPTS
E05C B4D5        312      MOV    AH,0D5H                ; SET SF, CF, ZF, AND AF FLAGS ON
E05E 9E          313      SAHF                          ;
E05F 734C        314      JNC    ERR01                  ; GO TO ERR ROUTINE IF CF NOT SET
E061 754A        315      JNZ    ERR01                  ; GO TO ERR ROUTINE IF ZF NOT SET
E063 7B48        316      JNP    ERR01                  ; GO TO ERR ROUTINE IF PF NOT SET
E065 7946        317      JNS    ERR01                  ; GO TO ERR ROUTINE IF SF NOT SET
E067 9F          318      LAHF                          ; LOAD FLAG IMAGE TO AH
E068 B105        319      MOV    CL,5                    ; LOAD CNT REG WITH SHIFT CNT
E06A D2EC        320      SHR    AH,CL                  ; SHIFT AF INTO CARRY BIT POS
E06C 733F        321      JNC    ERR01                  ; GO TO ERR ROUTINE IF AF NOT SET
E06E B040        322      MOV    AL,40H                 ; SET THE OF FLAG ON
E070 D0E0        323      SHL    AL,1                    ; SETUP FOR TESTING
E072 7139        324      JNO    ERR01                  ; GO TO ERR ROUTINE IF OF NOT SET
E074 32E4        325      XOR    AH,AH                  ; SET AH = 0
E076 9E          326      SAHF                          ; CLEAR SF, CF, ZF, AND PF
E077 7634        327      JBE    ERR01                  ; GO TO ERR ROUTINE IF CF ON
328                                     ; GO TO ERR ROUTINE IF ZF ON
E079 7832        329      JS     ERR01                  ; GO TO ERR ROUTINE IF SF ON
E07B 7A30        330      JP     ERR01                  ; GO TO ERR ROUTINE IF PF ON
E07D 9F          331      LAHF                          ; LOAD FLAG IMAGE TO AH
E07E B105        332      MOV    CL,5                    ; LOAD CNT REG WITH SHIFT CNT
E080 D2EC        333      SHR    AH,CL                  ; SHIFT AF' INTO CARRY BIT POS
E082 7229        334      JC     ERR01                  ; GO TO ERR ROUTINE IF ON
E084 D0E4        335      SHL    AH,1                    ; CHECK THAT OF' IS CLEAR
E086 7025        336      JO     ERR01                  ; GO TO ERR ROUTINE IF ON
337
338      ;---- READ/WRITE THE 8086 GENERAL AND SEGMENTATION REGISTERS
339      ; WITH ALL ONE'S AND ZEROES'S.
340
E088 B8FFFF      341      MOV    AX,0FFFFH              ; SETUP ONE'S PATTERN IN AX
E08B F9          342      STC                          ;
E08C 8ED8        343      C8:  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 8BFE        352      MOV    DI,SI
E0A0 7307        353      JNC    C9                      ; TST1A
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:  ; TST1A
E0A9 0BC7        359      OR     AX,DI                  ; ZERO PATTERN MAKE IT THRU?
E0AB 7401        360      JZ     C10                    ; YES - GO TO NEXT TEST
E0AD F4          361      ERR01: HLT                    ; HALT SYSTEM
362      ;-----
363      ; ROS CHECKSUM TEST I      :

```

```

LOC OBJ          LINE   SOURCE
364             ; DESCRIPTION                               :
365             ;   A CHECKSUM IS DONE FOR THE 8K          :
366             ;   ROS MODULE CONTAINING POD AND        :
367             ;   BIOS.                                  :
368             ;-----:
EOAE            369     C10:
370             ; ZERO IN AL ALREADY
EOAE E6A0       371     OUT    0A0H,AL          ; DISABLE NMI INTERRUPTS
EOB0 E6B3       372     OUT    83H,AL          ; INITIALIZE DMA PAGE REG.
EOB2 BAD803     373     MOV    DX,308H
EOB5 EE         374     OUT    DX,AL          ; DISABLE COLOR VIDEO
EOB6 FEC0       375     INC    AL
EOB8 B2BB       376     MOV    DL,0B8H
EOBA EE         377     OUT    DX,AL          ; DISABLE B/W VIDEO,EN HIGH RES
EOBB B089       378     MOV    AL,89H          ; SET 8255 FOR B,A=OUT, C=IN
EOBD E643       379     OUT    CMD_PORT,AL
EOBF B0A5       380     MOV    AL,10100101B
381
EOC1 E661       382     OUT    PORT_B,AL      ; ENABLE PARITY CHECKERS AND
383             ; PULL KB CLOCK HI, TRI-STATE
384             ; KEYBOARD INPUTS,ENABLE HIGH
385             ; BANK OF SWITCHES->PORT C(0-3)
EOC3 B001       385     MOV    AL,01H          ; <><><><><><><><><><><><><><>
EOC5 E660       386     OUT    PORT_A,AL      ; <><><>CHECKPOINT 1<><><>
EOC7 8CC8       387     MOV    AX,CS          ; SETUP SS SEG REG
EOC9 8ED0       388     MOV    SS,AX
EOCB 8ED8       389     MOV    DS,AX          ; SET UP DATA SEG TO POINT TO
390             ; ROM ADDRESS
EOCD FC         391     CLD                    ; SET DIRECTION FLAG TO INC.
392     ASSUME SS:CODE
EOCE BB00E0     393     MOV    BX,0E000H        ; SETUP STARTING ROS ADDR
EO01 BC16E0     394     MOV    SP,OFFSET C1    ; SETUP RETURN ADDRESS
EO04 E91B18     395     JMP    ROS_CHECKSUM
EO07 75D4       396     C11: JNE    ERROR1      ; HALT SYSTEM IF ERROR
397             ;-----:
398             ;   8237 DMA INITIALIZATION CHANNEL REGISTER TEST :
399             ; DESCRIPTION :
400             ;   DISABLE THE 8237 DMA CONTROLLER. VERIFY THAT :
401             ;   TIMER 1 FUNCTIONS OK. WRITE/READ THE CURRENT :
402             ;   ADDRESS AND WORD COUNT REGISTERS FOR ALL :
403             ;   CHANNELS. INITIALIZE AND START DMA FOR MEMORY :
404             ;   REFRESH. :
405             ;-----:
406
407             ;---- DISABLE DMA CONTROLLER
408
EO09 B002       409     MOV    AL,02H          ; <><><><><><><><><><><><><><>
EO0B E660       410     OUT    PORT_A,AL      ; <><><>CHECKPOINT 2<><><>
EO0D B004       411     MOV    AL,04          ; DISABLE DMA CONTROLLER
EO0F E608       412     OUT    DMA08,AL
413
414             ;---- VERIFY THAT TIMER 1 FUNCTIONS OK
415
EOE1 B054       416     MOV    AL,54H          ; SEL TIMER 1,LSB,MODE 2
EOE3 E643       417     OUT    TIMER+3,AL
EOE5 8AC1       418     MOV    AL,CL          ; SET INITIAL TIMER CNT TO 0
EOE7 E641       419     OUT    TIMER+1,AL
EOE9            420     C12:                    ; TIMER1_BITS_ON
EOE9 B040       421     MOV    AL,40H          ; LATCH TIMER 1 COUNT
EOEB E643       422     OUT    TIMER+3,AL
EOED 80FBFF     423     CMP    BL,OFFH        ; YES - SEE IF ALL BITS GO OFF
EOEF 7407       424     JE    C13             ; TIMER1_BITS_OFF
EOF2 E441       425     IN    AL,TIMER+1      ; READ TIMER 1 COUNT
EOF4 0ADB       426     OR    BL,AL           ; ALL BITS ON IN TIMER
EOF6 E2F1       427     LOOP  C12            ; TIMER1_BITS_ON
EOF8 F4         428     HLT                    ; TIMER 1 FAILURE, HALT SYS
EOF9            429     C13:                    ; TIMER1_BITS_OFF
EOF9 8AC3       430     MOV    AL,BL           ; SET TIMER 1 CNT
EOFB 2BC9       431     SUB    CX,CX
E0FD E641       432     OUT    TIMER+1,AL
E0FF            433     C14:                    ; TIMER_LOOP
E0FF B040       434     MOV    AL,40H          ; LATCH TIMER 1 COUNT
E101 E643       435     OUT    TIMER+3,AL
E103 90         436     NOP                    ; DELAY FOR TIMER
E104 90         437     NOP
E105 E441       438     IN    AL,TIMER+1      ; READ TIMER 1 COUNT
E107 22D8       439     AND    BL,AL
E109 7403       440     JZ    C15             ; WRAP_DMA_REG

```



```

LOC OBJ                LINE  SOURCE
E1FA B91000           593      MOV     CX,16
E1FD A5               594      D3A:  MOVSW          ; MOVE VECTOR TABLE TO RAM
E1FE 47               595      INC     DI           ; SKIP SEGMENT POINTER
E1FF 47               596      INC     DI
E200 E2FB            597      LOOP   D3A
598      ;-----
599      ;   DETERMINE CONFIGURATION AND MFG. MODE   :
600      ;-----
601
E202 1F               602      POP     DS
E203 1E               603      PUSH   DS           ; RECOVER DATA SEG
E204 E462             604      IN     AL,PORT_C    ; GET SWITCH INFO
E206 240F             605      AND    AL,00001111B ; ISOLATE SWITCHES
E208 8AE0             606      MOV    AH,AL        ; SAVE
E20A B0AD             607      MOV    AL,10101101B ; ENABLE OTHER BANK OF SWS.
E20C E661             608      OUT   PORT_B,AL
E20E 90               609      NOP
E20F E462             610      IN     AL,PORT_C
E211 B104             611      MOV    CL,4
E213 D2C0             612      ROL   AL,CL         ; ROTATE TO HIGH NIBBLE
E215 24F0             613      AND    AL,11110000B ; ISOLATE
E217 0AC4             614      OR    AL,AH         ; COMBINE WITH OTHER BANK
E219 2AE4             615      SUB   AH,AH
E21B A31004           616      MOV    DATA_WORD[OFFSET EQUIP_FLAG],AX ; SAVE SWITCH INFO
E21E B099             617      MOV    AL,99H
E220 E663             618      OUT   CMD_PORT,AL
E222 E80518           619      CALL  KBD_RESET     ; SEE IF MFG. JUMPER IN
E225 80FBAA           620      CMP   BL,0AAH       ; KEYBOARD PRESENT?
E228 7418             621      JE    E6
E22A 80FB65           622      CMP   BL,065H       ; LOAD MFG. TEST REQUEST?
E22D 7503             623      JNE   D3B
E22F E9EFFF           624      JMP   MFG_BOOT      ; GO TO BOOTSTRAP IF SO
E232 B038             625      D3B:  MOV    AL,38H
E234 E661             626      OUT   PORT_B,AL
E236 90               627      NOP
E237 90               628      NOP
E238 E460             629      IN     AL,PORT_A
E23A 24FF             630      AND   AL,0FFH       ; WAS DATA LINE GROUNDED
E23C 7504             631      JNZ   E6
E23E FE061204         632      DATA_AREA[OFFSET MFG_TST] ; SET MANUFACTURING TEST FLAG
633
634      ;-----
635      ;   INITIALIZE AND START CRT CONTROLLER (6845) :
636      ;   TEST VIDEO READ/WRITE STORAGE.           :
637      ;   DESCRIPTION                               :
638      ;   RESET THE VIDEO ENABLE SIGNAL.           :
639      ;   SELECT ALPHANUMERIC MODE, 40 # 25, B & W. :
640      ;   READ/WRITE DATA PATTERNS TO STG. CHECK STG :
641      ;   ADDRESSABILITY.                           :
642      ;   ERROR = 1 LONG AND 2 SHORT BEEPS        :
643      ;-----
E242                 644      E6:
E242 A11004           645      MOV    AX,DATA_WORD[OFFSET EQUIP_FLAG] ; GET SENSE SWITCH INFO
E245 50               646      PUSH  AX           ; SAVE IT
E246 B030             647      MOV    AL,30H
E248 A31004           648      MOV    DATA_WORD[OFFSET EQUIP_FLAG],AX
E24B 2AE4             649      SUB   AH,AH
E24D CD10             650      INT   10H         ; SEND INIT TO B/W CARD
E24F B020             651      MOV    AL,20H
E251 A31004           652      MOV    DATA_WORD[OFFSET EQUIP_FLAG],AX
E254 2AE4             653      SUB   AH,AH        ; AND INIT COLOR CARD
E256 CD10             654      INT   10H
E258 58               655      POP   AX           ; RECOVER REAL SWITCH INFO
E259 A31004           656      MOV    DATA_WORD[OFFSET EQUIP_FLAG],AX ; RESTORE IT
657      ; AND CONTINUE
E25C 2430             658      AND   AL,30H       ; ISOLATE VIDEO SWS
E25E 750A             659      JNZ   E7           ; VIDEO SWS SET TO 0?
E260 BF4000           660      MOV   DI,OFFSET VIDEO_INT ; SET INT 10H TO DUMMY
E263 C7054BFF         661      MOV   [DI],OFFSET DUMMY_RETURN ; RETURN IF NO VIDEO CARD
E267 E9A000           662      JMP   E18_1        ; BYPASS VIDEO TEST
E26A                 663      E7:
E26A 3C30             664      CMP   AL,30H       ; TEST_VIDEO:
E26C 7408             665      JE    E8           ; B/W CARD ATTACHED?
E26E FEC4             666      INC  AH           ; YES - SET MODE FOR B/W CARD
E270 3C20             667      CMP   AL,20H       ; SET COLOR MODE FOR COLOR CD
E272 7502             668      JNE   E8           ; 80X25 MODE SELECTED?
E274 B403             669      MOV   AH,3         ; NO - SET MODE FOR 40X25
                       ; SET MODE FOR 80X25

```

```

LOC OBJ          LINE  SOURCE
E276 86E0        670  E8:  XCHG  AH,AL          ; SET_MODE:
E278 50          671      PUSH  AX              ; SAVE VIDEO MODE ON STACK
E279 2AE4        672      SUB   AH,AH          ; INITIALIZE TO ALPHANUMERIC MD
E27B CD10        673      INT  10H            ; CALL VIDEO_IO
E27D 58          674      POP   AX             ; RESTORE VIDEO SENSE SMS IN AH
E27E 50          675      PUSH  AX             ; RESAVE VALUE
E27F BB00B0      676      MOV   BX,0B000H     ; BEG VIDEO RAM ADDR B/W CD
E282 BAB803      677      MOV   DX,3B8H       ; MODE REG FOR B/W
E285 B90008      678      MOV   CX,2048        ; RAM WORD CNT FOR B/W CD
E288 B001        679      MOV   AL,1           ; SET MODE FOR BW CARD
E28A 80FC30      680      CMP   AH,30H         ; B/W VIDEO CARD ATTACHED?
E28D 7409        681      JE    E9             ; YES - GO TEST VIDEO STG
E28F B7B8        682      MOV   BH,0B0H        ; BEG VIDEO RAM ADDR COLOR CD
E291 BAD803      683      MOV   DX,3D8H        ; MODE REG FOR COLOR CD
E294 B520        684      MOV   CH,20H         ; RAM WORD CNT FOR COLOR CD
E296 FEC8        685      DEC   AL             ; SET MODE TO 0 FOR COLOR CD
E298            686  E9:      ; TEST_VIDEO_STG:
E298 EE          687      OUT  DX,AL          ; DISABLE VIDEO FOR COLOR CD
E299 813E72043412 688      CMP   DATA_WORD[OFFSET RESET_FLAG],1234H ; POD INIT BY KBD RESET?
E29F 8EC3        689      MOV   ES,BX          ; POINT ES TO VIDEO RAM STG
E2A1 7407        690      JE    E10            ; YES - SKIP VIDEO RAM TEST
E2A3 8EDB        691      MOV   DS,BX          ; POINT DS TO VIDEO RAM STG
E2A5 E0C703      692      ASSUME DS=NOTHING,ES=NOTHING
E2A8 7546        693      CALL STGTEST_CNT    ; GO TEST VIDEO R/W STG
E2A8 7546        694      JNE  E17             ; R/W STG FAILURE - BEEP SPK
695  ;-----
696  ;   SETUP VIDEO DATA ON SCREEN FOR VIDEO   :
697  ;   LINE TEST.                             :
698  ; DESCRIPTION                               :
699  ;   ENABLE VIDEO SIGNAL AND SET MODE.       :
700  ;   DISPLAY A HORIZONTAL BAR ON SCREEN.     :
701  ;-----
E2AA            702  E10:
E2AA 58          703      POP   AX             ; GET VIDEO SENSE SMS (AH)
E2AB 50          704      PUSH  AX             ; SAVE IT
E2AC B400        705      MOV   AH,0           ; ENABLE VIDEO AND SET MODE
E2AE CD10        706      INT  10H            ; VIDEO
E2B0 B82070      707      MOV   AX,7020H      ; WRT BLANKS IN REVERSE VIDEO
708
709
710
E2B3 EB11        711      JMP   SHORT E10A
E2C3            712      ORG  0E2C3H
E2C3 E99915      713      JMP   NMI_INT
714
E2C6            715  E10A:
E2C6 2BFF        716      SUB   DI,DI          ; SETUP STARTING LOC
E2CB B92800      717      MOV   CX,40          ; NO. OF BLANKS TO DISPLAY
E2CB F3          718      REP  STOSW          ; WRITE VIDEO STORAGE
E2CC AB
719  ;-----
720  ;   CRT INTERFACE LINES TEST               :
721  ; DESCRIPTION                               :
722  ;   SENSE ON/OFF TRANSITION OF THE        :
723  ;   VIDEO ENABLE AND HORIZONTAL           :
724  ;   SYNC LINES.                           :
725  ;-----
E2CD 58          726      POP   AX             ; GET VIDEO SENSE SW INFO
E2CE 50          727      PUSH  AX             ; SAVE IT
E2CF 80FC30      728      CMP   AH,30H         ; B/W CARD ATTACHED?
E2D2 BABA03      729      MOV   DX,03BAH       ; SETUP ADDR OF BW STATUS PORT
E2D5 7403        730      JE    E11            ; YES - GO TEST LINES
E2D7 BADA03      731      MOV   DX,03DAH       ; COLOR CARD IS ATTACHED
E2DA            732  E11:      ; LINE_TST:
E2DA B408        733      MOV   AH,8           ;
E2DC            734  E12:      ; OFLOOP_CNT:
E2DC 2BC9        735      SUB   CX,CX
E2DE            736  E13:
E2DE EC          737      IN   AL,DX          ; READ CRT STATUS PORT
E2DF 22C4        738      AND  AL,AH          ; CHECK VIDEO/HORZ LINE
E2E1 7504        739      JNZ  E14            ; ITS ON - CHECK IF IT GOES OFF
E2E3 E2F9        740      LOOP E13            ; LOOP TILL ON OR TIMEOUT
E2E5 EB09        741      JMP  SHORT E17      ; GO PRINT ERROR MSG
E2E7            742  E14:
E2E7 2BC9        743      SUB   CX,CX
E2E9            744  E15:
E2E9 EC          745      IN   AL,DX          ; READ CRT STATUS PORT

```


LOC OBJ	LINE	SOURCE
E3CC E2FE	900	LOOP F5 ; DELAY FOR A WHILE
E3CE E460	901	IN AL,KBD_IN ; CHECK FOR STUCK KEYS
E3D0 3C00	902	CMF AL,0 ; SCAN CODE = 0?
E3D2 740A	903	JE F7 ; YES - CONTINUE TESTING
E3D4 E8B415	904	CALL XPC_BYTE ; CONVERT AND PRINT
E3D7	905	F6:
E3D7 BE4CEC90	906	MOV SI,OFFSET F1 ; GET MSG ADDR
E3DB E8CB15	907	CALL E_MSG ; PRINT MSG ON SCREEN
	908	;-----
	909	; SETUP HARDWARE INT. VECTOR TABLE ;
	910	;-----
E3DE	911	F7:
E3DE 1E	912	PUSH DS ; SETUP_INT_TABLE:
E3DF 2BC0	913	SUB AX,AX
E3E1 8EC0	914	MOV ES,AX
E3E3 B90800	915	MOV CX,08 ; GET VECTOR CNT
E3E6 0E	916	PUSH CS ; SETUP DS SEG REG
E3E7 1F	917	POP DS
E3E8 BEF3FE90	918	MOV SI,OFFSET VECTOR_TABLE
E3EC BF2000	919	MOV DI,OFFSET INT_PTR
E3EF	920	F7A:
E3EF A5	921	MOVSW
E3F0 47	922	INC DI ; SKIP OVER SEGMENT
E3F1 47	923	INC DI
E3F2 E2FB	924	LOOP F7A
E3F4 1F	925	POP DS
	926	
	927	;---- SET UP OTHER INTERRUPTS AS NECESSARY
	928	
E3F5 C70608005FF8	929	MOV NMI_PTR,OFFSET NMI_INT ; NMI INTERRUPT
E3FB C7061400054FF	930	MOV INT5_PTR,OFFSET PRINT_SCREEN ; PRINT SCREEN
E401 C706620000F6	931	MOV BASIC_PTR+2,0F600H ; SEGMENT FOR CASSETTE BASIC
	932	
	933	;---- SETUP TIMER 0 TO BLINK LED IF MANUFACTURING TEST MODE
	934	
E407 803E120401	935	CMF DATA_AREA(OFFSET MFG_TST1,01H) ; MFG. TEST MODE?
E40C 750A	936	JNZ EXP_IO
E40E C70670003CF9	937	MOV WORD PRT1(CH*4),OFFSET BLINK_INT; SETUP TIMER INTR TO BLINK LED
E414 B0FE	938	MOV AL,0FEH ; ENABLE TIMER INTERRUPT
E416 E621	939	OUT INTA01,AL
	940	;-----
	941	; EXPANSION I/O BOX TEST ;
	942	; CHECK TO SEE IF EXPANSION BOX PRESENT - IF INSTALLED, ;
	943	; TEST DATA AND ADDRESS BUSES TO I/O BOX ;
	944	; ERROR='1601' ;
	945	;-----
	946	
	947	;---- DETERMINE IF BOX IS PRESENT
	948	
E418	949	EXP_IO:
E418 BA1002	950	MOV DX,0210H ; (CARD WAS ENABLED EARLIER)
E41B B85555	951	MOV AX,5555H ; CONTROL PORT ADDRESS
E41E EE	952	OUT DX,AL ; SET DATA PATTERN
E41F B001	953	MOV AL,01H ; MAKE AL DIFFERENT
E421 EC	954	IN AL,DX ; RECOVER DATA
E422 3AC4	955	CMF AL,AH ; REPLY?
E424 7544	956	JNE E19 ; NO RESPONSE, GO TO NEXT TEST
E426 F7D0	957	NOT AX ; MAKE DATA=AAAA
E428 EE	958	OUT DX,AL
E429 B001	959	MOV AL,01H
E42B EC	960	IN AL,DX ; RECOVER DATA
E42C 3AC4	961	CMF AL,AH
E42E 753A	962	JNE E19
	963	
	964	;---- CHECK ADDRESS BUS
	965	
E430	966	EXP2:
E430 BB0100	967	MOV BX,0001H
E433 BA1502	968	MOV DX,0215H ; LOAD HI ADDR. REG ADDRESS
E436 B91000	969	MOV CX,0016 ; GO ACROSS 16 BITS
E439	970	EXP3:
E439 2E8807	971	MOV CS:[BX],AL ; WRITE ADDRESS F0000+BX
E43C 90	972	NOP
E43D EC	973	IN AL,DX ; READ ADDR. HIGH
E43E 3AC7	974	CMF AL,BH
E440 7521	975	JNE EXP_ERR ; GO ERROR IF HISCOMPARE
E442 42	976	INC DX ; DX=216H (ADDR. LOW REG)

LOC OBJ	LINE	SOURCE	
E443 EC	977	IN	AL,DX
E444 3AC3	978	CMF	AL,BL ; COMPARE TO LOW ADDRESS
E446 751B	979	JNE	EXP_ERR
E448 4A	980	DEC	DX ; DX BACK TO 215H
E449 DIE3	981	SHL	BX,1
E448 E2EC	982	LOOP	EXP3 ; LOOP TILL '1' WALKS ACROSS BX
	983		
	984	;----- CHECK DATA BUS	
	985		
E440 B90800	986	MOV	CX,0008 ; DO 8 TIMES
E450 B001	987	MOV	AL,01
E452 4A	988	DEC	DX ; MAKE DX=214H (DATA BUS REG)
E453	989	EXP4:	
E453 8AE0	990	MOV	AH,AL ; SAVE DATA BUS VALUE
E455 EE	991	OUT	DX,AL ; SEND VALUE TO REG
E456 B001	992	MOV	AL,01H
E458 EC	993	IN	AL,DX ; RETRIEVE VALUE FROM REG
E459 3AC4	994	CMF	AL,AH ; = TO SAVED VALUE
E458 7506	995	JNE	SHORT EXP_ERR
E450 D0E0	996	SHL	AL,1 ; FORM NEW DATA PATTERN
E45F E2F2	997	LOOP	EXP4 ; LOOP TILL BIT WALKS ACROSS AL
E461 EB07	998	JMP	SHORT E19 ; GO ON TO NEXT TEST
E463	999	EXP_ERR:	
E463 BE0FF990	1000	MOV	SI,OFFSET F3C
E467 E03F15	1001	CALL	E_MSG
	1002	;-----	
	1003	; ADDITIONAL READ/WRITE STORAGE TEST ;	
	1004	; DESCRIPTION ;	
	1005	; WRITE/READ DATA PATTERNS TO ANY READ/WRITE ;	
	1006	; STORAGE AFTER THE FIRST 32K. STORAGE ;	
	1007	; ADDRESSABILITY IS CHECKED. ;	
	1008	;-----	
	1009	ASSUME	DS:DATA
E46A	1010	E19:	
E46A E0EC15	1011	CALL	DDS
E460 1E	1012	PUSH	DS
E46E	1013	E20:	
E46E 013E72003412	1014	CMF	RESET_FLAG,1234H ; HARM START?
E474 7503	1015	JNE	E20A ; CONTINUE TEST IF NOT
E476 E99F00	1016	JMP	ROM_SCAN ; GO TO NEXT ROUTINE IF SO
E479	1017	E20A:	
E479 B81000	1018	MOV	AX,16 ; STARTING AMT. OF MEMORY OK
E47C EB28	1019	JMP	SHORT PRT_SIZ ; POST MESSAGE
E47E	1020	E20B:	
E47E 8B1E1300	1021	MOV	BX,MEMORY_SIZE ; GET MEM. SIZE WORD
E482 03EB10	1022	SUB	BX,16 ; 1ST 16K ALREADY DONE
E485 B104	1023	MOV	CL,04H
E487 D3EB	1024	SHR	BX,CL ; DIVIDE BY 16
E489 8BCB	1025	MOV	CX,BX ; SAVE COUNT OF 16K BLOCKS
E48B BB0004	1026	MOV	BX,0400H ; SET PTR. TO RAM SEGMENT>16K
E48E	1027	E21:	
E48E 8EDB	1028	MOV	DS,BX ; SET SEG. REG
E490 8EC3	1029	MOV	ES,BX
E492 01C30004	1030	ADD	BX,0400H ; POINT TO NEXT 16K
E496 52	1031	PUSH	DX
E497 51	1032	PUSH	CX ; SAVE WORK REGS
E498 53	1033	PUSH	BX
E499 50	1034	PUSH	AX
E49A B90020	1035	MOV	CX,2000H ; SET COUNT FOR 8K WORDS
E49D E0CF01	1036	CALL	STGTST_CNT
E4A0 754C	1037	JNZ	E21A ; GO PRINT ERROR
E4A2 58	1038	POP	AX ; RECOVER TESTED MEM NUMBER
E4A3 051000	1039	ADD	AX,16
E4A6	1040	PRT_SIZ:	
E4A6 50	1041	PUSH	AX
E4A7 BB0A00	1042	MOV	BX,10 ; SET UP FOR DECIMAL CONVERT
E4AA B90300	1043	MOV	CX,3 ; OF 3 NIBBLES
E4AD	1044	DECIMAL_LOOP:	
E4AD 33D2	1045	XOR	DX,DX
E4AF F7F3	1046	DIV	BX ; DIVIDE BY 10
E4B1 80CA30	1047	OR	DL,30H ; MAKE INTO ASCII
E4B4 52	1048	PUSH	DX ; SAVE
E4B5 E2F6	1049	LOOP	DECIMAL_LOOP
E4B7 B90300	1050	MOV	CX,3
E4BA	1051	PRT_DEC_LOOP:	
E4BA 58	1052	POP	AX ; RECOVER A NUMBER
E4BB E0DE14	1053	CALL	PRT_HEX

Appendix A

LOC OBJ	LINE	SOURCE	
E530	1131	E4:	
E53D 26DB	1132	SUB BX,BX	; SETUP STARTING ROS ADDR
E53F 8EDA	1133	MOV DS,DX	
	1134		; CHECK ROS
E541 E8AE13	1135	CALL ROS_CHECKSUM	
E544 7403	1136	JE E5	; CONTINUE IF OK
E546 E88201	1137	CALL ROM_ERR	; POST ERROR
E549	1138		
E549 81C20002	1139	E5:	
E54D FECC	1140	ADD DX,0200H	; POINT TO NEXT 8K MODULE
E54F 75EC	1141	DEC AH	; ANY MORE TO DO?
	1141	JNZ E4	; YES - CONTINUE
	1142		
	1143	;-----	
	1143	; DISKETTE ATTACHMENT TEST	:
	1144	; DESCRIPTION	:
	1145	; CHECK IF IPL DISKETTE DRIVE IS ATTACHED TO SYSTEM. IF	:
	1146	; ATTACHED, VERIFY STATUS OF NEC FDC AFTER A RESET. ISSUE	:
	1147	; A RECAL AND SEEK CMD TO FDC AND CHECK STATUS. COMPLETE	:
	1148	; SYSTEM INITIALIZATION THEN PASS CONTROL TO THE BOOT	:
	1149	; LOADER PROGRAM.	:
	1150	;-----	
E551	1151	F9:	
E551 IF	1152	POP DS	
E552 A01000	1153	MOV AL,BYTE PTR EQUIP_FLAG	; DISKETTE PRESENT?
E555 2401	1154	AND AL,01H	; NO - BYPASS DISKETTE TEST
E557 743E	1155	JZ F15	
E559	1156	F10:	; DISK_TEST:
E559 E421	1157	IN AL,INTA01	
E55B 24BF	1158	AND AL,0BFH	; ENABLE DISKETTE INTERRUPTS
E55D E621	1159	OUT INTA01,AL	
E55F B400	1160	MOV AH,0	; RESET NEC FDC
E561 8AD4	1161	MOV DL,AH	; SET FOR DRIVE 0
E563 CD13	1162	INT 13H	; VERIFY STATUS AFTER RESET
E565 F6C4FF	1163	TEST AH,0FFH	; STATUS OK?
E568 7520	1164	JNZ F13	; NO - FDC FAILED
	1165		
	1166	;-----	
	1166	TURN DRIVE 0 MOTOR ON	
	1167		
E56A BAF203	1168	MOV DX,03F2H	; GET ADDR OF FDC CARD
E56D B01C	1169	MOV AL,1CH	; TURN MOTOR ON, EN DMA/INT
E56F EE	1170	OUT DX,AL	; WRITE FDC CONTROL REG
E570 2BC9	1171	SUB CX,CX	
E572	1172	F11:	; MOTOR_WAIT:
E572 E2FE	1173	LOOP F11	; WAIT FOR 1 SECOND
E574	1174	F12:	; MOTOR_WAIT1:
E574 E2FE	1175	LOOP F12	
E576 3302	1176	XOR DX,DX	; SELECT DRIVE 0
E578 B501	1177	MOV CH,1	; SELECT TRACK 1
E57A 88163F00	1178	MOV SEEK_STATUS,DL	
E57E E8FC08	1179	CALL SEEK	; RECALIBRATE DISKETTE
E581 7207	1180	JC F13	; GO TO ERR SUBROUTINE IF ERR
E583 B522	1181	MOV CH,34	; SELECT TRACK 34
E585 E8F508	1182	CALL SEEK	; SEEK TO TRACK 34
E588 7307	1183	JNC F14	; OK, TURN MOTOR OFF
E58A	1184	F13:	; DSK_ERR:
E58A BE52EC90	1185	MOV SI,OFFSET F3	; GET ADDR OF MSG
E58E E81814	1186	CALL E_MSG	; GO PRINT ERROR MSG
	1187		
	1188	;-----	
	1188	TURN DRIVE 0 MOTOR OFF	
	1189		
E591	1190	F14:	; DR0_OFF:
E591 B00C	1191	MOV AL,0CH	; TURN DRIVE 0 MOTOR OFF
E593 BAF203	1192	MOV DX,03F2H	; FDC CTL ADDRESS
E596 EE	1193	OUT DX,AL	
	1194		
	1195	;-----	
	1195	SETUP PRINTER AND RS232 BASE ADDRESSES IF DEVICE ATTACHED	
	1196		
E597	1197	F15:	
E597 C606B0000	1198	MOV INTR_FLAG,00H	; SET STRAY INTERRUPT FLAG = 00
E59C BE1E00	1199	SI,OFFSET KB_BUFFER	; SETUP KEYBOARD PARAMETERS
E59F 89361A00	1200	MOV BUFFER_HEAD,SI	
E5A3 89361C00	1201	MOV BUFFER_TAIL,SI	
E5A7 89368000	1202	MOV BUFFER_START,SI	
E5AB 83C620	1203	ADD SI,32	;DEFAULT BUFFER OF 32 BYTES
E5AE 89368200	1204	MOV BUFFER_END,SI	
E5B2 BF7800	1205	MOV DI,OFFSET PRINT_TIM_OUT	;SET DEFAULT PRINTER TIMEOUT
E5B5 1E	1206	PUSH DS	
E5B6 07	1207	POP ES	

LOC	OBJ	LINE	SOURCE	
ESB7	B81414	1208	MOV AX,1414H	; DEFAULT=20
ESBA	AB	1209	STOSW	
ESBB	AB	1210	STOSW	
ESBC	B80101	1211	MOV AX,0101H	;RS232 DEFAULT=01
ESBF	AB	1212	STOSW	
ESCO	AB	1213	STOSW	
ESC1	E421	1214	IN AL,INTA01	
ESC3	24FC	1215	AND AL,0FCH	; ENABLE TIMER AND KB INTS
ESC5	E621	1216	OUT INTA01,AL	
ESC7	83F000	1217	CMF BP,0000H	; CHECK FOR BP= NON-ZERO
		1218		; (ERROR HAPPENED)
E5CA	7419	1219	JE F15A_0	; CONTINUE IF NO ERROR
E5CC	BA0200	1220	MOV DX,2	; 2 SHORT BEEPS (ERROR)
E5CF	E80614	1221	CALL ERR_BEEP	
E5D2	BE09E890	1222	MOV SI,OFFSET F3D	; LOAD ERROR MSG
E5D6	E8F113	1223	CALL P_MSG	
E5D9		1224	ERR_WAIT:	
E5D9	B400	1225	MOV AH,00	
E5DB	CD16	1226	INT 16H	; WAIT FOR 'F1' KEY
E5DD	80FC3B	1227	CMF AH,3BH	
E5E0	75F7	1228	JNE ERR_WAIT	
E5E2	EB0E90	1229	JMP F15A	; BYPASS ERROR
E5E5		1230	F15A_0:	
E5E5	803E120001	1231	CMF MFG_TST,1	; MFG MODE
E5EA	7406	1232	JE F15A	; BYPASS BEEP
E5EC	BA0100	1233	MOV DX,1	; 1 SHORT BEEP (NO ERRORS)
E5EF	E8E613	1234	CALL ERR_BEEP	
E5F2	A01000	1235	F15A: MOV AL,BYTE PTR EQUIP_FLAG	; GET SWITCHES
E5F5	2401	1236	AND AL,00000001B	; 'LOOP POST' SWITCH ON
E5F7	7503	1237	JNZ F15B	; CONTINUE WITH BRING-UP
E5F9	E95FFA	1238	JMP START	
E5FC	2AE4	1239	F15B: SUB AH,AH	
E5FE	A04900	1240	MOV AL,CRT_MODE	
E601	CD10	1241	INT 10H	; CLEAR SCREEN
E603		1242	F15C:	
E603	BDA3F990	1243	MOV BP,OFFSET F4	; PRT_SRC_TBL
E607	BE0000	1244	MOV SI,0	
E60A		1245	F16:	; PRT_BASE:
E60A	2E8B5600	1246	MOV DX,CS:[BP]	; GET PRINTER BASE ADDR
E60E	BOAA	1247	MOV AL,0AAH	; WRITE DATA TO PORT A
E610	EE	1248	OUT DX,AL	
E611	1E	1249	PUSH DS	; BUS SETTLEING
E612	EC	1250	IN AL,DX	; READ PORT A
E613	1F	1251	POP DS	
E614	3CAA	1252	CMF AL,0AAH	; DATA PATTERN SAME
E616	7505	1253	JNE F17	; NO - CHECK NEXT PRT CD
E618	895408	1254	MOV PRINTER_BASE[SI],DX	; YES - STORE PRT BASE ADDR
E61B	46	1255	INC SI	; INCREMENT TO NEXT WORD
E61C	46	1256	INC SI	
E61D		1257	F17:	
E61D	45	1258	INC BP	; POINT TO NEXT BASE ADDR
E61E	45	1259	INC BP	
E61F	81FDA9F9	1260	CMF BP,OFFSET F4E	; ALL POSSIBLE ADDRS CHECKED?
E623	75E5	1261	JNE F16	; PRT_BASE
E625	BB0000	1262	MOV BX,0	; POINTER TO RS232 TABLE
E628	BAA03	1263	MOV DX,3FAH	; CHECK IF RS232 CD 1 ATTCH?
E62B	EC	1264	IN AL,DX	; READ INTR ID REG
E62C	A8F8	1265	TEST AL,0F8H	
E62E	7506	1266	JNZ F18	
E630	C707F803	1267	MOV RS232_BASE[BX],3F8H	; SETUP RS232 CD #1 ADDR
E634	43	1268	INC BX	
E635	43	1269	INC BX	
E636		1270	F18:	
E636	BAA02	1271	MOV DX,2FAH	; CHECK IF RS232 CD 2 ATTCH
E639	EC	1272	IN AL,DX	; READ INTERRUPT ID REG
E63A	A8F8	1273	TEST AL,0F8H	
E63C	7506	1274	JNZ F19	; BASE_END
E63E	C707F802	1275	MOV RS232_BASE[BX],2F8H	; SETUP RS232 CD #2
E642	43	1276	INC BX	
E643	43	1277	INC BX	
		1278		
		1279	;----- SET UP EQUIP FLAG TO INDICATE NUMBER OF PRINTERS AND RS232 CARDS	
		1280		
E644		1281	F19:	; BASE_END:
E644	8BC6	1282	MOV AX,SI	; SI HAS 2* NUMBER OF RS232
E646	B103	1283	MOV CL,3	; SHIFT COUNT
E648	D2C8	1284	ROR AL,CL	; ROTATE RIGHT 3 POSITIONS

LOC OBJ	LINE	SOURCE	
E64A 0AC3	1285	OR	AL,BL ; OR IN THE PRINTER COUNT
E64C A21100	1286	MOV	BYTE PTR EQUIP_FLAG+1,AL ; STORE AS SECOND BYTE
E64F BA0102	1287	MOV	DX,201H
E652 EC	1288	IN	AL,DX
E653 90	1289	NOP	
E654 90	1290	NOP	
E655 90	1291	NOP	
E656 A80F	1292	TEST	AL,0FH
E658 7505	1293	JNZ	F20 ; NO_GAME_CARD
E65A 800E110010	1294	OR	BYTE PTR EQUIP_FLAG+1,16
E65F	1295	F20:	; NO_GAME_CARD:
	1296		
	1297		;----- ENABLE NMI INTERRUPTS
	1298		
E65F E461	1299	IN	AL,PORT_B ; RESET CHECK ENABLES
E661 0C30	1300	OR	AL,30H
E663 E661	1301	OUT	PORT_B,AL
E665 24CF	1302	AND	AL,0CFH
E667 E661	1303	OUT	PORT_B,AL
E669 B080	1304	MOV	AL,80H ; ENABLE NMI INTERRUPTS
E66B E6A0	1305	OUT	0A0H,AL
E66D	1306	F21:	; LOAD_BOOT_STRAP:
E66D CD19	1307	INT	19H ; GO TO THE BOOT LOADER
	1308		
	1309		;-----
	1310		; THIS SUBROUTINE PERFORMS A READ/WRITE STORAGE TEST ON A BLOCK :
	1311		; OF STORAGE. :
	1312		; ENTRY REQUIREMENTS: :
	1313		; ES = ADDRESS OF STORAGE SEGMENT BEING TESTED :
	1314		; DS = ADDRESS OF STORAGE SEGMENT BEING TESTED :
	1315		; CX = WORD COUNT OF STORAGE BLOCK TO BE TESTED :
	1316		; EXIT PARAMETERS: :
	1317		; ZERO FLAG = 0 IF STORAGE ERROR (DATA COMPARE OR PARITY :
	1318		; CHECK. AL=0 DENOTES A PARITY CHECK. ELSE AL=XOR'ED :
	1319		; BIT PATTERN OF THE EXPECTED DATA PATTERN VS THE ACTUAL :
	1320		; DATA READ. :
	1321		; AX,BX,CX,DX,DI, AND SI ARE ALL DESTROYED. :
	1322		;-----
	1323		
E66F	1324	STGTST_CNT	PROC NEAR
E66F FC	1325	CLD	; SET DIR FLAG TO INCREMENT
E670 2BFF	1326	SUB	DI,DI ; SET DI-OFFSET 0 REL TO ES REG
E672 2BC0	1327	SUB	AX,AX ; SETUP FOR 0->FF PATTERN TEST
E674	1328	C2_1:	
E674 8B05	1329	MOV	[DI],AL ; ON FIRST BYTE
E676 8A05	1330	MOV	AL,[DI]
E678 32C4	1331	XOR	AL,AH ; 0.K.?
E67A 754D	1332	JNZ	C7 ; GO ERROR IF NOT
E67C FEC4	1333	INC	AH
E67E 8AC4	1334	MOV	AL,AH
E680 75F2	1335	JNZ	C2_1 ; LOOP TILL WRAP THROUGH FF
E682 8BD9	1336	MOV	BX,CX ; SAVE WORD COUNT OF BLOCK TO TEST
E684 D1E3	1337	SHL	BX,1 ; CONVERT TO A BYTE COUNT
E686 B8AAAA	1338	MOV	AX,0AAAAH ; GET INITIAL DATA PATTERN TO WRITE
E689 BA55FF	1339	MOV	DX,0FF55H ; SETUP OTHER DATA PATTERNS TO USE
E68C F3	1340	REP	STOSW ; FILL STORAGE LOCATIONS IN BLOCK
E68D AB			
E68E E461	1341	IN	AL,PORT_B
E690 0C30	1342	OR	AL,00110000B ; TOGGLE PARITY CHECK LATCHES
E692 E661	1343	OUT	PORT_B,AL
E694 90	1344	NOP	
E695 24CF	1345	AND	AL,11001111B
E697 E661	1346	OUT	PORT_B,AL
E699	1347	C3:	
E699 4F	1348	DEC	DI ; POINT TO LAST BYTE JUST WRITTEN
E69A FD	1349	STD	; SET DIR FLAG TO GO BACKWARDS
E69B	1350	C4:	
E69B 8BF7	1351	MOV	SI,DI ; INITIALIZE DESTINATION POINTER
E69D 8BCB	1352	MOV	CX,BX ; SETUP BYTE COUNT FOR LOOP
E69F	1353	C5:	; INNER TEST LOOP
E69F AC	1354	LODSB	; READ OLD TEST BYTE FROM STORAGE [SI]E6A0 32
E6A0 32C4	1355	XOR	AL,AH ; DATA READ AS EXPECTED ?
E6A2 7525	1356	JNE	C7 ; NO - GO TO ERROR ROUTINE
E6A4 8AC2	1357	MOV	AL,DL ; GET NEXT DATA PATTERN TO WRITE
E6A6 AA	1358	STOSB	; WRITE INTO LOC JUST READ [DI]+
E6A7 E2F6	1359	LOOP	C5 ; DECREMENT BYTE COUNT AND LOOP CX
	1360		
E6A9 22E4	1361	AND	AH,AH ; ENDING ZERO PATTERN WRITTEN TO ST6 ?
E6AB 7416	1362	JZ	C6X ; YES - RETURN TO CALLER WITH AL=0


```

LOC OBJ          LINE  SOURCE
E710 8EC2        1440      MOV     ES,DX
E712 BB007C      1441      MOV     BX,OFFSET BOOT_LOCN
1442
1443                                ; DRIVE 0, HEAD 0
E715 B90100      1443      MOV     CX,1                ; SECTOR 1, TRACK 0
E718 CD13        1444      INT     13H                ; DISKETTE_IO
E71A             1445      H2:
E71A 59           1446      POP     CX                ; RECOVER RETRY COUNT
E71B 7304        1447      JNC     H4                ; CF SET BY UNSUCCESSFUL READ
E71D E2E5        1448      LOOP   H1                ; DO IT FOR RETRY TIMES
1449
1450      ;----- UNABLE TO IPL FROM THE DISKETTE
1451
E71F             1452      H3:
E71F CD18        1453      INT     18H                ; GO TO RESIDENT BASIC
1454
1455      ;----- IPL WAS SUCCESSFUL
1456
E721             1457      H4:
E721 EA007C0000  1458      JMP     BOOT_LOCN
1459      BOOT_STRAP      ENDP
1460
1461      ;-----INT 14-----
1462      ; RS232_IO
1463      ; THIS ROUTINE PROVIDES BYTE STREAM I/O TO THE COMMUNICATIONS
1464      ; PORT ACCORDING TO THE PARAMETERS:
1465      ; (AH)=0 INITIALIZE THE COMMUNICATIONS PORT
1466      ; (AL) HAS PARAMETERS FOR INITIALIZATION
1467      ;
1468      ;          7      6      5      4      3      2      1      0
1469      ; ----- BAUD RATE -- --PARITY-- STOPBIT --WORD LENGTH--
1470      ; 000 - 110      X0 - NONE      0 - 1      10 - 7 BITS
1471      ; 001 - 150      01 - ODD      1 - 2      11 - 8 BITS
1472      ; 010 - 300      11 - EVEN
1473      ; 011 - 600
1474      ; 100 - 1200
1475      ; 101 - 2400
1476      ; 110 - 4800
1477      ; 111 - 9600
1478      ;
1479      ; ON RETURN, CONDITIONS SET AS IN CALL TO COMMO STATUS (AH=3)
1480      ; (AH)=1 SEND THE CHARACTER IN (AL) OVER THE COMMO LINE
1481      ; (AL) REGISTER IS PRESERVED
1482      ; ON EXIT, BIT 7 OF AH IS SET IF THE ROUTINE WAS UNABLE
1483      ; TO TRANSMIT THE BYTE OF DATA OVER THE LINE.
1484      ; IF BIT 7 OF AH IS NOT SET, THE REMAINDER OF AH
1485      ; IS SET AS IN A STATUS REQUEST, REFLECTING THE
1486      ; CURRENT STATUS OF THE LINE.
1487      ; (AH)=2 RECEIVE A CHARACTER IN (AL) FROM COMMO LINE BEFORE
1488      ; RETURNING TO CALLER
1489      ; ON EXIT, AH HAS THE CURRENT LINE STATUS, AS SET BY THE
1490      ; STATUS ROUTINE, EXCEPT THAT THE ONLY BITS
1491      ; LEFT ON ARE THE ERROR BITS (7,4,3,2,1)
1492      ; IF AH HAS BIT 7 ON (TIME OUT) THE REMAINING
1493      ; BITS ARE NOT PREDICTABLE.
1494      ; THUS, AH IS NON ZERO ONLY WHEN AN ERROR
1495      ; OCCURRED.
1496      ; (AH)=3 RETURN THE COMMO PORT STATUS IN (AX)
1497      ; AH CONTAINS THE LINE STATUS
1498      ; BIT 7 = TIME OUT
1499      ; BIT 6 = TRANS SHIFT REGISTER EMPTY
1500      ; BIT 5 = TRAN HOLDING REGISTER EMPTY
1501      ; BIT 4 = BREAK DETECT
1502      ; BIT 3 = FRAMING ERROR
1503      ; BIT 2 = PARITY ERROR
1504      ; BIT 1 = OVERRUN ERROR
1505      ; BIT 0 = DATA READY
1506      ; AL CONTAINS THE MODEM STATUS
1507      ; BIT 7 = RECEIVED LINE SIGNAL DETECT
1508      ; BIT 6 = RING INDICATOR
1509      ; BIT 5 = DATA SET READY
1510      ; BIT 4 = CLEAR TO SEND
1511      ; BIT 3 = DELTA RECEIVE LINE SIGNAL DETECT
1512      ; BIT 2 = TRAILING EDGE RING DETECTOR
1513      ; BIT 1 = DELTA DATA SET READY
1514      ; BIT 0 = DELTA CLEAR TO SEND
1515      ;
1516      ; (DX) = PARAMETER INDICATING WHICH RS232 CARD (0,1 ALLOWED)

```

```

LOC OBJ          LINE   SOURCE

1517             ;
1518             ; DATA AREA RS232_BASE CONTAINS THE BASE ADDRESS OF THE 8250 ON THE
1519             ; CARD LOCATION 400H CONTAINS UP TO 4 RS232 ADDRESSES POSSIBLE
1520             ; DATA AREA LABEL RS232_TIM_OUT (BYTE) CONTAINS OUTER LOOP COUNT
1521             ; VALUE FOR TIMEOUT (DEFAULT=1)
1522             ; OUTPUT
1523             ; AX MODIFIED ACCORDING TO PARMS OF CALL
1524             ; ALL OTHERS UNCHANGED
1525             ;-----
1526             ASSUME CS:CODE,DS:DATA
E729             1527             ORG 0E729H
E729             1528             A1 LABEL WORD ; TABLE OF INIT VALUES
E729 1704         1529             DW 1047 ; 110 BAUD
E72B 0003         1530             DW 768 ; 150
E72D 8001         1531             DW 384 ; 300
E72F C000         1532             DW 192 ; 600
E731 6000         1533             DW 96 ; 1200
E733 3000         1534             DW 48 ; 2400
E735 1800         1535             DW 24 ; 4800
E737 0C00         1536             DW 12 ; 9600
1537
E739             1538             RS232_IO PROC FAR
1539
1540             ;----- VECTOR TO APPROPRIATE ROUTINE
1541
E739 FB           1542             STI ; INTERRUPTS BACK ON
E73A 1E           1543             PUSH DS ; SAVE SEGMENT
E73B 52           1544             PUSH DX
E73C 56           1545             PUSH SI
E73D 57           1546             PUSH DI
E73E 51           1547             PUSH CX
E73F 53           1548             PUSH BX
E740 8BF2         1549             MOV SI,DX ; RS232 VALUE TO SI
E742 8BFA         1550             MOV DI,DX
E744 D1E6         1551             SHL SI,1 ; WORD OFFSET
E746 E81013       1552             CALL DDS
E7 49 8B14        1553             MOV DX,RS232_BASE[SI] ; GET BASE ADDRESS
E74B 0BD2         1554             OR DX,DX ; TEST FOR 0 BASE ADDRESS
E74D 7413         1555             JZ A3 ; RETURN
E74F 0AE4         1556             OR AH,AH ; TEST FOR (AH)=0
E751 7416         1557             JZ A4 ; COMMUN INIT
E753 FECC         1558             DEC AH ; TEST FOR (AH)=-1
E755 7445         1559             JZ A5 ; SEND AL
E757 FECC         1560             DEC AH ; TEST FOR (AH)=-2
E759 746A         1561             JZ A12 ; RECEIVE INTO AL
E75B             1562             A2:
E75B FECC         1563             DEC AH ; TEST FOR (AH)=-3
E75D 7503         1564             JNZ A3
E75F E98300       1565             JMP A18 ; COMMUNICATION STATUS
E762             1566             A3:
E762 5B           1567             POP BX
E763 59           1568             POP CX
E764 5F           1569             POP DI
E765 5E           1570             POP SI
E766 5A           1571             POP DX
E767 1F           1572             POP DS
E768 CF           1573             IRET ; RETURN TO CALLER, NO ACTION
1574
1575             ;----- INITIALIZE THE COMMUNICATIONS PORT
1576
E769             1577             A4:
E769 8AE0         1578             MOV AH,AL ; SAVE INIT PARMS IN AH
E76B 83C203       1579             ADD DX,3 ; POINT TO 8250 CONTROL REGISTER
E76E B080         1580             MOV AL,80H
E770 EE           1581             OUT DX,AL ; SET DLAB=1
1582
1583             ;----- DETERMINE BAUD RATE DIVISOR
1584
E771 8AD4         1585             MOV DL,AH ; GET PARMS TO DL
E773 B104         1586             MOV CL,4
E775 D2C2         1587             ROL DL,CL
E777 81E20E00    1588             AND DX,0EH ; ISOLATE THEM
E77B BF29E7       1589             MOV DI,OFFSET A1 ; BASE OF TABLE
E77E 03FA         1590             ADD DI,DX ; PUT INTO INDEX REGISTER
E780 8B14         1591             MOV DX,RS232_BASE[SI] ; POINT TO HIGH ORDER OF DIVISOR
E782 42           1592             INC DX
E783 2E8A4501    1593             MOV AL,CS:[DI]*1 ; GET HIGH ORDER OF DIVISOR

```

LOC OBJ	LINE	SOURCE			
E787 EE	1594	OUT	DX,AL		; SET MS OF DIV TO 0
E788 4A	1595	DEC	DX		
E789 2E8A05	1596	MOV	AL,SC:[DI]		; GET LOW ORDER OF DIVISOR
E78C EE	1597	OUT	DX,AL		; SET LOW OF DIVISOR
E78D 83C203	1598	ADD	DX,3		
E790 8AC4	1599	MOV	AL,AH		; GET PARMS BACK
E792 241F	1600	AND	AL,01FH		; STRIP OFF THE BAUD BITS
E794 EE	1601	OUT	DX,AL		; LINE CONTROL TO 8 BITS
E795 4A	1602	DEC	DX		
E796 4A	1603	DEC	DX		
E797 B000	1604	MOV	AL,0		
E799 EE	1605	OUT	DX,AL		; INTERRUPT ENABLES ALL OFF
E79A EB49	1606	JMP	SHORT A18		; COM_STATUS
	1607				
	1608				;----- SEND CHARACTER IN (AL) OVER COMMO LINE
	1609				
E79C	1610	A5:			
E79C 50	1611	PUSH	AX		; SAVE CHAR TO SEND
E79D 83C204	1612	ADD	DX,4		; MODEM CONTROL REGISTER
E7A0 B003	1613	MOV	AL,3		; DTR AND RTS
E7A2 EE	1614	OUT	DX,AL		; DATA TERMINAL READY, REQUEST TO SEND
E7A3 42	1615	INC	DX		; MODEM STATUS REGISTER
E7A4 42	1616	INC	DX		
E7A5 B730	1617	MOV	BH,30H		; DATA SET READY & CLEAR TO SEND
E7A7 E84800	1618	CALL	WAIT_FOR_STATUS		; ARE BOTH TRUE
E7AA 7408	1619	JE	A9		; YES, READY TO TRANSMIT CHAR
E7AC	1620	A7:			
E7AC 59	1621	POP	CX		
E7AD 8AC1	1622	MOV	AL,CL		; RELOAD DATA BYTE
E7AF	1623	A8:			
E7AF 80CC80	1624	OR	AH,80H		; INDICATE TIME OUT
E7B2 EBAE	1625	JMP	A3		; RETURN
E7B4	1626	A9:			; CLEAR_TO_SEND
E7B4 4A	1627	DEC	DX		; LINE STATUS REGISTER
E7B5	1628	A10:			; WAIT_SEND
E7B5 B720	1629	MOV	BH,20H		; IS TRANSMITTER READY
E7B7 E83800	1630	CALL	WAIT_FOR_STATUS		; TEST FOR TRANSMITTER READY
E7BA 75F0	1631	JNZ	A7		; RETURN WITH TIME OUT SET
E7BC	1632	A11:			; OUT_CHAR
E7BC 83EA05	1633	SUB	DX,5		; DATA PORT
E7BF 59	1634	POP	CX		; RECOVER IN CX TEMPORARILY
E7C0 8AC1	1635	MOV	AL,CL		; MOVE CHAR TO AL FOR OUT, STATUS IN AH
E7C2 EE	1636	OUT	DX,AL		; OUTPUT CHARACTER
E7C3 EB9D	1637	JMP	A3		; RETURN
	1638				
	1639				;----- RECEIVE CHARACTER FROM COMMO LINE
	1640				
E7C5	1641	A12:			
E7C5 83C204	1642	ADD	DX,4		; MODEM CONTROL REGISTER
E7C8 B001	1643	MOV	AL,1		; DATA TERMINAL READY
E7CA EE	1644	OUT	DX,AL		
E7CB 42	1645	INC	DX		; MODEM STATUS REGISTER
E7CC 42	1646	INC	DX		
E7CD	1647	A13:			; WAIT_DSR
E7CD B720	1648	MOV	BH,20H		; DATA SET READY
E7CF E82000	1649	CALL	WAIT_FOR_STATUS		; TEST FOR DSR
E7D2 750B	1650	JNZ	A8		; RETURN WITH ERROR
E7D4	1651	A15:			; WAIT_DSR_END
E7D4 4A	1652	DEC	DX		; LINE STATUS REGISTER
E7D5	1653	A16:			; WAIT_RECV
E7D5 B701	1654	MOV	BH,1		; RECEIVE BUFFER FULL
E7D7 E81800	1655	CALL	WAIT_FOR_STATUS		; TEST FOR REC. BUFF. FULL
E7DA 7503	1656	JNZ	A8		; SET TIME OUT ERROR
E7DC	1657	A17:			; GET_CHAR
E7DC 80E41E	1658	AND	AH,00011110B		; TEST FOR ERR CONDITIONS ON RECV CHAR
E7DF 8B14	1659	MOV	DX,RS232_BASE[SI]		; DATA PORT
E7E1 EC	1660	IN	AL,DX		; GET CHARACTER FROM LINE
E7E2 E97DFF	1661	JMP	A3		; RETURN
	1662				
	1663				;----- COMMO PORT STATUS ROUTINE
	1664				
E7E5	1665	A18:			
E7E5 8B14	1666	MOV	DX,RS232_BASE[SI]		
E7E7 83C205	1667	ADD	DX,5		; CONTROL PORT
E7EA EC	1668	IN	AL,DX		; GET LINE CONTROL STATUS
E7EB 8AE0	1669	MOV	AH,AL		; PUT IN AH FOR RETURN
E7ED 42	1670	INC	DX		; POINT TO MODEM STATUS REGISTER

```

LOC OBJ                LINE    SOURCE

E7EE EC                1671          IN      AL,DX          ; GET MODEM CONTROL STATUS
E7EF E970FF           1672          JMP      A3              ; RETURN
1673                  ;-----
1674          ; WAIT FOR STATUS ROUTINE          :
1675          ;                                :
1676          ; ENTRY:                          :
1677          ; BH=STATUS BIT(S) TO LOOK FOR,   :
1678          ; DX=ADDR. OF STATUS REG         :
1679          ; EXIT:                          :
1680          ; ZERO FLAG ON = STATUS FOUND     :
1681          ; ZERO FLAG OFF = TIMEOUT.       :
1682          ; AH=LAST STATUS READ          :
1683          ;-----
E7F2                  1684  WAIT_FOR_STATUS PROC  NEAR
E7F2 8A507C           1685          MOV      BL,RS232_TIM_OUTDI] ; LOAD OUTER LOOP COUNT
E7F5                  1686  WFS0:
E7F5 2BC9             1687          SUB      CX,CX
E7F7                  1688  WFS1:
E7F7 EC             1689          IN      AL,DX          ; GET STATUS
E7F8 8AE0           1690          MOV      AH,AL          ; MOVE TO AH
E7FA 22C7           1691          AND      AL,BH          ; ISOLATE BITS TO TEST
E7FC 3AC7           1692          CMP      AL,BH          ; EXACTLY = TO MASK
E7FE 7408           1693          JE      WFS_END        ; RETURN WITH ZERO FLAG ON
E800 E2F5           1694          LOOP   WFS1          ; TRY AGAIN
E802 FECB           1695          DEC      BL
E804 75EF           1696          JNZ     WFS0
1697
E806 0AFF           1698          OR      BH,BH          ; SET ZERO FLAG OFF
E808                  1699  WFS_END:
E808 C3              1700          RET
1701          WAIT_FOR_STATUS ENDP
1702          RS232_IO      ENDP
1703
E809 4552524F522E20  1704  F3D  DB      'ERROR. (RESUME = F1 KEY)',13,10 ; ERROR PROMPT
      28524553554045
      203D2022463122
      204B455929
E823 0D
E824 0A

1705
1706          ;---- INT 16 -----
1707          ; KEYBOARD I/O
1708          ; THESE ROUTINES PROVIDE KEYBOARD SUPPORT
1709          ; INPUT
1710          ; (AH)=0 READ THE NEXT ASCII CHARACTER STRUCK FROM THE KEYBOARD
1711          ; RETURN THE RESULT IN (AL), SCAN CODE IN (AH)
1712          ; (AH)=1 SET THE Z FLAG TO INDICATE IF AN ASCII CHARACTER IS
1713          ; AVAILABLE TO BE READ.
1714          ; (ZF)=1 -- NO CODE AVAILABLE
1715          ; (ZF)=0 -- CODE IS AVAILABLE
1716          ; IF ZF = 0, THE NEXT CHARACTER IN THE BUFFER TO BE READ
1717          ; IS IN AX, AND THE ENTRY REMAINS IN THE BUFFER
1718          ; (AH)=2 RETURN THE CURRENT SHIFT STATUS IN AL REGISTER
1719          ; THE BIT SETTINGS FOR THIS CODE ARE INDICATED IN THE
1720          ; THE EQUATES FOR KB_FLAG
1721          ; OUTPUT
1722          ; AS NOTED ABOVE, ONLY AX AND FLAGS CHANGED
1723          ; ALL REGISTERS PRESERVED
1724          ;-----
1725          ASSUME  CS:CODE,DS:DATA
E82E                  1726  ORG      0E82EH
E82E                  1727  KEYBOARD_IO  PROC  FAR
E82E FB             1728          STI
E82F 1E             1729          PUSH  DS              ; INTERRUPTS BACK ON
E830 53             1730          PUSH  BX              ; SAVE CURRENT DS
E831 E82512         1731          CALL  DDS              ; SAVE BX TEMPORARILY
E834 0AE4           1732          OR      AH,AH          ; AH=0
E836 740A           1733          JZ      K1              ; ASCII_READ
E838 FECC           1734          DEC      AH              ; AH=1
E83A 741E           1735          JZ      K2              ; ASCII_STATUS
E83C FECC           1736          DEC      AH              ; AH=2
E83E 742B           1737          JZ      K3              ; SHIFT_STATUS
E840 EB2C           1738          JMP      SHORT INT10_END ; EXIT
1739
1740          ;----- READ THE KEY TO FIGURE OUT WHAT TO DO
1741
E842                  1742  K1:
      ; ASCII READ

```

LOC OBJ	LINE	SOURCE	
E842 FB	1743	STI	; INTERRUPTS BACK ON DURING LOOP
E843 90	1744	NOP	; ALLOW AN INTERRUPT TO OCCUR
E844 FA	1745	CLI	; INTERRUPTS BACK OFF
E845 8B1E1A00	1746	MOV BX,BUFFER_HEAD	; GET POINTER TO HEAD OF BUFFER
E849 3B1E1C00	1747	CMP BX,BUFFER_TAIL	; TEST END OF BUFFER
E84D 74F3	1748	JZ K1	; LOOP UNTIL SOMETHING IN BUFFER
E84F 8B07	1749	MOV AX,[BX]	; GET SCAN CODE AND ASCII CODE
E851 E81000	1750	CALL K4	; MOVE POINTER TO NEXT POSITION
E854 891E1A00	1751	MOV BUFFER_HEAD,BX	; STORE VALUE IN VARIABLE
E858 EB14	1752	JMP SHORT INT10_END	; RETURN
	1753		
	1754	;----- ASCII STATUS	
	1755		
E85A	1756	K2:	
E85A FA	1757	CLI	; INTERRUPTS OFF
E85B 8B1E1A00	1758	MOV BX,BUFFER_HEAD	; GET HEAD POINTER
E85F 3B1E1C00	1759	CMP BX,BUFFER_TAIL	; IF EQUAL (Z=1) THEN NOTHING THERE
E863 8B07	1760	MOV AX,[BX]	
E865 FB	1761	STI	; INTERRUPTS BACK ON
E866 5B	1762	POP BX	; RECOVER REGISTER
E867 1F	1763	POP DS	; RECOVER SEGMENT
E868 CA0200	1764	RET 2	; THROW AWAY FLAGS
	1765		
	1766	;----- SHIFT STATUS	
	1767		
E86B	1768	K3:	
E86B A01700	1769	MOV AL,KB_FLAG	; GET THE SHIFT STATUS FLAGS
E86E	1770	INT10_END:	
E86E 5B	1771	POP BX	; RECOVER REGISTER
E86F 1F	1772	POP DS	; RECOVER REGISTERS
E870 CF	1773	IRET	; RETURN TO CALLER
	1774	KEYBOARD_IO	ENDP
	1775		
	1776	;----- INCREMENT A BUFFER POINTER	
	1777		
E871	1778	K4	PROC NEAR
E871 43	1779	INC BX	; MOVE TO NEXT WORD IN LIST
E872 43	1780	INC BX	
E873 3B1E8200	1781	CMP BX,BUFFER_END	; AT END OF BUFFER?
E877 7504	1782	JNE K5	; NO, CONTINUE
E879 8B1E8000	1783	MOV BX,BUFFER_START	; YES, RESET TO BUFFER BEGINNING
E87D	1784	K5:	
E87D C3	1785	RET	
	1786	K4	ENDP
	1787		
	1788	;----- TABLE OF SHIFT KEYS AND MASK VALUES	
	1789		
E87E	1790	K6	LABEL BYTE
E87E 52	1791	DB INS_KEY	; INSERT KEY
E87F 3A	1792	DB CAPS_KEY,NUM_KEY,SCROLL_KEY,ALT_KEY,CTL_KEY	
E880 45			
E881 46			
E882 38			
E883 1D			
E884 2A	1793	DB LEFT_KEY,RIGHT_KEY	
E885 36			
0008	1794	K6L	EQU 8-K6
	1795		
	1796	;----- SHIFT_MASK_TABLE	
	1797		
E886	1798	K7	LABEL BYTE
E886 80	1799	DB INS_SHIFT	; INSERT MODE SHIFT
E887 40	1800	DB CAPS_SHIFT,NUM_SHIFT,SCROLL_SHIFT,ALT_SHIFT,CTL_SHIFT	
E888 20			
E889 10			
E88A 08			
E88B 04			
E88C 02	1801	DB LEFT_SHIFT,RIGHT_SHIFT	
E88D 01			
	1802		
	1803	;----- SCAN CODE TABLES	
	1804		
E88E 1B	1805	K8	DB 27,-1,0,-1,-1,-1,30,-1
E88F FF			
E890 00			
E891 FF			
E892 FF			

LOC OBJ	LINE	SOURCE	
E893 FF			
E894 1E			
E895 FF			
E896 FF	1806	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	1807	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	1808	DB	16,27,29,10,-1,1,19
E8A7 1B			
E8A8 1D			
E8A9 0A			
E8AA FF			
E8AB 01			
E8AC 13			
E8AD 04	1809	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	1810	DB	-1,-1,28,26,24,3,22,2
E8B7 FF			
E8B8 1C			
E8B9 1A			
E8BA 18			
E8BB 03			
E8BC 16			
E8BD 02			
E8BE 0E	1811	DB	14,13,-1,-1,-1,-1,-1,-1
E8BF 0D			
E8C0 FF			
E8C1 FF			
E8C2 FF			
E8C3 FF			
E8C4 FF			
E8C5 FF			
E8C6 20	1812	DB	' ', -1
E8C7 FF			
E8C8	1813	;	----- CTL TABLE SCAN
E8C8 5E	1814	K9	LABEL BYTE
E8C9 5F	1815	DB	94,95,96,97,98,99,100,101
E8CA 60			
E8CB 61			
E8CC 62			
E8CD 63			
E8CE 64			
E8CF 65			
E8D0 66	1816	DB	102,103,-1,-1,119,-1,132,-1
E8D1 67			
E8D2 FF			
E8D3 FF			
E8D4 77			
E8D5 FF			
E8D6 84			
E8D7 FF			
E8D8 72	1817	DB	115,-1,116,-1,117,-1,118,-1
E8D9 FF			
E8DA 74			
E8DB FF			
E8DC 75			
E8DD FF			

LOC OBJ	LINE	SOURCE		
E80E 76				
E80F FF				
E8E0 FF	1818		DB	-1
	1819		;----- LC TABLE	
E8E1	1820	K10	LABEL	BYTE
E8E1 1B	1821		DB	01BH,'1234567890=-','06H,09H
E8E2 31323334353637 363930203D				
E8EE 08				
E8EF 09				
E8F0 71776572747975 696F705B5D	1822		DB	'qwertyuiop[]',0DH,-1,'asdfghjkl;',027H
E8FC 0D				
E8FD FF				
E8FE 6173646667686A 686C3B				
E908 27				
E909 60	1823		DB	60H,-1,5CH,'zxcvbnm,./',-1,'*,-1,' '
E90A FF				
E90B 5C				
E90C 7A786376626E6D 2C2E2F				
E916 FF				
E917 2A				
E918 FF				
E919 20				
E91A FF	1824		DB	-1
	1825		;----- UC TABLE	
E91B	1826	K11	LABEL	BYTE
E91B 1B	1827		DB	27,'!@#&',37,05EH,'&#()_+',08H,0
E91C 21402324				
E920 25				
E921 5E				
E922 262A28295F2B				
E928 08				
E929 00				
E92A 51574552545955 494F507B7D	1828		DB	'QWERTYUIOP()',0DH,-1,'ASDFGHJKL:'''
E936 0D				
E937 FF				
E938 4153444647484A 4B4C3A22				
E943 7E	1829		DB	07EH,-1,' ZXCVBNM{>?',-1,0,-1,' ',-1
E944 FF				
E945 7C5A584356424E 4D3C3E3F				
E950 FF				
E951 00				
E952 FF				
E953 20				
E954 FF				
	1830		;----- UC TABLE SCAN	
E955	1831	K12	LABEL	BYTE
E955 54	1832		DB	84,85,86,87,88,89,90
E956 55				
E957 56				
E958 57				
E959 58				
E95A 59				
E95B 5A				
E95C 5B	1833		DB	91,92,93
E95D 5C				
E95E 5D				
	1834		;----- ALT TABLE SCAN	
E95F	1835	K13	LABEL	BYTE
E95F 68	1836		DB	104,105,106,107,108
E960 69				
E961 6A				
E962 6B				
E963 6C				
E964 6D	1837		DB	109,110,111,112,113
E965 6E				
E966 6F				
E967 70				
E968 71				
	1838		;----- NUM STATE TABLE	
E969	1839	K14	LABEL	BYTE

LOC OBJ	LINE	SOURCE	
E969 3738392D343536 2B3132333302E	1840	DB	'789-656+1230.'
E976	1841	;----- BASE CASE TABLE	
E976 47	1842	K15 LABEL	BYTE
E977 48	1843	DB	71,72,73,-1,75,-1,77
E978 49			
E979 FF			
E97A 4B			
E97B FF			
E97C 4D			
E97D FF	1844	DB	-1,79,80,81,82,83
E97E 4F			
E97F 50			
E980 51			
E981 52			
E982 53			
	1845	;----- KEYBOARD INTERRUPT ROUTINE	
	1846		
	1847		
E987	1848	ORG	0E987H
E987	1849	KB_INT PROC	FAR
E987 FB	1850	STI	; ALLOW FURTHER INTERRUPTS
E988 50	1851	PUSH	AX
E989 53	1852	PUSH	BX
E98A 51	1853	PUSH	CX
E98B 52	1854	PUSH	DX
E98C 56	1855	PUSH	SI
E98D 57	1856	PUSH	DI
E98E 1E	1857	PUSH	DS
E98F 06	1858	PUSH	ES
E990 FC	1859	CLD	; FORWARD DIRECTION
E991 E8C510	1860	CALL	DDS
E994 E460	1861	IN	AL,KB_DATA ; READ IN THE CHARACTER
E996 50	1862	PUSH	AX ; SAVE IT
E997 E461	1863	IN	AL,KB_CTL ; GET THE CONTROL PORT
E999 8AE0	1864	MOV	AH,AL ; SAVE VALUE
E99B 0CA0	1865	OR	AL,80H ; RESET BIT FOR KEYBOARD
E99D E661	1866	OUT	KB_CTL,AL
E99F 86E0	1867	XCHG	AH,AL ; GET BACK ORIGINAL CONTROL
E9A1 E661	1868	OUT	KB_CTL,AL ; KB HAS BEEN RESET
E9A3 58	1869	POP	AX ; RECOVER SCAN CODE
E9A4 8AE0	1870	MOV	AH,AL ; SAVE SCAN CODE IN AH ALSO
	1871		
	1872	;----- TEST FOR OVERRUN SCAN CODE FROM KEYBOARD	
	1873		
E9A6 3CFF	1874	CMP	AL,0FFH ; IS THIS AN OVERRUN CHAR
E9A8 7503	1875	JNZ	K16 ; NO, TEST FOR SHIFT KEY
E9AA E97A02	1876	JMP	K62 ; BUFFER_FULL_BEEP
	1877		
	1878	;----- TEST FOR SHIFT KEYS	
	1879		
E9AD	1880	K16:	; TEST_SHIFT
E9AD 247F	1881	AND	AL,07FH ; TURN OFF THE BREAK BIT
E9AF 0E	1882	PUSH	CS
E9B0 07	1883	POP	ES ; ESTABLISH ADDRESS OF SHIFT TABLE
E9B1 BF7EE8	1884	MOV	DI,OFFSET K6 ; SHIFT KEY TABLE
E9B4 B90600	1885	MOV	CX,K6L ; LENGTH
E9B7 F2	1886	REPNE	SCASB ; LOOK THROUGH THE TABLE FOR A MATCH
E9B8 AE			
E9B9 8AC4	1887	MOV	AL,AH ; RECOVER SCAN CODE
E9BB 7403	1888	JE	K17 ; JUMP IF MATCH FOUND
E9BD E98500	1889	JMP	K25 ; IF NO MATCH, THEN SHIFT NOT FOUND
	1890		
	1891	;----- SHIFIT KEY FOUND	
	1892		
E9C0 81EF7FE8	1893	K17: SUB	DI,OFFSET K6+1 ; ADJUST PTR TO SCAN CODE MTHC
E9C4 2E8AA586E8	1894	MOV	AH,CS:K7IDI1 ; GET MASK INTO AH
E9C9 A880	1895	TEST	AL,80H ; TEST FOR BREAK KEY
E9CB 7551	1896	JNZ	K23 ; BREAK_SHIFT_FOUND
	1897		
	1898	;----- SHIFT MAKE FOUND, DETERMINE SET OR TOGGLE	
	1899		
E9CD 80FC10	1900	CMP	AH,SCROLL_SHIFT ; IF SCROLL SHIFT OR ABOVE, TOGGLE KEY
E9DD 7307	1901	JAE	K18
	1902		
	1903	;----- PLAIN SHIFT KEY, SET SHIFT ON	

LOC OBJ	LINE	SOURCE
	1904	
E9D2 08261700	1905	OR KB_FLAG,AH ; TURN ON SHIFT BIT
E9D6 E98000	1906	JMP K26 ; INTERRUPT_RETURN
	1907	
	1908	!----- TOGGLE SHIFT KEY, TEST FOR 1ST MAKE OR NOT
	1909	
E9D9	1910	K18: ; SHIFT-TOGGLE
E9D9 F066170004	1911	TEST KB_FLAG,CTL_SHIFT ; CHECK CTL SHIFT STATE
E9DE 7565	1912	JNZ K25 ; JUMP IF CTL STATE
E9E0 3C52	1913	CMF AL,INS_KEY ; CHECK FOR INSERT KEY
E9E2 7522	1914	JNZ K22 ; JUMP IF NOT INSERT KEY
E9E4 F066170008	1915	TEST KB_FLAG,ALT_SHIFT ; CHECK FOR ALTERNATE SHIFT
E9E9 755A	1916	JNZ K25 ; JUMP IF ALTERNATE SHIFT
E9EB F066170020	1917	K19: TEST KB_FLAG,NUM_STATE ; CHECK FOR BASE STATE
E9F0 7500	1918	JNZ K21 ; JUMP IF NUM LOCK IS ON
E9F2 F066170003	1919	TEST KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT
E9F7 7400	1920	JZ K22 ; JUMP IF BASE STATE
	1921	
E9F9	1922	K20: ; NUMERIC ZERO, NOT INSERT KEY
E9F9 B83052	1923	MOV AX,5230H ; PUT OUT AN ASCII ZERO
E9FC E9D601	1924	JMP K57 ; BUFFER_FILL
E9FF	1925	K21: ; MIGHT BE NUMERIC
E9FF F066170003	1926	TEST KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT
EA04 74F3	1927	JZ K20 ; JUMP NUMERIC, NOT INSERT
	1928	
EA06	1929	K22: ; SHIFT TOGGLE KEY HIT; PROCESS IT
EA06 84261800	1930	TEST AH,KB_FLAG_1 ; IS KEY ALREADY DEPRESSED
EA0A 7540	1931	JNZ K26 ; JUMP IF KEY ALREADY DEPRESSED
EA0C 08261800	1932	OR KB_FLAG_1,AH ; INDICATE THAT THE KEY IS DEPRESSED
EA10 30261700	1933	XOR KB_FLAG,AH ; TOGGLE THE SHIFT STATE
EA14 3C52	1934	CMF AL,INS_KEY ; TEST FOR 1ST MAKE OF INSERT KEY
EA16 7541	1935	JNE K26 ; JUMP IF NOT INSERT KEY
EA18 B80052	1936	MOV AX,INS_KEY*256 ; SET SCAN CODE INTO AH, 0 INTO AL
EA1B E9B701	1937	JMP K57 ; PUT INTO OUTPUT BUFFER
	1938	
	1939	!----- BREAK SHIFT FOUND
	1940	
EA1E	1941	K23: ; BREAK-SHIFT-FOUND
EA1E 80FC10	1942	CMF AH,SCROLL_SHIFT ; IS THIS A TOGGLE KEY
EA21 731A	1943	JAE K24 ; YES, HANDLE BREAK TOGGLE
EA23 F6D4	1944	NOT AH ; INVERT MASK
EA25 20261700	1945	AND KB_FLAG,AH ; TURN OFF SHIFT BIT
EA29 3CB8	1946	CMF AL,ALT_KEY+80H ; IS THIS ALTERNATE SHIFT RELEASE
EA2B 752C	1947	JNE K26 ; INTERRUPT_RETURN
	1948	
	1949	!----- ALTERNATE SHIFT KEY RELEASED, GET THE VALUE INTO BUFFER
	1950	
EA2D A01900	1951	MOV AL,ALT_INPUT
EA30 B400	1952	MOV AH,0 ; SCAN CODE OF 0
EA32 88261900	1953	MOV ALT_INPUT,AH ; ZERO OUT THE FIELD
EA36 3C00	1954	CMF AL,0 ; WAS THE INPUT=0
EA38 741F	1955	JE K26 ; INTERRUPT_RETURN
EA3A E9A101	1956	JMP K58 ; ARE WE IN HOLD STATE
EA3D	1957	K24: ; BREAK-TOGGLE
EA3D F6D4	1958	NOT AH ; INVERT MASK
EA3F 20261800	1959	AND KB_FLAG_1,AH ; INDICATE NO LONGER DEPRESSED
EA43 EB14	1960	JMP SHORT K26 ; INTERRUPT_RETURN
	1961	
	1962	!----- TEST FOR HOLD STATE
	1963	
EA45	1964	K25: ; NO-SHIFT-FOUND
EA45 3C80	1965	CMF AL,80H ; TEST FOR BREAK KEY
EA47 7310	1966	JAE K26 ; NOTHING FOR BREAK CHARS FROM HERE ON
EA49 F066180008	1967	TEST KB_FLAG_1,HOLD_STATE ; ARE WE IN HOLD STATE
EA4E 7417	1968	JZ K28 ; BRANCH AROUND TEST IF NOT
EA50 3C45	1969	CMF AL,NUM_KEY
EA52 7405	1970	JE K26 ; CAN'T END HOLD ON NUM_LOCK
EA54 80261800F7	1971	AND KB_FLAG_1,NOT_HOLD_STATE ; TURN OFF THE HOLD STATE BIT
EA59	1972	K26: ; INTERRUPT_RETURN
EA59 FA	1973	CLI ; TURN OFF INTERRUPTS
EA5A B020	1974	MOV AL,EOI ; END OF INTERRUPT COMMAND
EA5C E620	1975	OUT 020H,AL ; SEND COMMAND TO INT CONTROL PORT
EA5E	1976	K27: ; INTERRUPT_RETURN-NO-EOI
EA5E 07	1977	POP ES
EA5F 1F	1978	POP DS
EA60 5F	1979	POP DI
EA61 5E	1980	POP SI

LOC OBJ	LINE	SOURCE			
EA62	5A	1981	POP	DX	
EA63	59	1982	POP	CX	
EA64	5B	1983	POP	BX	
EA65	58	1984	POP	AX	; RESTORE STATE
EA66	CF	1985	IRET		; RETURN, INTERRUPTS BACK ON
		1986			; WITH FLAG CHANGE
		1987			
		1988	;-----	NOT IN	HOLD STATE, TEST FOR SPECIAL CHARS
		1989			
EA67		1990	K28:		; NO-HOLD-STATE
EA67	F606170008	1991	TEST	KB_FLAG,ALT_SHIFT	; ARE WE IN ALTERNATE SHIFT
EA6C	7503	1992	JNZ	K29	; JUMP IF ALTERNATE SHIFT
EA6E	E99100	1993	JMP	K38	; JUMP IF NOT ALTERNATE
		1994			
		1995	;-----	TEST FOR RESET KEY SEQUENCE (CTL ALT DEL)	
		1996			
EA71		1997	K29:		; TEST-RESET
EA71	F606170004	1998	TEST	KB_FLAG,CTL_SHIFT	; ARE WE IN CONTROL SHIFT ALSO
EA76	7433	1999	JZ	K31	; NO_RESET
EA78	3C53	2000	CMP	AL,DEL_KEY	; SHIFT STATE IS THERE, TEST KEY
EA7A	752F	2001	JNE	K31	; NO_RESET
		2002			
		2003	;-----	CTL-ALT-DEL HAS BEEN FOUND, DO I/O CLEANUP	
		2004			
EA7C	C70672003412	2005	MOV	RESET_FLAG, 1234H	; SET FLAG FOR RESET FUNCTION
EA82	EA58E000F0	2006	JMP	RESET	; JUMP TO POWER ON DIAGNOSTICS
		2007			
		2008	;-----	ALT-INPUT-TABLE	
EA87		2009	K30	LABEL	BYTE
EA87	52	2010	DB	82,79,80,81,75,76,77	
EA88	4F				
EA89	50				
EA8A	51				
EA8B	4B				
EA8C	4C				
EA8D	4D				
EA8E	47	2011	DB	71,72,73	; 10 NUMBERS ON KEYPAD
EA8F	48				
EA90	49				
		2012	;-----	SUPER-SHIFT-TABLE	
EA91	10	2013	DB	16,17,18,19,20,21,22,23	; A-Z TYPEWRITER CHARS
EA92	11				
EA93	12				
EA94	13				
EA95	14				
EA96	15				
EA97	16				
EA98	17				
EA99	18	2014	DB	24,25,30,31,32,33,34,35	
EA9A	19				
EA9B	1E				
EA9C	1F				
EA9D	20				
EA9E	21				
EA9F	22				
EA00	23				
EA01	24	2015	DB	36,37,38,44,45,46,47,48	
EA02	25				
EA03	26				
EA04	2C				
EA05	2D				
EA06	2E				
EA07	2F				
EA08	30				
EA09	31	2016	DB	49,50	
EA0A	32				
		2017			
		2018	;-----	IN ALTERNATE SHIFT, RESET NOT FOUND	
		2019			
EA0B		2020	K31:		; NO-RESET
EA0B	3C39	2021	CMP	AL,57	; TEST FOR SPACE KEY
EA0D	7505	2022	JNE	K32	; NOT THERE
EA0F	B020	2023	MOV	AL,' '	; SET SPACE CHAR
EA13	E92101	2024	JMP	K57	; BUFFER_FILL
		2025			
		2026	;-----	LOOK FOR KEY PAD ENTRY	
		2027			

LOC OBJ	LINE	SOURCE	
EAB4	2020	K32:	; ALT-KEY-PAD
EAB4 BF07EA	2029	MOV DI,OFFSET K30	; ALT-INPUT-TABLE
EAB7 B90A00	2030	MOV CX,10	; LOOK FOR ENTRY USING KEYPAD
EABA F2	2031	REPNE SCASB	; LOOK FOR MATCH
EABB AE			
EABC 7512	2032	JNE K33	; NO_ALT_KEYPAD
EABE 81EF88EA	2033	SUB DI,OFFSET K30+1	; DI NOW HAS ENTRY VALUE
EAC2 A01900	2034	MOV AL,ALT_INPUT	; GET THE CURRENT BYTE
EACS B40A	2035	MOV AH,10	; MULTIPLY BY 10
EAC7 F6E4	2036	MUL AH	
EAC9 03C7	2037	ADD AX,DI	; ADD IN THE LATEST ENTRY
EACB A21900	2038	MOV ALT_INPUT,AL	; STORE IT AWAY
EACE EB89	2039	JMP K26	; THROW AWAY THAT KEYSTROKE
	2040		
	2041	;-----	LOOK FOR SUPERSHIFT ENTRY
	2042		
EAD0	2043	K33:	; NO-ALT-KEYPAD
EAD0 C06190000	2044	MOV ALT_INPUT,0	; ZERO ANY PREVIOUS ENTRY INTO INPUT
EAD5 B91A00	2045	MOV CX,26	; DI,ES ALREADY POINTING
EAD8 F2	2046	REPNE SCASB	; LOOK FOR MATCH IN ALPHABET
EAD9 AE			
EADA 7505	2047	JNE K34	; NOT FOUND, FUNCTION KEY OR OTHER
EADC B000	2048	MOV AL,0	; ASCII CODE OF ZERO
EADE E9F400	2049	JMP K57	; PUT IT IN THE BUFFER
	2050		
	2051	;-----	LOOK FOR TOP ROW OF ALTERNATE SHIFT
	2052		
EAE1	2053	K34:	; ALT-TOP-ROW
EAE1 3C02	2054	CMP AL,2	; KEY WITH '1' ON IT
EAE3 720C	2055	JB K35	; NOT ONE OF INTERESTING KEYS
EAE5 3C0E	2056	CMP AL,14	; IS IT IN THE REGION
EAE7 7308	2057	JAE K35	; ALT-FUNCTION
EAE9 80C476	2058	ADD AH,118	; CONVERT PSEUDO SCAN CODE TO RANGE
EAEC B000	2059	MOV AL,0	; INDICATE AS SUCH
EAEF E9E400	2060	JMP K57	; BUFFER_FILL
	2061		
	2062	;-----	TRANSLATE ALTERNATE SHIFT PSEUDO SCAN CODES
	2063		
EAF1	2064	K35:	; ALT-FUNCTION
EAF1 3C3B	2065	CMP AL,59	; TEST FOR IN TABLE
EAF3 7303	2066	JAE K37	; ALT-CONTINUE
EAF5	2067	K36:	; CLOSE-RETURN
EAF5 E961FF	2068	JMP K26	; IGNORE THE KEY
EAF8	2069	K37:	; ALT-CONTINUE
EAF8 3C47	2070	CMP AL,71	; IN KEYPAD REGION
EAF8 73F9	2071	JAE K36	; IF SO, IGNORE
EAF8 BB5FE9	2072	MOV BX,OFFSET K13	; ALT SHIFT PSEUDO SCAN TABLE
EAF8 E91B01	2073	JMP K63	; TRANSLATE THAT
	2074		
	2075	;-----	NOT IN ALTERNATE SHIFT
	2076		
EB02	2077	K38:	; NOT-ALT-SHIFT
EB02 F606170004	2078	TEST KB_FLAG,CTL_SHIFT	; ARE WE IN CONTROL SHIFT
EB07 7458	2079	JZ K44	; NOT-CTL-SHIFT
	2080		
	2081	;-----	CONTROL SHIFT, TEST SPECIAL CHARACTERS
	2082	;-----	TEST FOR BREAK AND PAUSE KEYS
	2083		
EB09 3C46	2084	CMP AL,SCROLL_KEY	; TEST FOR BREAK
EB0B 7518	2085	JNE K39	; NO-BREAK
EB0D 8B1E8000	2086	MOV BX,BUFFER_START	; RESET BUFFER TO EMPTY
EB11 891E1A00	2087	MOV BUFFER_HEAD,BX	
EB15 891E1C00	2088	MOV BUFFER_TAIL,BX	
EB19 C606710080	2089	MOV BIOS_BREAK,80H	; TURN ON BIOS_BREAK BIT
EB1E CD1B	2090	INT 1BH	; BREAK INTERRUPT VECTOR
EB20 2BC0	2091	SUB AX,AX	; PUT OUT DUMMY CHARACTER
EB22 E9B000	2092	JMP K57	; BUFFER_FILL
EB25	2093	K39:	; NO-BREAK
EB25 3C45	2094	CMP AL,NUM_KEY	; LOOK FOR PAUSE KEY
EB27 7521	2095	JNE K41	; NO-PAUSE
EB29 800E180008	2096	OR KB_FLAG_1,HOLD_STATE	; TURN ON THE HOLD FLAG
EB2E B020	2097	MOV AL,E01	; END OF INTERRUPT TO CONTROL PORT
EB30 E620	2098	OUT 020H,AL	; ALLOW FURTHER KEYSTROKE INTS
	2099		
	2100	;-----	DURING PAUSE INTERVAL, TURN CRT BACK ON
	2101		
EB32 803E490007	2102	CMP CRT_MODE,7	; IS THIS BLACK AND WHITE CARD

LOC OBJ	LINE	SOURCE	
EB37 7407	2103	JE K40	; YES, NOTHING TO DO
EB39 BAD803	2104	MOV DX,03D8H	; PORT FOR COLOR CARD
EB3C A06500	2105	MOV AL,CRT_MODE_SET	; GET THE VALUE OF THE CURRENT MODE
EB3F EE	2106	OUT DX,AL	; SET THE CRT MODE, SO THAT CRT IS ON
EB40	2107	K40:	; PAUSE-LOOP
EB40 F606180008	2108	TEST KB_FLAG_1,HOLD_STATE	
EB45 75F9	2109	JNZ K40	; LOOP UNTIL FLAG TURNED OFF
EB47 E914FF	2110	JMP K27	; INTERRUPT_RETURN_NO_EOI
EB4A	2111	K41:	; NO-PAUSE
	2112		
	2113	;-----	TEST SPECIAL CASE KEY 55
	2114		
EB4A 3C37	2115	CMR AL,55	
EB4C 7506	2116	JNE K42	; NOT-KEY-55
EB4E B80072	2117	MOV AX,114*256	; START/STOP PRINTING SWITCH
EB51 E98100	2118	JMP K57	; BUFFER_FILL
	2119		
	2120	;-----	SET UP TO TRANSLATE CONTROL SHIFT
	2121		
EB54	2122	K42:	; NOT-KEY-55
EB54 B88EE8	2123	MOV BX,OFFSET K8	; SET UP TO TRANSLATE CTL
EB57 3C3B	2124	CMR AL,59	; IS IT IN TABLE
	2125		; CTL-TABLE-TRANSLATE
EB59 7276	2126	JB K56	; YES, GO TRANSLATE CHAR
EB5B	2127	K43:	; CTL-TABLE-TRANSLATE
EB5B B8C8E8	2128	MOV BX,OFFSET K9	; CTL TABLE SCAN
EB5E E9BC00	2129	JMP K63	; TRANSLATE_SCAN
	2130		
	2131	;-----	NOT IN CONTROL SHIFT
	2132		
EB61	2133	K44:	; NOT-CTL-SHIFT
EB61 3C47	2134	CMR AL,71	; TEST FOR KEYPAD REGION
EB63 732C	2135	JAE K48	; HANDLE KEYPAD REGION
EB65 F606170003	2136	TEST KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT	
EB6A 745A	2137	JZ K54	; TEST FOR SHIFT STATE
	2138		
	2139	;-----	UPPER CASE, HANDLE SPECIAL CASES
	2140		
EB6C 3C0F	2141	CMR AL,15	; BACK TAB KEY
EB6E 7505	2142	JNE K45	; NOT-BACK-TAB
EB70 B8000F	2143	MOV AX,15*256	; SET PSEUDO SCAN CODE
EB73 EB60	2144	JMP SHORT K57	; BUFFER_FILL
EB75	2145	K45:	; NOT-BACK-TAB
EB75 3C37	2146	CMR AL,55	; PRINT SCREEN KEY
EB77 7509	2147	JNE K46	; NOT-PRINT-SCREEN
	2148		
	2149	;-----	ISSUE INTERRUPT TO INDICATE PRINT SCREEN FUNCTION
	2150		
EB79 B020	2151	MOV AL,EOI	; END OF CURRENT INTERRUPT
EB7B E620	2152	OUT 020H,AL	; SO FURTHER THINGS CAN HAPPEN
EB7D CD05	2153	INT 5H	; ISSUE PRINT SCREEN INTERRUPT
EB7F E90CFE	2154	JMP K27	; GO BACK WITHOUT EOI OCCURRING
EB82	2155	K46:	; NOT-PRINT-SCREEN
EB82 3C3B	2156	CMR AL,59	; FUNCTION KEYS
EB84 7206	2157	JB K47	; NOT-UPPER-FUNCTION
EB86 B855E9	2158	MOV BX,OFFSET K12	; UPPER CASE PSEUDO SCAN CODES
EB89 E99100	2159	JMP K63	; TRANSLATE_SCAN
EB8C	2160	K47:	; NOT-UPPER-FUNCTION
EB8C BB1BE9	2161	MOV BX,OFFSET K11	; POINT TO UPPER CASE TABLE
EB8F EB40	2162	JMP SHORT K56	; OK, TRANSLATE THE CHAR
	2163		
	2164	;-----	KEYPAD KEYS, MUST TEST NUM LOCK FOR DETERMINATION
	2165		
EB91	2166	K48:	; KEYPAD-REGION
EB91 F606170020	2167	TEST KB_FLAG,NUM_STATE	; ARE WE IN NUM_LOCK
EB96 7520	2168	JNZ K52	; TEST FOR SURE
EB98 F606170003	2169	TEST KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT	; ARE WE IN SHIFT STATE
EB9D 7520	2170	JNZ K53	; IF SHIFTED, REALLY NUM STATE
	2171		
	2172	;-----	BASE CASE FOR KEYPAD
	2173		
EB9F	2174	K49:	; BASE-CASE
EB9F 3C4A	2175	CMR AL,74	; SPECIAL CASE FOR A COUPLE OF KEYS
EBA1 740B	2176	JE K50	; MINUS
EBA3 3C4E	2177	CMR AL,78	
EBA5 740C	2178	JE K51	
EBA7 2C47	2179	SUB AL,71	; CONVERT ORIGIN

LOC OBJ	LINE	SOURCE		
EBA9 BB76E9	2180	MOV	BX,OFFSET K15	; BASE CASE TABLE
EBAC EB71	2181	JMP	SHORT K64	; CONVERT TO PSEUDO SCAN
EBAE	2182	K50:		
EBAE B82D4A	2183	MOV	AX,74*256+'-'	; MINUS
EBB1 EB22	2184	JMP	SHORT K57	; BUFFER_FILL
EBB3	2185	K51:		
EBB3 B82B4E	2186	MOV	AX,78*256+'+'	; PLUS
EBB6 EB1D	2187	JMP	SHORT K57	; BUFFER_FILL
	2188			
	2189		};----- MIGHT BE NUM LOCK, TEST SHIFT STATUS	
	2190			
EBB8	2191	K52:		; ALMOST-NUM-STATE
EBB8 F606170003	2192	TEST	KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT	
EBDD 75E0	2193	JNZ	K49	; SHIFTED TEMP OUT OF NUM STATE
EBBF	2194	K53:		; REALLY_NUM_STATE
EBBF 2C46	2195	SUB	AL,70	; CONVERT ORIGIN
EBC1 B869E9	2196	MOV	BX,OFFSET K14	; NUM STATE TABLE
EBC4 EB0B	2197	JMP	SHORT K56	; TRANSLATE_CHAR
	2198			
	2199		};----- PLAIN OLD LOWER CASE	
	2200			
EBC6	2201	K54:		; NOT-SHIFT
EBC6 3C3B	2202	CMP	AL,59	; TEST FOR FUNCTION KEYS
EBC8 7204	2203	JB	K55	; NOT-LOWER-FUNCTION
EBCA B000	2204	MOV	AL,0	; SCAN CODE IN AH ALREADY
EBCC EB07	2205	JMP	SHORT K57	; BUFFER_FILL
EBCE	2206	K55:		; NOT-LOWER-FUNCTION
EBCE BBE1E8	2207	MOV	BX,OFFSET K10	; LC TABLE
	2208			
	2209		};----- TRANSLATE THE CHARACTER	
	2210			
EBD1	2211	K56:		; TRANSLATE-CHAR
EBD1 FEC8	2212	DEC	AL	; CONVERT ORIGIN
EBD3 2ED7	2213	XLAT	CS:K11	; CONVERT THE SCAN CODE TO ASCII
	2214			
	2215		};----- PUT CHARACTER INTO BUFFER	
	2216			
EBD5	2217	K57:		; BUFFER-FILL
EBD5 3CFF	2218	CMP	AL,-1	; IS THIS AN IGNORE CHAR
EBD7 741F	2219	JE	K59	; YES, DO NOTHING WITH IT
EBD9 80FCFF	2220	CMP	AH,-1	; LOOK FOR -1 PSEUDO SCAN
EBDC 741A	2221	JE	K59	; NEAR_INTERRUPT_RETURN
	2222			
	2223		};----- HANDLE THE CAPS LOCK PROBLEM	
	2224			
EBDE	2225	K58:		; BUFFER-FILL-NOTEST
EBDE F606170040	2226	TEST	KB_FLAG,CAPS_STATE	; ARE WE IN CAPS LOCK STATE
EBE3 7420	2227	JZ	K61	; SKIP IF NOT
	2228			
	2229		};----- IN CAPS LOCK STATE	
	2230			
EBE5 F606170003	2231	TEST	KB_FLAG,LEFT_SHIFT+RIGHT_SHIFT	; TEST FOR SHIFT STATE
EBEA 740F	2232	JZ	K60	; IF NOT SHIFT, CONVERT LOWER TO UPPER
	2233			
	2234		};----- CONVERT ANY UPPER CASE TO LOWER CASE	
	2235			
EBEC 3C41	2236	CMP	AL,'A'	; FIND OUT IF ALPHABETIC
EBEE 7215	2237	JB	K61	; NOT_CAPS_STATE
EBF0 3C5A	2238	CMP	AL,'Z'	
EBF2 7711	2239	JA	K61	; NOT_CAPS_STATE
EBF4 0420	2240	ADD	AL,'a'-'A'	; CONVERT TO LOWER CASE
EBF6 EB0D	2241	JMP	SHORT K61	; NOT_CAPS_STATE
EBF8	2242	K59:		; NEAR-INTERRUPT-RETURN
EBF8 E95EFE	2243	JMP	K26	; INTERRUPT_RETURN
	2244			
	2245		};----- CONVERT ANY LOWER CASE TO UPPER CASE	
	2246			
EBFB	2247	K60:		; LOWER-TO-UPPER
EBFB 3C61	2248	CMP	AL,'a'	; FIND OUT IF ALPHABETIC
EBFD 7206	2249	JB	K61	; NOT_CAPS_STATE
EBFF 3C7A	2250	CMP	AL,'z'	
EC01 7702	2251	JA	K61	; NOT_CAPS_STATE
EC03 2C20	2252	SUB	AL,'a'-'A'	; CONVERT TO UPPER CASE
EC05	2253	K61:		; NOT-CAPS-STATE
EC05 8B1E1C00	2254	MOV	BX,BUFFER_TAIL	; GET THE END POINTER TO THE BUFFER
EC09 8BF3	2255	MOV	SI,BX	; SAVE THE VALUE
EC0B E863FC	2256	CALL	K4	; ADVANCE THE TAIL

```

LOC OBJ          LINE    SOURCE
EC0E 3B1E1A00    2257      CMP     BX,BUFFER_HEAD      ; HAS THE BUFFER WRAPPED AROUND
EC12 7413        2258      JE      K62                  ; BUFFER_FULL_BEEP
EC14 8904        2259      MOV     [SI],AX              ; STORE THE VALUE
EC16 891E1C00    2260      MOV     BUFFER_TAIL,BX      ; MOVE THE POINTER UP
EC1A E93CFE      2261      JMP     K26                  ; INTERRUPT_RETURN
                2262
                2263      ;----- TRANSLATE SCAN FOR PSEUDO SCAN CODES
                2264
EC1D            2265      K63:                      ; TRANSLATE-SCAN
EC10 2C3B        2266      SUB     AL,59                ; CONVERT ORIGIN TO FUNCTION KEYS
EC1F            2267      K64:                      ; TRANSLATE-SCAN-ORGO
EC1F 2ED7        2268      XLAT   CS:K9                ; CTL TABLE SCAN
EC21 8AE0        2269      MOV     AH,AL               ; PUT VALUE INTO AH
EC23 B000        2270      MOV     AL,0                ; ZERO ASCII CODE
EC25 EBAE        2271      JMP     K57                  ; PUT IT INTO THE BUFFER
                2272
                2273      KB_INT  ENDP
                2274
                2275      ;----- BUFFER IS FULL, SOUND THE BEEPER
                2276
EC27            2277      K62:                      ; BUFFER-FULL-BEEP
EC27 B020        2278      MOV     AL,EOI              ; END OF INTERRUPT COMMAND
EC29 E620        2279      OUT    20H,AL              ; SEND COMMAND TO INT CONTROL PORT
EC2B BB8000      2280      MOV     BX,080H            ; NUMBER OF CYCLES FOR 1/12 SECOND TONE
EC2E E461        2281      IN     AL,KB_CTL           ; GET CONTROL INFORMATION
EC30 50          2282      PUSH   AX                  ; SAVE
EC31            2283      K65:                      ; BEEP-CYCLE
EC31 24FC        2284      AND    AL,0FCH             ; TURN OFF TIMER GATE AND SPEAKER DATA
EC33 E661        2285      OUT    KB_CTL,AL           ; OUTPUT TO CONTROL
EC35 B94800      2286      MOV     CX,48H             ; HALF CYCLE TIME FOR TONE
EC38            2287      K66:                      ;
EC38 E2FE        2288      LOOP   K66                 ; SPEAKER OFF
EC3A 0C02        2289      OR     AL,2                ; TURN ON SPEAKER BIT
EC3C E661        2290      OUT    KB_CTL,AL           ; OUTPUT TO CONTROL
EC3E B94800      2291      MOV     CX,48H             ; SET UP COUNT
EC41            2292      K67:                      ;
EC41 E2FE        2293      LOOP   K67                 ; ANOTHER HALF CYCLE
EC43 4B          2294      DEC    BX                  ; TOTAL TIME COUNT
EC44 75EB        2295      JNZ   K65                  ; DO ANOTHER CYCLE
EC46 58          2296      POP    AX                  ; RECOVER CONTROL
EC47 E661        2297      OUT    KB_CTL,AL           ; OUTPUT THE CONTROL
EC49 E912FE      2298      JMP     K27
                2299
EC4C 20333031    2300      F1    DB    ' 301',13,10    ; KEYBOARD ERROR
EC50 0D          2301
EC51 0A          2302
EC52 363031      2301      F3    DB    '601',13,10    ; DISKETTE ERROR
EC55 0D          2302
EC56 0A          2303
                2304      ;-- INT 13 -----
                2305      ; DISKETTE I/O
                2306      ; THIS INTERFACE PROVIDES ACCESS TO THE 5 1/4 DISKETTE DRIVES
                2307      ; INPUT
                2308      ; (AH)=0  RESET DISKETTE SYSTEM
                2309      ; HARD RESET TO NEC, PREPARE COMMAND, RECAL REQUIRED
                2310      ; ON ALL DRIVES
                2311      ; (AH)=1  READ THE STATUS OF THE SYSTEM INTO (AL)
                2312      ; DISKETTE_STATUS FROM LAST OPERATION IS USED
                2313      ;
                2314      ; REGISTERS FOR READ/WRITE/VERIFY/FORMAT
                2315      ; (DL) - DRIVE NUMBER (0-3 ALLOWED, VALUE CHECKED)
                2316      ; (DH) - HEAD NUMBER (0-1 ALLOWED, NOT VALUE CHECKED)
                2317      ; (CH) - TRACK NUMBER (0-39, NOT VALUE CHECKED)
                2318      ; (CL) - SECTOR NUMBER (1-8, NOT VALUE CHECKED,
                2319      ; NOT USED FOR FORMAT)
                2320      ; (AL) - NUMBER OF SECTORS ( MAX = 8, NOT VALUE CHECKED, NOT USED
                2321      ; FOR FORMAT)
                2322      ; (ES:BX) - ADDRESS OF BUFFER ( NOT REQUIRED FOR VERIFY)
                2323      ;
                2324      ; (AH)=2  READ THE DESIRED SECTORS INTO MEMORY
                2325      ; (AH)=3  WRITE THE DESIRED SECTORS FROM MEMORY
                2326      ; (AH)=4  VERIFY THE DESIRED SECTORS
                2327      ; (AH)=5  FORMAT THE DESIRED TRACK
                2328      ; FOR THE FORMAT OPERATION, THE BUFFER POINTER (ES,BX)
                2329      ; MUST POINT TO THE COLLECTION OF DESIRED ADDRESS FIELDS
                2330      ; FOR THE TRACK. EACH FIELD IS COMPOSED OF 4 BYTES,

```

```

2330 ; (C,H,R,N), WHERE C = TRACK NUMBER, H=HEAD NUMBER, ;
2331 ; R = SECTOR NUMBER, N= NUMBER OF BYTES PER SECTOR ;
2332 ; (00=128, 01=256, 02=512, 03=1024). THERE MUST BE ONE ;
2333 ; ENTRY FOR EVERY SECTOR ON THE TRACK. THIS INFORMATION ;
2334 ; IS USED TO FIND THE REQUESTED SECTOR DURING READ/WRITE ;
2335 ; ACCESS. ;
2336 ; ;
2337 ; DATA VARIABLE -- DISK_POINTER ;
2338 ; DOUBLE WORD POINTER TO THE CURRENT SET OF DISKETTE PARAMETERS ;
2339 ; OUTPUT ;
2340 ; AH = STATUS OF OPERATION ;
2341 ; STATUS BITS ARE DEFINED IN THE EQUATES FOR ;
2342 ; DISKETTE_STATUS VARIABLE IN THE DATA SEGMENT OF THIS ;
2343 ; MODULE. ;
2344 ; CY = 0 SUCCESSFUL OPERATION (AH=0 ON RETURN) ;
2345 ; CY = 1 FAILED OPERATION (AH HAS ERROR REASON) ;
2346 ; FOR READ/WRITE/VERIFY ;
2347 ; DS,BX,DX,CH,CL PRESERVED ;
2348 ; AL = NUMBER OF SECTORS ACTUALLY READ ;
2349 ; ***** AL MAY NOT BE CORRECT IF TIME OUT ERROR OCCURS ;
2350 ; NOTE: IF AN ERROR IS REPORTED BY THE DISKETTE CODE, THE ;
2351 ; APPROPRIATE ACTION IS TO RESET THE DISKETTE, THEN RETRY ;
2352 ; THE OPERATION. ON READ ACCESSES, NO MOTOR START DELAY ;
2353 ; IS TAKEN, SO THAT THREE RETRIES ARE REQUIRED ON READS ;
2354 ; TO ENSURE THAT THE PROBLEM IS NOT DUE TO MOTOR ;
2355 ; START-UP. ;
2356 ;-----
2357 ; ASSUME CS:CODE,DS:DATA,ES:DATA
EC59 2358 ORG 0EC59H
EC59 2359 DISKETTE_IO PROC FAR
EC59 FB 2360 STI ; INTERRUPTS BACK ON
ECA5 53 2361 PUSH BX ; SAVE ADDRESS
EC5B 51 2362 PUSH CX
EC5C 1E 2363 PUSH DS ; SAVE SEGMENT REGISTER VALUE
EC5D 56 2364 PUSH SI ; SAVE ALL REGISTERS DURING OPERATION
EC5E 57 2365 PUSH DI
EC5F 55 2366 PUSH BP
EC60 52 2367 PUSH DX
EC61 8BEC 2368 MOV BP,SP ; SET UP POINTER TO HEAD PARM
EC63 8BF30D 2369 CALL DDS
EC66 E81C00 2370 CALL J1 ; CALL THE REST TO ENSURE DS RESTORED
EC69 BB0400 2371 MOV BX,4 ; GET THE MOTOR WAIT PARAMETER
EC6C E8FD01 2372 CALL GET_PARM
EC6F 8A264000 2373 MOV MOTOR_COUNT,AH ; SET THE TIMER COUNT FOR THE MOTOR
EC73 8A264100 2374 MOV AH,DISKETTE_STATUS ; GET STATUS OF OPERATION
EC77 80FC01 2375 CHP AH,1 ; SET THE CARRY FLAG TO INDICATE
EC7A F5 2376 CHC ; SUCCESS OR FAILURE
EC7B 5A 2377 POP DX ; RESTORE ALL REGISTERS
EC7C 5D 2378 POP BP
EC7D 5F 2379 POP DI
EC7E 5E 2380 POP SI
EC7F 1F 2381 POP DS
EC80 59 2382 POP CX
EC81 5B 2383 POP BX ; RECOVER ADDRESS
EC82 CA0200 2384 RET 2 ; THROW AWAY SAVED FLAGS
2385 DISKETTE_IO ENDP
2386
EC85 2387 J1 PROC NEAR
EC85 8AF0 2388 MOV DH,AL ; SAVE # SECTORS IN DH
EC87 80263F007F 2389 AND MOTOR_STATUS,07FH ; INDICATE A READ OPERATION
EC8C 0AE4 2390 OR AH,AH ; AH=0
EC8E 7427 2391 JZ DISK_RESET
EC90 FECC 2392 DEC AH ; AH=1
EC92 7473 2393 JZ DISK_STATUS
EC94 C606410000 2394 MOV DISKETTE_STATUS,0 ; RESET THE STATUS INDICATOR
EC99 80FA04 2395 CHP DL,4 ; TEST FOR DRIVE IN 0-3 RANGE
EC9C 7313 2396 JAE J3 ; ERROR IF ABOVE
EC9E FECC 2397 DEC AH ; AH=2
ECA0 7469 2398 JZ DISK_READ
ECA2 FECC 2399 DEC AH ; AH=3
ECA4 7503 2400 JNZ J2 ; TEST_DISK_VERF
ECA6 E99500 2401 JMP DISK_WRITE
ECA9 2402 J2: ; TEST_DISK_VERF
ECA9 FECC 2403 DEC AH ; AH=4
ECAB 7467 2404 JZ DISK_VERF
ECAD FECC 2405 DEC AH ; AH=5
ECAF 7467 2406 JZ DISK_FORMAT

```

```

LOC OBJ          LINE    SOURCE

ECB1             2407    J3:                ; BAD_COMMAND
ECB1 C606410001  2408    MOV    DISKETTE_STATUS,BAD_CHD ; ERROR CODE, NO SECTORS TRANSFERRED
ECB6 C3          2409    RET                ; UNDEFINED OPERATION
                2410    J1    ENDP
                2411
                2412    ;----- RESET THE DISKETTE SYSTEM
                2413
ECB7             2414    DISK_RESET    PROC    NEAR
ECB7 BAF203      2415    MOV    DX,03F2H    ; ADAPTER CONTROL PORT
ECBA FA         2416    CLI                ; NO INTERRUPTS
ECBB A03F00     2417    MOV    AL,MOTOR_STATUS ; WHICH MOTOR IS ON
ECBE B104       2418    MOV    CL,4        ; SHIFT COUNT
ECC0 D2E0       2419    SAL    AL,CL       ; MOVE MOTOR VALUE TO HIGH NYBBLE
ECC2 A820       2420    TEST   AL, 20H     ; SELECT CORRESPONDING DRIVE
ECC4 750C       2421    JNZ   J5           ; JUMP IF MOTOR ONE IS ON
ECC6 A840       2422    TEST   AL, 40H     ;
ECC8 7506       2423    JNZ   J4           ; JUMP IF MOTOR TWO IS ON
ECCA A880       2424    TEST   AL, 80H     ;
ECCC 7406       2425    JZ    J6           ; JUMP IF MOTOR ZERO IS ON
ECCE FEC0       2426    INC    AL
ECD0             2427    J4:
ECD0 FEC0       2428    INC    AL
ECD2             2429    J5:
ECD2 FEC0       2430    INC    AL
ECD4             2431    J6:
ECD4 0C08       2432    OR     AL,6        ; TURN ON INTERRUPT ENABLE
ECD6 EE         2433    OUT    DX,AL       ; RESET THE ADAPTER
ECD7 C6063E0000 2434    MOV    SEEK_STATUS,0 ; SET RECAL REQUIRED ON ALL DRIVES
ECD8 C606410000 2435    MOV    DISKETTE_STATUS,0 ; SET OK STATUS FOR DISKETTE
ECE1 0C04       2436    OR     AL,4        ; TURN OFF RESET
ECE3 EE         2437    OUT    DX,AL       ; TURN OFF THE RESET
ECE4 FB         2438    STI                ; REENABLE THE INTERRUPTS
ECE5 E82A02     2439    CALL   CHK_STAT_2  ; DO SENSE INTERRUPT STATUS
                2440    ; FOLLOWING RESET
ECE8 A04200     2441    MOV    AL,NEC_STATUS ; IGNORE ERROR RETURN AND DO DMN TEST
ECEB 3CC0       2442    CHP    AL,0C0H     ; TEST FOR DRIVE READY TRANSITION
ECED 7406       2443    JZ    J7           ; EVERYTHING OK
ECEF 800E410020 2444    OR     DISKETTE_STATUS,BAD_NEC ; SET ERROR CODE
ECF4 C3         2445    RET
                2446
                2447    ;----- SEND SPECIFY COMMAND TO NEC
                2448
ECF5             2449    J7:                ; DRIVE_READY
ECF5 B403       2450    MOV    AH,03H      ; SPECIFY COMMAND
ECF7 E84701     2451    CALL   NEC_OUTPUT  ; OUTPUT THE COMMAND
ECFA BB0100     2452    MOV    BX,1        ; FIRST BYTE PARM IN BLOCK
ECFD E86C01     2453    CALL   GET_PARM    ; TO THE NEC CONTROLLER
ED00 BB0300     2454    MOV    BX,3        ; SECOND BYTE PARM IN BLOCK
ED03 E86601     2455    CALL   GET_PARM    ; TO THE NEC CONTROLLER
ED06             2456    J8:                ; RESET_RET
ED06 C3         2457    RET                ; RETURN TO CALLER
                2458    DISK_RESET    ENDP
                2459
                2460    ;----- DISKETTE STATUS ROUTINE
                2461
ED07             2462    DISK_STATUS    PROC    NEAR
ED07 A04100     2463    MOV    AL,DISKETTE_STATUS
ED0A C3         2464    RET
                2465    DISK_STATUS    ENDP
                2466
                2467    ;----- DISKETTE READ
                2468
ED0B             2469    DISK_READ      PROC    NEAR
ED0B B046       2470    MOV    AL,046H    ; READ COMMAND FOR DMA
ED0D             2471    J9:                ; DISK_READ_CONT
ED0D E6B801     2472    CALL   DMA_SETUP  ; SET UP THE DMA
ED10 B4E6       2473    MOV    AH,0E6H    ; SET UP RD COMMAND FOR NEC CONTROLLER
ED12 EB36       2474    JMP    SHORT RM_OPN ; GO DO THE OPERATION
                2475    DISK_READ      ENDP
                2476
                2477    ;----- DISKETTE VERIFY
                2478
ED14             2479    DISK_VERF      PROC    NEAR
ED14 B042       2480    MOV    AL,042H    ; VERIFY COMMAND FOR DMA
ED16 EB5F       2481    JMP    J9         ; DO AS IF DISK READ
                2482    DISK_VERF      ENDP
                2483

```


LOC OBJ	LINE	SOURCE	
	2484	;	----- DISKETTE FORMAT
	2485		
ED18	2486	DISK_FORMAT	PROC NEAR
ED18 800E3F0080	2487	OR	MOTOR_STATUS,80H ; INDICATE WRITE OPERATION
ED1D B04A	2488	MOV	AL,04AH ; WILL WRITE TO THE DISKETTE
ED1F E8A601	2489	CALL	DMA_SETUP ; SET UP THE DMA
ED22 B44D	2490	MOV	AH,04DH ; ESTABLISH THE FORMAT COMMAND
ED24 EB24	2491	JMP	SHORT_RM_OPN ; DO THE OPERATION
ED26	2492	J10:	; CONTINUATION OF RM_OPN FOR FMT
ED26 BB0700	2493	MOV	BX,7 ; GET THE
ED29 EB4001	2494	CALL	GET_PARM ; BYTES/SECTOR VALUE TO NEC
ED2C BB0900	2495	MOV	BX,9 ; GET THE
ED2F E83A01	2496	CALL	GET_PARM ; SECTORS/TRACK VALUE TO NEC
ED32 BB0F00	2497	MOV	BX,15 ; GET THE
ED35 E83401	2498	CALL	GET_PARM ; GAP LENGTH VALUE TO NEC
ED38 BB1100	2499	MOV	BX,17 ; GET THE FILLER BYTE
ED3B E9AB00	2500	JMP	J16 ; TO THE CONTROLLER
	2501	DISK_FORMAT	ENDP
	2502		
	2503	;	----- DISKETTE WRITE ROUTINE
	2504		
ED3E	2505	DISK_WRITE	PROC NEAR
ED3E 800E3F0080	2506	OR	MOTOR_STATUS,80H ; INDICATE WRITE OPERATION
ED43 B04A	2507	MOV	AL,04AH ; DMA WRITE COMMAND
ED45 E88001	2508	CALL	DMA_SETUP
ED48 B4C5	2509	MOV	AH,0C5H ; NEC COMMAND TO WRITE TO DISKETTE
	2510	DISK_WRITE	ENDP
	2511		
	2512	;	----- ALLOW WRITE ROUTINE TO FALL INTO RM_OPN
	2513		
	2514	;	-----
	2515	;	RM_OPN ;
	2516	;	THIS ROUTINE PERFORMS THE READ/WRITE/VERIFY OPERATION ;
	2517	;	-----
ED4A	2518	RM_OPN	PROC NEAR
ED4A 7308	2519	JNC	J11 ; TEST FOR DMA ERROR
ED4C C06410009	2520	MOV	DISKETTE_STATUS,DMA_BOUNDARY ; SET ERROR
ED51 B000	2521	MOV	AL,0 ; NO SECTORS TRANSFERRED
ED53 C3	2522	RET	; RETURN TO MAIN ROUTINE
ED54	2523	J11:	; DO_RM_OPN
ED54 50	2524	PUSH	AX ; SAVE THE COMMAND
	2525		
	2526	;	----- TURN ON THE MOTOR AND SELECT THE DRIVE
	2527		
ED55 51	2528	PUSH	CX ; SAVE THE T/S PARMS
ED56 0ACA	2529	MOV	CL,DL ; GET DRIVE NUMBER AS SHIFT COUNT
ED58 B001	2530	MOV	AL,1 ; MASK FOR DETERMINING MOTOR BIT
ED5A D2E0	2531	SAL	AL,CL ; SHIFT THE MASK BIT
ED5C FA	2532	CLI	; NO INTERRUPTS WHILE DETERMINING
	2533		; MOTOR STATUS
ED5D C6064000FF	2534	MOV	MOTOR_COUNT,0FFH ; SET LARGE COUNT DURING OPERATION
ED62 84063F00	2535	TEST	AL,MOTOR_STATUS ; TEST THAT MOTOR FOR OPERATING
ED66 7531	2536	JNZ	J14 ; IF RUNNING, SKIP THE WAIT
ED68 80263F00F0	2537	AND	MOTOR_STATUS,0F0H ; TURN OFF ALL MOTOR BITS
ED6D 08063F00	2538	OR	MOTOR_STATUS,AL ; TURN ON THE CURRENT MOTOR
ED71 FB	2539	STI	; INTERRUPTS BACK ON
ED72 B010	2540	MOV	AL,10H ; MASK BIT
ED74 D2E0	2541	SAL	AL,CL ; DEVELOP BIT MASK FOR MOTOR ENABLE
ED76 0AC2	2542	OR	AL,DL ; GET DRIVE SELECT BITS IN
ED78 0C0C	2543	OR	AL,0CH ; NO RESET, ENABLE DMA/INT
ED7A 52	2544	PUSH	DX ; SAVE REG
ED7B BAF203	2545	MOV	DX,03F2H ; CONTROL PORT ADDRESS
ED7E EE	2546	OUT	DX,AL
ED7F 5A	2547	POP	DX ; RECOVER REGISTERS
	2548		
	2549	;	----- WAIT FOR MOTOR IF WRITE OPERATION
	2550		
ED80 F6063F0080	2551	TEST	MOTOR_STATUS,80H ; IS THIS A WRITE
ED85 7412	2552	JZ	J14 ; NO, CONTINUE WITHOUT WAIT
ED87 BB1400	2553	MOV	BX,20 ; GET THE MOTOR WAIT
ED8A E8DF00	2554	CALL	GET_PARM ; PARAMETER
ED8D 0AE4	2555	OR	AH,AH ; TEST FOR NO WAIT
ED8F	2556	J12:	; TEST_WAIT_TIME
ED8F 7408	2557	JZ	J14 ; EXIT WITH TIME EXPIRED
ED91 2BC9	2558	SUB	CX,CX ; SET UP 1/8 SECOND LOOP TIME
ED93	2559	J13:	
ED93 E2FE	2560	LOOP	J13 ; WAIT FOR THE REQUIRED TIME

LOC OBJ	LINE	SOURCE	
ED95 FECC	2561	DEC AH	; DECREMENT TIME VALUE
ED97 EBF6	2562	JMP J12	; ARE WE DONE YET
ED99	2563	J14:	; MOTOR_RUNNING
ED99 FB	2564	STI	; INTERRUPTS BACK ON FOR BYPASS WAIT
ED9A 59	2565	POP CX	
	2566		
	2567	;----- DO THE SEEK OPERATION	
	2568		
ED9B E8DF00	2569	CALL SEEK	; MOVE TO CORRECT TRACK
ED9E 58	2570	POP AX	; RECOVER COMMAND
ED9F 8AFC	2571	MOV BH,AH	; SAVE COMMAND IN BH
EDA1 B600	2572	MOV DH,0	; SET NO SECTORS READ IN CASE OF ERROR
EDA3 724B	2573	JC J17	; IF ERROR, THEN EXIT AFTER MOTOR OFF
EDA5 BEFOED90	2574	MOV SI,OFFSET J17	; DUMMY RETURN ON STACK FOR NEC_OUTPUT
EDA9 56	2575	PUSH SI	; SO THAT IT WILL RETURN TO MOTOR OFF
	2576		; LOCATION
	2577		
	2578	;----- SEND OUT THE PARAMETERS TO THE CONTROLLER	
	2579		
EDAA E89400	2580	CALL NEC_OUTPUT	; OUTPUT THE OPERATION COMMAND
EDAD 8A6601	2581	MOV AH,[BP+1]	; GET THE CURRENT HEAD NUMBER
EDB0 D0E4	2582	SAL AH,1	; MOVE IT TO BIT 2
EDB2 D0E4	2583	SAL AH,1	
EDB4 80E404	2584	AND AH,4	; ISOLATE THAT BIT
EDB7 0AE2	2585	OR AH,DL	; OR IN THE DRIVE NUMBER
EDB9 E88500	2586	CALL NEC_OUTPUT	
	2587		
	2588	;----- TEST FOR FORMAT COMMAND	
	2589		
EDBC 80FF40	2590	CMPL BH,040H	; IS THIS A FORMAT OPERATION
EDBF 7503	2591	JNE J15	; NO. CONTINUE WITH R/W/V
EDC1 E962FF	2592	JMP J10	; IF SO, HANDLE SPECIAL
EDC4	2593	J15:	
EDC4 8AE5	2594	MOV AH,CH	; CYLINDER NUMBER
EDC6 E87800	2595	CALL NEC_OUTPUT	
EDC9 8A6601	2596	MOV AH,[BP+1]	; HEAD NUMBER FROM STACK
EDCC E87200	2597	CALL NEC_OUTPUT	
EDCF 8AE1	2598	MOV AH,CL	; SECTOR NUMBER
EDD1 E86D00	2599	CALL NEC_OUTPUT	
EDD4 B80700	2600	MOV BX,7	; BYTES/SECTOR PARM FROM BLOCK
EDD7 E89200	2601	CALL GET_PARM	; TO THE NEC
EDDA B80900	2602	MOV BX,9	; EOT PARM FROM BLOCK
EDDD E88C00	2603	CALL GET_PARM	; TO THE NEC
EDD8 B80B00	2604	MOV BX,11	; GAP LENGTH PARM FROM BLOCK
EDE3 E88600	2605	CALL GET_PARM	; TO THE NEC
EDE6 B80D00	2606	MOV BX,13	; DTL PARM FROM BLOCK
EDE9	2607	J16:	; RW_OPN_FINISH
EDE9 E88000	2608	CALL GET_PARM	; TO THE NEC
EDEC 5E	2609	POP SI	; CAN NOW DISCARD THAT DUMMY
	2610		; RETURN ADDRESS
	2611		
	2612	;----- LET THE OPERATION HAPPEN	
	2613		
EDED E84301	2614	CALL WAIT_INT	; WAIT FOR THE INTERRUPT
EDF0	2615	J17:	; MOTOR_OFF
EDF0 7245	2616	JC J21	; LOOK FOR ERROR
EDF2 E87401	2617	CALL RESULTS	; GET THE NEC STATUS
EDF5 723F	2618	JC J20	; LOOK FOR ERROR
	2619		
	2620	;----- CHECK THE RESULTS RETURNED BY THE CONTROLLER	
	2621		
EDF7 FC	2622	CLD	; SET THE CORRECT DIRECTION
EDF8 BE4200	2623	MOV SI,OFFSET NEC_STATUS	; POINT TO STATUS FIELD
EDFB AC	2624	LODS NEC_STATUS	; GET ST0
EDFC 24C0	2625	AND AL,0C0H	; TEST FOR NORMAL TERMINATION
EDFE 743B	2626	JZ J22	; OPN_OK
EE00 3C40	2627	CMPL AL,040H	; TEST FOR ABNORMAL TERMINATION
EE02 7529	2628	JNZ J10	; NOT ABNORMAL, BAD NEC
	2629		
	2630	;----- ABNORMAL TERMINATION, FIND OUT WHY	
	2631		
EE04 AC	2632	LODS NEC_STATUS	; GET ST1
EE05 D0E0	2633	SAL AL,1	; TEST FOR EOT FOUND
EE07 B404	2634	MOV AH,RECORD_NOT_FND	
EE09 7224	2635	JC J19	; RW_FAIL
EE0B D0E0	2636	SAL AL,1	
EE0D D0E0	2637	SAL AL,1	; TEST FOR CRC ERROR

LOC OBJ	LINE	SOURCE	
EE0F B410	2638	MOV	AH,BAD_CRC
EE11 721C	2639	JC	J19 ; RM_FAIL
EE13 D0E0	2640	SAL	AL,1 ; TEST FOR DMA OVERRUN
EE15 B408	2641	MOV	AH,BAD_DMA
EE17 7216	2642	JC	J19 ; RM_FAIL
EE19 D0E0	2643	SAL	AL,1
EE1B D0E0	2644	SAL	AL,1 ; TEST FOR RECORD NOT FOUND
EE1D B404	2645	MOV	AH,RECORD_NOT_FND
EE1F 720E	2646	JC	J19 ; RM_FAIL
EE21 D0E0	2647	SAL	AL,1
EE23 B403	2648	MOV	AH,WRITE_PROTECT ; TEST FOR WRITE_PROTECT
EE25 7208	2649	JC	J19 ; RM_FAIL
EE27 D0E0	2650	SAL	AL,1 ; TEST MISSING ADDRESS MARK
EE29 B402	2651	MOV	AH,BAD_ADDR_MARK
EE2B 7202	2652	JC	J19 ; RM_FAIL
	2653		
	2654		;---- NEC MUST HAVE FAILED
	2655		
EE2D	2656	J18:	; RM-NEC-FAIL
EE2D B420	2657	MOV	AH,BAD_NEC
EE2F	2658	J19:	; RM-FAIL
EE2F 08264100	2659	OR	DISKETTE_STATUS,AH
EE33 E87801	2660	CALL	NUM_TRANS ; HOW MANY WERE REALLY TRANSFERRED
EE36	2661	J20:	; RM_ERR
EE36 C3	2662	RET	; RETURN TO CALLER
EE37	2663	J21:	; RM_ERR_RES
EE37 E82F01	2664	CALL	RESULTS ; FLUSH THE RESULTS BUFFER
EE3A C3	2665	RET	
	2666		
	2667		;---- OPERATION WAS SUCCESSFUL
	2668		
EE3B	2669	J22:	; OPN_OK
EE3B E87001	2670	CALL	NUM_TRANS ; HOW MANY GOT MOVED
EE3E 32E4	2671	XOR	AH,AH ; NO ERRORS
EE40 C3	2672	RET	
	2673	RM_OPN	ENDP
	2674		;
	2675		; NEC_OUTPUT
	2676		; THIS ROUTINE SENDS A BYTE TO THE NEC CONTROLLER AFTER TESTING
	2677		; FOR CORRECT DIRECTION AND CONTROLLER READY THIS ROUTINE WILL
	2678		; TIME OUT IF THE BYTE IS NOT ACCEPTED WITHIN A REASONABLE
	2679		; AMOUNT OF TIME, SETTING THE DISKETTE STATUS ON COMPLETION.
	2680		; INPUT
	2681		; (AH) BYTE TO BE OUTPUT
	2682		; OUTPUT
	2683		; CY = 0 SUCCESS
	2684		; CY = 1 FAILURE -- DISKETTE STATUS UPDATED
	2685		; IF A FAILURE HAS OCCURRED, THE RETURN IS MADE ONE LEVEL
	2686		; HIGHER THAN THE CALLER OF NEC_OUTPUT.
	2687		; THIS REMOVES THE REQUIREMENT OF TESTING AFTER EVERY
	2688		; CALL OF NEC_OUTPUT.
	2689		; (AL) DESTROYED
	2690		;
	2691		-----
EE41	2691	NEC_OUTPUT	PROC NEAR
EE41 52	2692	PUSH	DX ; SAVE REGISTERS
EE42 51	2693	PUSH	CX
EE43 BAF403	2694	MOV	DX,03F4H ; STATUS PORT
EE46 33C9	2695	XOR	CX,CX ; COUNT FOR TIME OUT
EE48	2696	J23:	
EE48 EC	2697	IN	AL,DX ; GET STATUS
EE49 A840	2698	TEST	AL,040H ; TEST DIRECTION BIT
EE4B 740C	2699	JZ	J25 ; DIRECTION OK
EE4D E2F9	2700	LOOP	J23
EE4F	2701	J24:	; TIME_ERROR
EE4F 800E410080	2702	OR	DISKETTE_STATUS,TIME_OUT
EE54 59	2703	POP	CX
EE55 5A	2704	POP	DX ; SET ERROR CODE AND RESTORE REGS
EE56 58	2705	POP	AX ; DISCARD THE RETURN ADDRESS
EE57 F9	2706	STC	; INDICATE ERROR TO CALLER
EE58 C3	2707	RET	
EE59	2708	J25:	
EE59 33C9	2709	XOR	CX,CX ; RESET THE COUNT
EE5B	2710	J26:	
EE5B EC	2711	IN	AL,DX ; GET THE STATUS
EE5C A880	2712	TEST	AL,080H ; IS IT READY
EE5E 7504	2713	JNZ	J27 ; YES, GO OUTPUT
EE60 E2F9	2714	LOOP	J26 ; COUNT DOWN AND TRY AGAIN

LOC OBJ

LINE SOURCE

```

EE62 EBEB      2715          JMP     J24          ; ERROR CONDITION
EE64           2716      J27:          ; OUTPUT
EE64 8AC4      2717          MOV     AL,AH        ; GET BYTE TO OUTPUT
EE66 B2F5      2718          MOV     DL,0F5H      ; DATA PORT (3F5)
EE68 EE        2719          OUT     DX,AL        ; OUTPUT THE BYTE
EE69 59        2720          POP     CX           ; RECOVER REGISTERS
EE6A 5A        2721          POP     DX
EE6B C3        2722          RET                    ; CY = 0 FROM TEST INSTRUCTION
2723      NEC_OUTPUT      ENDP
2724      ;-----
2725      ; GET_PARM          ;
2726      ; THIS ROUTINE FETCHES THE INDEXED POINTER FROM THE DISK_BASE ;
2727      ; BLOCK POINTED AT BY THE DATA VARIABLE DISK_POINTER. A BYTE FROM ;
2728      ; THAT TABLE IS THEN MOVED INTO AH, THE INDEX OF THAT BYTE BEING ;
2729      ; THE PARM IN BX ;
2730      ; ENTRY -- ;
2731      ; BX = INDEX OF BYTE TO BE FETCHED * 2 ;
2732      ; IF THE LOW BIT OF BX IS ON, THE BYTE IS IMMEDIATELY OUTPUT ;
2733      ; TO THE NEC CONTROLLER ;
2734      ; EXIT -- ;
2735      ; AH = THAT BYTE FROM BLOCK ;
2736      ;-----
EE6C           2737      GET_PARM      PROC      NEAR
EE6C 1E        2738          PUSH   DS           ; SAVE SEGMENT
EE6D 2BC0      2739          SUB     AX,AX        ; ZERO TO AX
EE6F 8ED8      2740          MOV     DS,AX
2741          ASSUME DS:ABS0
EE71 C5367800  2742          LDS     SI,DISK_POINTER ; POINT TO BLOCK
EE75 D1EB      2743          SHR     BX,1         ; DIVIDE BX BY 2, AND SET FLAG
2744          ; FOR EXIT
EE77 8A20      2745          MOV     AH,{SI+BX}   ; GET THE WORD
EE79 1F        2746          POP     DS           ; RESTORE SEGMENT
2747          ASSUME DS:DATA
EE7A 72C5      2748          JC     NEC_OUTPUT   ; IF FLAG SET, OUTPUT TO CONTROLLER
EE7C C3        2749          RET                    ; RETURN TO CALLER
2750      GET_PARM      ENDP
2751      ;-----
2752      ; SEEK          ;
2753      ; THIS ROUTINE WILL MOVE THE HEAD ON THE NAMED DRIVE TO THE ;
2754      ; NAMED TRACK. IF THE DRIVE HAS NOT BEEN ACCESSED SINCE THE ;
2755      ; DRIVE RESET COMMAND HAS ISSUED, THE DRIVE WILL BE RECALIBRATED. ;
2756      ; INPUT ;
2757      ; (DL) = DRIVE TO SEEK ON ;
2758      ; (CH) = TRACK TO SEEK TO ;
2759      ; OUTPUT ;
2760      ; CY = 0 SUCCESS ;
2761      ; CY = 1 FAILURE -- DISKETTE_STATUS SET ACCORDINGLY ;
2762      ; (AX) DESTROYED ;
2763      ;-----
EE7D           2764      SEEK      PROC      NEAR
EE7D B001      2765          MOV     AL,1         ; ESTABLISH MASK FOR RECAL TEST
EE7F 51        2766          PUSH   CX           ; SAVE INPUT VALUES
EE80 8ACA      2767          MOV     CL,DL        ; GET DRIVE VALUE INTO CL
EE82 D2C0      2768          ROL     AL,CL        ; SHIFT IT BY THE DRIVE VALUE
EE84 59        2769          POP     CX           ; RECOVER TRACK VALUE
EE85 04063E00  2770          TEST   AL,SEEK_STATUS ; TEST FOR RECAL REQUIRED
EE89 7513      2771          JNZ    J28          ; NO_RECAL
EE8B 08063E00  2772          OR     SEEK_STATUS,AL ; TURN ON THE NO RECAL BIT IN FLAG
EE8F B407      2773          MOV     AH,07H      ; RECALIBRATE COMMAND
EE91 E8ADFF    2774          CALL   NEC_OUTPUT
EE94 8AE2      2775          MOV     AH,DL
EE96 E8A8FF    2776          CALL   NEC_OUTPUT   ; OUTPUT THE DRIVE NUMBER
EE99 E87600    2777          CALL   CHK_STAT_2   ; GET THE INTERRUPT AND SENSE INT STATUS
EE9C 7229      2778          JC     J32          ; SEEK_ERROR
2779
2780      ;----- DRIVE IS IN SYNCH WITH CONTROLLER, SEEK TO TRACK
2781
EE9E           2782      J28:
EE9E B40F      2783          MOV     AH,0FH      ; SEEK COMMAND TO NEC
EEA0 E89EFF    2784          CALL   NEC_OUTPUT
EEA3 8AE2      2785          MOV     AH,DL        ; DRIVE NUMBER
EEA5 E899FF    2786          CALL   NEC_OUTPUT
EEA8 8AE5      2787          MOV     AH,CH        ; TRACK NUMBER
EEAA E894FF    2788          CALL   NEC_OUTPUT
EEAD E86200    2789          CALL   CHK_STAT_2   ; GET ENDING INTERRUPT AND
2790          ; SENSE STATUS
2791

```

```

2792 ;----- WAIT FOR HEAD SETTLE
2793
EEB0 9C          2794          PUSHF          ; SAVE STATUS FLAGS
EEB1 BB1200     2795          MOV     BX,10      ; GET HEAD SETTLE PARAMETER
EEB4 E8B5FF     2796          CALL   GET_PARM
EEB7 51         2797          PUSH     CX          ; SAVE REGISTER
EEB8           2798          J29:          ; HEAD_SETTLE
EEB8 B92602     2799          MOV     CX,550      ; 1 MS LOOP
EEB8 0AE4       2800          OR     AH,AH        ; TEST FOR TIME EXPIRED
EEB8 7406       2801          JZ     J31
EEBF           2802          J30:
EEBF E2FE       2803          LOOP   J30          ; DELAY FOR 1 MS
EEC1 FECC       2804          DEC     AH          ; DECREMENT THE COUNT
EEC3 EBF3       2805          JMP     J29          ; DO IT SOME MORE
EEC5           2806          J31:
EEC5 59         2807          POP     CX          ; RECOVER STATE
EEC6 9D         2808          POPF
EEC7           2809          J32:          ; SEEK_ERROR
EEC7 C3         2810          RET              ; RETURN TO CALLER
2811          SEEK   ENDP
2812          ;-----
2813          ; DMA_SETUP          PROC   NEAR          ;
2814          ; THIS ROUTINE SETS UP THE DMA FOR READ/WRITE/VERIFY OPERATIONS. ;
2815          ; INPUT          ;
2816          ; (AL) = MODE BYTE FOR THE DMA ;
2817          ; (ES:BX) - ADDRESS TO READ/WRITE THE DATA ;
2818          ; OUTPUT          ;
2819          ; (AX) DESTROYED ;
2820          ;-----
EEC8           2821          DMA_SETUP   PROC   NEAR
EEC8 51         2822          PUSH     CX          ; SAVE THE REGISTER
EEC9 FA         2823          CLI              ; NO MORE INTERRUPTS
EECA E60C       2824          OUT     DMA+12,AL    ; SET THE FIRST/LAST F/F
EECC 50         2825          PUSH     AX
EECD 58         2826          POP      AX
EECE E60B       2827          OUT     DMA+11,AL    ; OUTPUT THE MODE BYTE
EED0 8CC0       2828          MOV     AX,ES        ; GET THE ES VALUE
EED2 B104       2829          MOV     CL,4         ; SHIFT COUNT
EED4 D3C0       2830          ROL     AX,CL        ; ROTATE LEFT
EED6 8AE8       2831          MOV     CH,AL        ; GET HIGHEST NYBLE OF ES TO CH
EED8 24F0       2832          AND     AL,0FH       ; ZERO THE LOW NYBBLE FROM SEGMENT
EEDA 03C3       2833          ADD     AX,BX        ; TEST FOR CARRY FROM ADDITION
EEDC 7302       2834          JNC     J33
EEDF FECS       2835          INC     CH          ; CARRY MEANS HIGH 4 BITS MUST BE INC
EEE0           2836          J33:
EEE0 50         2837          PUSH     AX          ; SAVE START ADDRESS
EEE1 E604       2838          OUT     DMA+4,AL     ; OUTPUT LOW ADDRESS
EEE3 8AC4       2839          MOV     AL,AH
EEE5 E604       2840          OUT     DMA+4,AL     ; OUTPUT HIGH ADDRESS
EEE7 8AC5       2841          MOV     AL,CH        ; GET HIGH 4 BITS
EEE9 240F       2842          AND     AL,0FH
EEEB E6B1       2843          OUT     081H,AL      ; OUTPUT THE HIGH 4 BITS TO
2844          ; THE PAGE REGISTER
2845
2846          ;----- DETERMINE COUNT
2847
EED 8AE6        2848          MOV     AH,DH        ; NUMBER OF SECTORS
EEEF 24C0       2849          SUB     AL,AL        ; TIMES 256 INTO AX
EEF1 D1E8       2850          SHR     AX,1         ; SECTORS * 128 INTO AX
EEF3 50         2851          PUSH     AX
EEF4 BB0600     2852          MOV     BX,6         ; GET THE BYTES/SECTOR PARM
EEF7 E872FF     2853          CALL   GET_PARM
EEFA 8ACC       2854          MOV     CL,AH        ; USE AS SHIFT COUNT (0=128, 1=256 ETC)
EEFC 58         2855          POP      AX
EEFD D3E0       2856          SHL     AX,CL        ; MULTIPLY BY CORRECT AMOUNT
EEFF 48         2857          DEC     AX           ; -1 FOR DMA VALUE
EF00 50         2858          PUSH     AX          ; SAVE COUNT VALUE
EF01 E605       2859          OUT     DMA+5,AL     ; LOW BYTE OF COUNT
EF03 8AC4       2860          MOV     AL,AH
EF05 E605       2861          OUT     DMA+5,AL     ; HIGH BYTE OF COUNT
EF07 FB         2862          STI              ; INTERRUPTS BACK ON
EF08 59         2863          POP     CX          ; RECOVER COUNT VALUE
EF09 58         2864          POP     AX          ; RECOVER ADDRESS VALUE
EF0A 03C1       2865          ADD     AX,CX        ; ADD, TEST FOR 64K OVERFLOW
EF0C 59         2866          POP     CX          ; RECOVER REGISTER
EF0D B002       2867          MOV     AL,2         ; MODE FOR 8237
EF0F E60A       2868          OUT     DMA+10,AL    ; INITIALIZE THE DISKETTE CHANNEL

```

```

EF11 C3      2869          RET                      ; RETURN TO CALLER,
              2870                      ; CFL SET BY ABOVE IF ERROR
              2871 DMA_SETUP          ENDP
              2872 ;-----;
              2873 ; CHK_STAT_2                      :
              2874 ; THIS ROUTINE HANDLES THE INTERRUPT RECEIVED AFTER A :
              2875 ; RECALIBRATE, SEEK, OR RESET TO THE ADAPTER. :
              2876 ; THE INTERRUPT IS WAITED FOR, THE INTERRUPT STATUS SENSED, :
              2877 ; AND THE RESULT RETURNED TO THE CALLER. :
              2878 ; INPUT                      :
              2879 ; NONE                      :
              2880 ; OUTPUT                      :
              2881 ; CY = 0 SUCCESS              :
              2882 ; CY = 1 FAILURE -- ERROR IS IN DISKETTE_STATUS :
              2883 ; (AX) DESTROYED          :
              2884 ;-----;
EF12         2885 CHK_STAT_2      PROC NEAR
EF12 E81E00  2886 CALL WAIT_INT          ; WAIT FOR THE INTERRUPT
EF15 7214   2887 JC J34                ; IF ERROR, RETURN IT
EF17 B408   2888 MOV AH,08H           ; SENSE INTERRUPT STATUS COMMAND
EF19 E825FF 2889 CALL NEC_OUTPUT
EF1C E84A00 2890 CALL RESULTS         ; READ IN THE RESULTS
EF1F 720A   2891 JC J34                ; CHK2_RETURN
EF21 A04200 2892 MOV AL,NEC_STATUS    ; GET THE FIRST STATUS BYTE
EF24 2460   2893 AND AL,060H         ; ISOLATE THE BITS
EF26 3C60   2894 CMP AL,060H         ; TEST FOR CORRECT VALUE
EF28 7402   2895 JZ J35                ; IF ERROR, GO MARK IT
EF2A F6     2896 CLC                  ; GOOD RETURN
EF2B        2897 J34:                 ;
EF2B C3     2898 RET                  ; RETURN TO CALLER
EF2C        2899 J35:                 ; CHK2_ERROR
EF2C 800E410040 2900 OR DISKETTE_STATUS,BAD_SEEK
EF31 F9     2901 STC                  ; ERROR RETURN CODE
EF32 C3     2902 RET
              2903 CHK_STAT_2      ENDP
              2904 ;-----;
              2905 ; WAIT_INT                      :
              2906 ; THIS ROUTINE WAITS FOR AN INTERRUPT TO OCCUR. A TIME OUT :
              2907 ; ROUTINE TAKES PLACE DURING THE WAIT, SO THAT AN ERROR MAY BE :
              2908 ; RETURNED IF THE DRIVE IS NOT READY. :
              2909 ; INPUT                      :
              2910 ; NONE                      :
              2911 ; OUTPUT                      :
              2912 ; CY = 0 SUCCESS              :
              2913 ; CY = 1 FAILURE -- DISKETTE_STATUS IS SET ACCORDINGLY :
              2914 ; (AX) DESTROYED          :
              2915 ;-----;
EF33         2916 WAIT_INT      PROC NEAR
EF33 FB     2917 STI                  ; TURN ON INTERRUPTS, JUST IN CASE
EF34 53     2918 PUSH BX
EF35 51     2919 PUSH CX              ; SAVE REGISTERS
EF36 B302   2920 MOV BL,2            ; CLEAR THE COUNTERS
EF38 33C9   2921 XOR CX,CX           ; FOR 2 SECOND WAIT
EF3A        2922 J36:                 ;
EF3A F6063E0080 2923 TEST SEEK_STATUS,INT_FLAG ; TEST FOR INTERRUPT OCCURRING
EF3F 750C   2924 JNZ J37             ;
EF41 E2F7   2925 LOOP J36            ; COUNT DOWN WHILE WAITING
EF43 FECB   2926 DEC BL              ; SECOND LEVEL COUNTER
EF45 75F3   2927 JNZ J36             ;
EF47 800E410080 2928 OR DISKETTE_STATUS,TIME_OUT ; NOTHING HAPPENED
EF4C F9     2929 STC                  ; ERROR RETURN
EF4D        2930 J37:                 ;
EF4D 9C     2931 PUSHF               ; SAVE CURRENT CARRY
EF4E 80263E007F 2932 AND SEEK_STATUS,NOT INT_FLAG ; TURN OFF INTERRUPT FLAG
EF53 9D     2933 POPF                ; RECOVER CARRY
EF54 59     2934 POP CX
EF55 5B     2935 POP BX              ; RECOVER REGISTERS
EF56 C3     2936 RET                  ; GOOD RETURN CODE COMES
EF57        2937 ; FROM TEST INST
              2938 WAIT_INT      ENDP
              2939 ;-----;
              2940 ; DISK_INT                      :
              2941 ; THIS ROUTINE HANDLES THE DISKETTE INTERRUPT :
              2942 ; INPUT                      :
              2943 ; NONE                      :
              2944 ; OUTPUT                      :
              2945 ; THE INTERRUPT FLAG IS SET IS SEEK_STATUS :
              2946 ;-----;

```

```

LOC OBJ          LINE  SOURCE

EF57             2947          ORG      0EF57H
EF57             2948  DISK_INT  PROC      FAR
EF57 FB          2949          STI              ; RE ENABLE INTERRUPTS
EF58 1E          2950          PUSH     DS
EF59 50          2951          PUSH     AX
EF5A E8FC0A      2952          CALL    DDS
EF5D 800E3E0080  2953          OR      SEEK_STATUS,INT_FLAG
EF62 B020        2954          MOV     AL,20H          ; END OF INTERRUPT MARKER
EF64 E620        2955          OUT    20H,AL          ; INTERRUPT CONTROL PORT
EF66 58          2956          POP     AX
EF67 1F          2957          POP     DS              ; RECOVER SYSTEM
EF68 CF          2958          IRET             ; RETURN FROM INTERRUPT
                2959  DISK_INT      ENDP
                2960  ;-----
                2961  ; RESULTS
                2962  ; THIS ROUTINE WILL READ ANYTHING THAT THE NEC CONTROLLER HAS
                2963  ; TO SAY FOLLOWING AN INTERRUPT.
                2964  ; INPUT
                2965  ; NONE
                2966  ; OUTPUT
                2967  ; CY = 0  SUCCESSFUL TRANSFER
                2968  ; CY = 1  FAILURE -- TIME OUT IN WAITING FOR STATUS
                2969  ; NEC_STATUS AREA HAS STATUS BYTE LOADED INTO IT
                2970  ; (AH) DESTROYED
                2971  ;-----
EF69             2972  RESULTS PROC  NEAR
EF69 FC          2973          CLD
EF6A BF4200      2974          MOV     DI,OFFSET NEC_STATUS ; POINTER TO DATA AREA
EF6D 51          2975          PUSH   CX              ; SAVE COUNTER
EF6E 52          2976          PUSH   DX
EF6F 53          2977          PUSH   BX
EF70 B307        2978          MOV     BL,7           ; MAX STATUS BYTES
                2979
                2980  ;----- WAIT FOR REQUEST FOR MASTER
                2981
EF72             2982  J30:
EF72 33C9        2983          XOR     CX,CX          ; INPUT_LOOP
EF74 BAF403      2984          MOV     DX,03F4H      ; COUNTER
EF77             2985  J39:
EF77 EC          2986          IN     AL,DX          ; STATUS PORT
EF78 A880        2987          TEST   AL,080H       ; WAIT FOR MASTER
EF7A 750C        2988          JNZ    J40A          ; GET STATUS
EF7C E2F9        2989          LOOP  J39            ; MASTER READY
EF7E 800E410080  2990          OR     DISKETTE_STATUS,TIME_OUT ; TEST_DIR
EF83             2991  J40:
EF83 F9          2992          STC              ; WAIT_MASTER
EF84 5B          2993          POP     BX          ; RESULTS_ERROR
EF85 5A          2994          POP     DX          ; SET ERROR RETURN
EF86 59          2995          POP     CX
EF87 C3          2996          RET
                2997
                2998  ;----- TEST THE DIRECTION BIT
                2999
EF88             3000  J40A:
EF88 EC          3001          IN     AL,DX          ; GET STATUS REG AGAIN
EF89 A840        3002          TEST   AL,040H       ; POINT AT DATA PORT
EF8B 7507        3003          JNZ    J41           ; TEST DIRECTION BIT
EF8D             3004  J41:
EF8D 800E410020  3005          OR     DISKETTE_STATUS,BAD_NEC ; OK TO READ STATUS
EF92 EBEF        3006          JMP    J40           ; NEC_FAIL
                3007
                3008  ;----- READ IN THE STATUS
                3009
EF94             3010  J42:
EF94 42          3011          INC     DX              ; INPUT_STAT
EF95 EC          3012          IN     AL,DX          ; POINT AT DATA PORT
EF96 8805        3013          MOV     EDI,AL        ; GET THE DATA
EF98 47          3014          INC     DI              ; STORE THE BYTE
EF99 B90A00      3015          MOV     CX,10         ; INCREMENT THE POINTER
EF9C E2FE        3016  J43:  LOOP  J43         ; LOOP TO KILL TIME FOR NEC
EF9E 4A          3017          DEC     DX              ; POINT AT STATUS PORT
EF9F EC          3018          IN     AL,DX          ; GET STATUS
EFA0 A810        3019          TEST   AL,010H       ; TEST FOR NEC STILL BUSY
EFA2 7406        3020          JZ     J44            ; RESULTS DONE
EFA4 FECB        3021          DEC     BL              ; DECREMENT THE STATUS COUNTER
EFA6 75CA        3022          JNZ    J38            ; GO BACK FOR MORE

```

```

LOC OBJ          LINE    SOURCE

EFA6 EBE3        3023          JMP     J41          ; CHIP HAS FAILED
                 3024
                 3025 ;----- RESULT OPERATION IS DONE
                 3026
EFAA             3027 J44:
EFAA 5B          3028     POP     BX
EFAA 5A          3029     POP     DX
EFAA 59          3030     POP     CX          ; RECOVER REGISTERS
EFAA C3          3031     RET          ; GOOD RETURN CODE FROM TEST INST
                 3032
                 3033 ; NUM_TRANS          :
                 3034 ;   THIS ROUTINE CALCULATES THE NUMBER OF SECTORS THAT   :
                 3035 ;   WERE ACTUALLY TRANSFERRED TO/FROM THE DISKETTE      :
                 3036 ; INPUT          :
                 3037 ;   (CH) = CYLINDER OF OPERATION          :
                 3038 ;   (CL) = START SECTOR OF OPERATION      :
                 3039 ; OUTPUT          :
                 3040 ;   (AL) = NUMBER ACTUALLY TRANSFERRED    :
                 3041 ;   NO OTHER REGISTERS MODIFIED          :
                 3042 ;-----
EFAE             3043 NUM_TRANS  PROC  NEAR
EFAE A04500      3044     MOV     AL,NEC_STATUS+3    ; GET CYLINDER ENDED UP ON
EFB1 3AC5        3045     CMP     AL,CH            ; SAME AS WE STARTED
EFB3 A04700      3046     MOV     AL,NEC_STATUS+5    ; GET ENDING SECTOR
EFB6 740A        3047     JZ      J45            ; IF ON SAME CYL, THEN NO ADJUST
EFB8 BB0800      3048     MOV     BX,B          ;
EFBB E8AEFE      3049     CALL    GET_PARM        ; GET EOT VALUE
EFBE BAC4        3050     MOV     AL,AH          ; INTO AL
EFC0 FEC0        3051     INC     AL            ; USE EOT+1 FOR CALCULATION
EFC2             3052 J45:
EFC2 2AC1        3053     SUB     AL,CL          ; SUBTRACT START FROM END
EFC4 C3          3054     RET
                 3055 NUM_TRANS  ENDP
                 3056 RESULTS ENDP
                 3057 ;-----
                 3058 ; DISK_BASE          :
                 3059 ;   THIS IS THE SET OF PARAMETERS REQUIRED FOR DISKETTE OPERATION. :
                 3060 ;   THEY ARE POINTED AT BY THE DATA VARIABLE DISK_POINTER. TO :
                 3061 ;   MODIFY THE PARAMETERS, BUILD ANOTHER PARAMETER BLOCK AND POINT :
                 3062 ;   DISK_POINTER TO IT.          :
                 3063 ;-----
EFC7             3064     ORG     0EFC7H
EFC7             3065 DISK_BASE LABEL BYTE
EFC7 CF          3066     DB     11001111B    ; SRT=C, HD UNLOAD=0F - 1ST SPECIFY BYTE
EFC8 02          3067     DB     2            ; HD LOAD=1, MODE=DMA - 2ND SPECIFY BYTE
EFC9 25          3068     DB     MOTOR_WAIT    ; WAIT AFTER OPN TIL MOTOR OFF
EFCA 02          3069     DB     2            ; 512 BYTES/SECTOR
EFCB 08          3070     DB     8            ; EOT ( LAST SECTOR ON TRACK)
EFCC 2A          3071     DB     02AH        ; GAP LENGTH
EFCD FF          3072     DB     0FFH        ; DTL
EFCE 50          3073     DB     050H        ; GAP LENGTH FOR FORMAT
EFCF F6          3074     DB     0F6H        ; FILL BYTE FOR FORMAT
EFD0 19          3075     DB     25            ; HEAD SETTLE TIME (MILLISECONDS)
EFD1 04          3076     DB     4            ; MOTOR START TIME (1/8 SECONDS)
                 3077
                 3078 ;--- INT 17 -----
                 3079 ; PRINTER_IO          :
                 3080 ;   THIS ROUTINE PROVIDES COMMUNICATION WITH THE PRINTER :
                 3081 ; INPUT          :
                 3082 ;   (AH)=0 PRINT THE CHARACTER IN (AL)          :
                 3083 ;   ON RETURN, AH=1 IF CHARACTER COULD NOT BE PRINTED :
                 3084 ;   (TIME OUT). OTHER BITS SET AS ON NORMAL STATUS CALL :
                 3085 ;   (AH)=1 INITIALIZE THE PRINTER PORT          :
                 3086 ;   RETURNS WITH (AH) SET WITH PRINTER STATUS :
                 3087 ;   (AH)=2 READ THE PRINTER STATUS INTO (AH) :
                 3088 ;   7   6   5   4   3   2-1  0 :
                 3089 ;   |   |   |   |   |   |   | TIME OUT :
                 3090 ;   |   |   |   |   |   |   | UNUSED :
                 3091 ;   |   |   |   |   |   |   | _ 1 = I/O ERROR :
                 3092 ;   |   |   |   |   |   |   | _ 1 = SELECTED :
                 3093 ;   |   |   |   |   |   |   | _ 1 = OUT OF PAPER :
                 3094 ;   |   |   |   |   |   |   | _ 1 = ACKNOWLEDGE :
                 3095 ;   |   |   |   |   |   |   | _ 1 = NOT BUSY :
                 3096 ;
                 3097 ;   (DX) = PRINTER TO BE USED (0,1,2) CORRESPONDING TO ACTUAL :
                 3098 ;   VALUES IN PRINTER_BASE AREA :
                 3099 ;

```



```

3100 ; DATA AREA PRINTER_BASE CONTAINS THE BASE ADDRESS OF THE PRINTER ;
3101 ; CARD(S) AVAILABLE (LOCATED AT BEGINNING OF DATA SEGMENT, ;
3102 ; 408H ABSOLUTE, 3 WORDS) ;
3103 ; ;
3104 ; DATA AREA PRINT_TIM_OUT (BYTE) MAY BE CHANGED TO CAUSE DIFFERENT ;
3105 ; TIME-OUT WAITS. DEFAULT=20 ;
3106 ; ;
3107 ; REGISTERS AH IS MODIFIED ;
3108 ; ALL OTHERS UNCHANGED ;
3109 ;-----
3110 ASSUME CS:CODE,DS:DATA
3111 ORG 0EFD2H
3112 PRINTER_IO PROC FAR
3113 STI ; INTERRUPTS BACK ON
3114 PUSH DS ; SAVE SEGMENT
3115 PUSH DX
3116 PUSH SI
3117 PUSH CX
3118 PUSH BX
3119 CALL DDS
3120 MOV SI,DX ; GET PRINTER PARM
3121 MOV BL,PRINT_TIM_OUT[SI] ; LOAD TIME-OUT PARM
3122 SHL SI,1 ; WORD OFFSET INTO TABLE
3123 MOV DX,PRINTER_BASE[SI] ; GET BASE ADDRESS FOR PRINTER CARD
3124 OR DX,DX ; TEST DX FOR ZERO,
3125 ; INDICATING NO PRINTER
3126 JZ B1 ; RETURN
3127 OR AH,AH ; TEST FOR (AH)=0
3128 JZ B2 ; PRINT_AL
3129 DEC AH ; TEST FOR (AH)=1
3130 JZ B0 ; INIT_PRT
3131 DEC AH ; TEST FOR (AH)=2
3132 JZ B5 ; PRINTER STATUS
3133 B1: ; RETURN
3134 POP BX
3135 POP CX
3136 POP SI ; RECOVER REGISTERS
3137 POP DX ; RECOVER REGISTERS
3138 POP DS
3139 IRET
3140
3141 ;----- PRINT THE CHARACTER IN (AL)
3142
3143 B2:
3144 PUSH AX ; SAVE VALUE TO PRINT
3145 OUT DX,AL ; OUTPUT CHAR TO PORT
3146 INC DX ; POINT TO STATUS PORT
3147 B3:
3148 SUB CX,CX ; WAIT_BUSY
3149 B3_1:
3150 IN AL,DX ; GET STATUS
3151 MOV AH,AL ; STATUS TO AH ALSO
3152 TEST AL,80H ; IS THE PRINTER CURRENTLY BUSY
3153 JNZ B4 ; OUT_STROBE
3154 LOOP B3_1 ; TRY AGAIN
3155 DEC BL ; DROP LOOP COUNT
3156 JNZ B3 ; GO TILL TIMEOUT ENDS
3157 OR AH,1 ; SET ERROR FLAG
3158 AND AH,0F9H ; TURN OFF THE OTHER BITS
3159 JHP SHORT B7 ; RETURN WITH ERROR FLAG SET
3160 B4: ; OUT_STROBE
3161 MOV AL,0DH ; SET THE STROBE HIGH
3162 INC DX ; STROBE IS BIT 0 OF PORT C OF 8255
3163 OUT DX,AL
3164 MOV AL,0CH ; SET THE STROBE LOW
3165 OUT DX,AL
3166 POP AX ; RECOVER THE OUTPUT CHAR
3167
3168 ;----- PRINTER STATUS
3169
3170 B5:
3171 PUSH AX ; SAVE AL REG
3172 B6:
3173 MOV DX,PRINTER_BASE[SI]
3174 INC DX
3175 IN AL,DX ; GET PRINTER STATUS
3176 MOV AH,AL

```

LOC OBJ

LINE SOURCE

```

F025 80E4F8 3177 AND AH,0FH ; TURN OFF UNUSED BITS
F028 3178 B7: ; STATUS_SET
F028 5A 3179 POP DX ; RECOVER AL REG
F029 8AC2 3180 MOV AL,DL ; GET CHARACTER INTO AL
F02B 80F446 3181 XOR AH,48H ; FLIP A COUPLE OF BITS
F02E EBC5 3182 JMP B1 ; RETURN FROM ROUTINE
3183
3184 ;----- INITIALIZE THE PRINTER PORT
3185
F030 3186 B8:
F030 50 3187 PUSH AX ; SAVE AL
F031 42 3188 INC DX ; POINT TO OUTPUT PORT
F032 42 3189 INC DX
F033 B008 3190 MOV AL,8 ; SET INIT LINE LOW
F035 EE 3191 OUT DX,AL
F036 B8E803 3192 MOV AX,1000
F039 3193 B9: ; INIT_LOOP
F039 48 3194 DEC AX ; LOOP FOR RESET TO TAKE
F03A 75FD 3195 JNZ B9 ; INIT_LOOP
F03C B00C 3196 MOV AL,0CH ; NO INTERRUPTS, NON AUTO LF,
3197 ; INIT HIGH
F03E EE 3198 OUT DX,AL
F03F EBD0 3199 JMP B6 ; PRT_STATUS_1
3200 PRINTER_IO ENDP
3201
3202
3203 ;--- INT 10 -----
3204 ; VIDEO_IO :
3205 ; THESE ROUTINES PROVIDE THE CRT INTERFACE :
3206 ; THE FOLLOWING FUNCTIONS ARE PROVIDED: :
3207 ; (AH)=0 SET MODE (AL) CONTAINS MODE VALUE :
3208 ; (AL)=0 40X25 BW (POWER ON DEFAULT) :
3209 ; (AL)=1 40X25 COLOR :
3210 ; (AL)=2 80X25 BW :
3211 ; (AL)=3 80X25 COLOR :
3212 ; GRAPHICS MODES :
3213 ; (AL)=4 320X200 COLOR :
3214 ; (AL)=5 320X200 BW :
3215 ; (AL)=6 640X200 BW :
3216 ; CRT MODE=7 80X25 B&W CARD (USED INTERNAL TO VIDEO ONLY) :
3217 ; *** NOTE BW MODES OPERATE SAME AS COLOR MODES, BUT :
3218 ; COLOR BURST IS NOT ENABLED :
3219 ; (AH)=1 SET CURSOR TYPE :
3220 ; (CH) = BITS 4-0 = START LINE FOR CURSOR :
3221 ; ** HARDWARE WILL ALWAYS CAUSE BLIN :
3222 ; ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC :
3223 ; BLINKING OR NO CURSOR AT ALL :
3224 ; (CL) = BITS 4-0 = END LINE FOR CURSOR :
3225 ; (AH)=2 SET CURSOR POSITION :
3226 ; (DH,DL) = ROW,COLUMN (0,0) IS UPPER LEFT :
3227 ; (BH) = PAGE NUMBER (MUST BE 0 FOR GRAPHICS MODES) :
3228 ; (AH)=3 READ CURSOR POSITION :
3229 ; (BH) = PAGE NUMBER (MUST BE 0 FOR GRAPHICS MODES) :
3230 ; ON EXIT (DH,DL) = ROW,COLUMN OF CURRENT CURSOR :
3231 ; (CH,CL) = CURSOR MODE CURRENTLY SET :
3232 ; (AH)=4 READ LIGHT PEN POSITION :
3233 ; ON EXIT: :
3234 ; (AH) = 0 -- LIGHT PEN SWITCH NOT DOWN/NOT TRIGGERED :
3235 ; (AH) = 1 -- VALID LIGHT PEN VALUE IN REGISTERS :
3236 ; (DH,DL) = ROW,COLUMN OF CHARACTER LP POSN :
3237 ; (CH) = RASTER LINE (0-199) :
3238 ; (BX) = PIXEL COLUMN (0-319,639) :
3239 ; (AH)=5 SELECT ACTIVE DISPLAY PAGE (VALID ONLY FOR ALPHA MODES) :
3240 ; (AL)=NEW PAGE VAL (0-7 FOR MODES 0&1, 0-3 FOR MODES 2&3) :
3241 ; (AH)=6 SCROLL ACTIVE PAGE UP :
3242 ; (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT BOTTOM :
3243 ; OF WINDOW :
3244 ; AL = 0 MEANS BLANK ENTIRE WINDOW :
3245 ; (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL :
3246 ; (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL :
3247 ; (BH) = ATTRIBUTE TO BE USED ON BLANK LINE :
3248 ; (AH)=7 SCROLL ACTIVE PAGE DOWN :
3249 ; (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT TOP :
3250 ; OF WINDOW :
3251 ; AL = 0 MEANS BLANK ENTIRE WINDOW :
3252 ; (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL :
3253 ; (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL :

```


LOC OBJ

LINE SOURCE

```

3331          ASSUME CS:CODE,DS:DATA,ES:VIDEO_RAM
3332          ORG     0F045H
F045          F045          M1 LABEL WORD           ; TABLE OF ROUTINES WITHIN VIDEO I/O
F045 FCF0          3334          DW     OFFSET SET_MODE
F047 CDF1          3335          DW     OFFSET SET_CTYPE
F049 EEF1          3336          DW     OFFSET SET_CPOS
F04B 39F2          3337          DW     OFFSET READ_CURSOR
F04D 9CF7          3338          DW     OFFSET READ_LPEN
F04F 17F2          3339          DW     OFFSET ACT_DISP_PAGE
F051 96F2          3340          DW     OFFSET SCROLL_UP
F053 38F3          3341          DW     OFFSET SCROLL_DOWN
F055 74F3          3342          DW     OFFSET READ_AC_CURRENT
F057 B9F3          3343          DW     OFFSET WRITE_AC_CURRENT
F059 ECF3          3344          DW     OFFSET WRITE_C_CURRENT
F05B 4EF2          3345          DW     OFFSET SET_COLOR
F05D 2FF4          3346          DW     OFFSET WRITE_DOT
F05F 1EF4          3347          DW     OFFSET READ_DOT
F061 18F7          3348          DW     OFFSET WRITE_TTY
F063 74F2          3349          DW     OFFSET VIDEO_STATE
          0020          3350          M1 L EQU     $-M1
          3351
F065          F065          ORG     0F065H
F065          F065          VIDEO_IO PROC     NEAR
          F065 FB          3354          STI           ; INTERRUPTS BACK ON
          F066 FC          3355          CLD           ; SET DIRECTION FORWARD
          F067 06          3356          PUSH        ES
          F068 1E          3357          PUSH        DS           ; SAVE SEGMENT REGISTERS
          F069 52          3358          PUSH        DX
          F06A 51          3359          PUSH        CX
          F06B 53          3360          PUSH        BX
          F06C 56          3361          PUSH        SI
          F06D 57          3362          PUSH        DI
          F06E 50          3363          PUSH        AX           ; SAVE AX VALUE
          F06F 8AC4          3364          MOV     AL,AH           ; GET INTO LOW BYTE
          F071 32E4          3365          XOR     AH,AH           ; ZERO TO HIGH BYTE
          F073 D1E0          3366          SAL     AX,1           ; *2 FOR TABLE LOOKUP
          F075 8BF0          3367          MOV     SI,AX           ; PUT INTO SI FOR BRANCH
          F077 3D2000          3368          CMP     AX,M1L         ; TEST FOR WITHIN RANGE
          F07A 7204          3369          JB     M2             ; BRANCH AROUND BRANCH
          F07C 58          3370          POP     AX             ; THROW AWAY THE PARAMETER
          F07D E94501          3371          JMP     VIDEO_RETURN   ; DO NOTHING IF NOT IN RANGE
          F080          3372          M2:
          F080 E8D609          3373          CALL    DDS
          F083 B800B8          3374          MOV     AX,0B800H      ; SEGMENT FOR COLOR CARD
          F086 8B3E1000          3375          MOV     DI,EQUIP_FLAG  ; GET EQUIPMENT SETTING
          F08A 81E73000          3376          AND     DI,30H         ; ISOLATE CRT SWITCHES
          F08E 83FF30          3377          CMP     DI,30H         ; IS SETTING FOR BW CARD?
          F091 7502          3378          JNE     M3
          F093 B4B0          3379          MOV     AH,0B0H       ; SEGMENT FOR BW CARD
          F095          3380          M3:
          F095 8EC0          3381          MOV     ES,AX          ; SET UP TO POINT AT VIDEO RAM AREAS
          F097 58          3382          POP     AX             ; RECOVER VALUE
          F09B 8A264900          3383          MOV     AH,CRT_MODE    ; GET CURRENT MODE INTO AH
          F09C 2EFA445F0          3384          JMP     WORD PRT CS:[SI+OFFSET M1]
          3385          VIDEO_IO ENDP
          3386          ;-----
          3387          ; SET_MODE :
          3388          ; THIS ROUTINE INITIALIZES THE ATTACHMENT TO :
          3389          ; THE SELECTED MODE. THE SCREEN IS BLANKED. :
          3390          ; INPUT :
          3391          ; (AL) = MODE SELECTED (RANGE 0-9) :
          3392          ; OUTPUT :
          3393          ; NONE :
          3394          ;-----
          3395
          3396          ;---- TABLES FOR USE IN SETTING OF MODE
          3397
F0A4          F0A4          ORG     0F0A4H
F0A4          F0A4          VIDEO_PARAMS LABEL BYTE
          3400          ;---- INIT_TABLE
          3401          DB     36H,28H,2DH,0AH,1FH,6,19H ; SET UP FOR 40X25
          F0A5 28
          F0A6 2D
          F0A7 0A
          F0A8 1F
          F0A9 06
          F0AA 19

```

LOC OBJ	LINE	SOURCE			
F0AB 1C	3402	DB	1CH,2,7,6,7		
F0AC 02					
F0AD 07					
F0AE 06					
F0AF 07					
F0B0 00	3403	DB	0,0,0,0		
F0B1 00					
F0B2 00					
F0B3 00					
0010	3404	M4	EQU	\$-VIDEO_PARMS	
	3405				
F0B4 71	3406	DB	71H,50H,5AH,0AH,1FH,6,19H		; SET UP FOR 80X25
F0B5 50					
F0B6 5A					
F0B7 0A					
F0B8 1F					
F0B9 06					
F0BA 19					
F0BB 1C	3407	DB	1CH,2,7,6,7		
F0BC 02					
F0BD 07					
F0BE 06					
F0BF 07					
F0C0 00	3408	DB	0,0,0,0		
F0C1 00					
F0C2 00					
F0C3 00					
	3409				
F0C4 38	3410	DB	38H,28H,2DH,0AH,7FH,6,64H		; SET UP FOR GRAPHICS
F0C5 28					
F0C6 2D					
F0C7 0A					
F0C8 7F					
F0C9 06					
F0CA 64					
F0CB 70	3411	DB	70H,2,1,6,7		
F0CC 02					
F0CD 01					
F0CE 06					
F0CF 07					
F0D0 00	3412	DB	0,0,0,0		
F0D1 00					
F0D2 00					
F0D3 00					
	3413				
F0D4 61	3414	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	3415	DB	19H,2,0DH,0BH,0CH		
F0DC 02					
F0DD 0D					
F0DE 0B					
F0DF 0C					
F0E0 00	3416	DB	0,0,0,0		
F0E1 00					
F0E2 00					
F0E3 00					
	3417				
F0E4	3418	M5	LABEL	WORD	; TABLE OF REGEN LENGTHS
F0E4 000B	3419		DM	2048	; 40X25
F0E6 0010	3420		DM	4096	; 80X25
F0E8 0040	3421		DM	16384	; GRAPHICS
F0EA 0040	3422		DM	16384	
	3423				
	3424				
	3425				
	3426	M6	LABEL	BYTE	
F0EC	3427		DB	40,40,80,80,40,40,80,80	
F0ED 28					
F0EE 50					
F0EF 50					
F0F0 28					
F0F1 28					

LOC OBJ	LINE	SOURCE	
F0F2 50			
F0F3 50			
	3428		
	3429	;----- C_REG_TAB	
	3430		
F0F4	3431	M7 LABEL BYTE	; TABLE OF MODE SETS
F0F4 2C	3432	DB	2CH,28H,2DH,29H,2AH,2EH,1EH,29H
F0F5 28			
F0F6 2D			
F0F7 29			
F0F8 2A			
F0F9 2E			
F0FA 1E			
F0FB 29			
	3433		
F0FC	3434	SET_MODE PROC NEAR	
F0FC BAD403	3435	MOV DX,03D4H	; ADDRESS OF COLOR CARD
F0FF B300	3436	MOV BL,0	; MODE SET FOR COLOR CARD
F101 83FF30	3437	CMP DI,30H	; IS BW CARD INSTALLED
F104 7506	3438	JNE M8	; OK WITH COLOR
F106 8007	3439	MOV AL,7	; INDICATE BW CARD MODE
F108 B2B4	3440	MOV DL,0B4H	; ADDRESS OF BW CARD (384)
F10A FEC3	3441	INC BL	; MODE SET FOR BW CARD
F10C	3442	M8:	
F10C 8AE0	3443	MOV AH,AL	; SAVE MODE IN AH
F10E A24900	3444	MOV CRT_MODE,AL	; SAVE IN GLOBAL VARIABLE
F111 89166300	3445	MOV ADDR_6845,DX	; SAVE ADDRESS OF BASE
F115 1E	3446	PUSH DS	; SAVE POINTER TO DATA SEGMENT
F116 50	3447	PUSH AX	; SAVE MODE
F117 52	3448	PUSH DX	; SAVE OUTPUT PORT VALUE
F118 83C204	3449	ADD DX,4	; POINT TO CONTROL REGISTER
F11B 8AC3	3450	MOV AL,BL	; GET MODE SET FOR CARD
F11D EE	3451	OUT DX,AL	; RESET VIDEO
F11E 5A	3452	POP DX	; BACK TO BASE REGISTER
F11F 2BC0	3453	SUB AX,AX	; SET UP FOR ABS0 SEGMENT
F121 8ED8	3454	MOV DS,AX	; ESTABLISH VECTOR TABLE ADDRESSING
	3455	ASSUME DS:ABS0	
F123 C51E7400	3456	LDS BX,PARM_PTR	; GET POINTER TO VIDEO PARMS
F127 58	3457	POP AX	; RECOVER PARMS
	3458	ASSUME DS:CODE	
F128 B91000	3459	MOV CX,M4	; LENGTH OF EACH ROW OF TABLE
F12B 80FC02	3460	CMP AH,2	; DETERMINE WHICH ONE TO USE
F12E 7210	3461	JC M9	; MODE IS 0 OR 1
F130 03D9	3462	ADD BX,CX	; MOVE TO NEXT ROW OF INIT TABLE
F132 80FC04	3463	CMP AH,4	
F135 7209	3464	JC M9	; MODE IS 2 OR 3
F137 03D9	3465	ADD BX,CX	; MOVE TO GRAPHICS ROW OF INIT_TABLE
F139 80FC07	3466	CMP AH,7	
F13C 7202	3467	JC M9	; MODE IS 4,5, OR 6
F13E 03D9	3468	ADD BX,CX	; MOVE TO BW CARD ROW OF INIT_TABLE
	3469		
	3470	;----- BX POINTS TO CORRECT ROW OF INITIALIZATION TABLE	
	3471		
F140	3472	M9:	; OUT_INIT
F140 50	3473	PUSH AX	; SAVE MODE IN AH
F141 32E4	3474	XOR AH,AH	; AH WILL SERVE AS REGISTER
	3475		; NUMBER DURING LOOP
	3476		
	3477	;----- LOOP THROUGH TABLE, OUTPUTTING REG ADDRESS, THEN VALUE FROM TABLE	
	3478		
F143	3479	M10:	; INIT LOOP
F143 8AC4	3480	MOV AL,AH	; GET 6845 REGISTER NUMBER
F145 EE	3481	OUT DX,AL	
F146 42	3482	INC DX	; POINT TO DATA PORT
F147 FEC4	3483	INC AH	; NEXT REGISTER VALUE
F149 8A07	3484	MOV AL,[BX]	; GET TABLE VALUE
F14B EE	3485	OUT DX,AL	; OUT TO CHIP
F14C 43	3486	INC BX	; NEXT IN TABLE
F14D 4A	3487	DEC DX	; BACK TO POINTER REGISTER
F14E E2F3	3488	LOOP M10	; DO THE WHOLE TABLE
F150 58	3489	POP AX	; GET MODE BACK
F151 1F	3490	POP DS	; RECOVER SEGMENT VALUE
	3491	ASSUME DS:DATA	
	3492		
	3493	;----- FILL REGEN AREA WITH BLANK	
	3494		
F152 33FF	3495	XOR DI,DI	; SET UP POINTER FOR REGEN

```

LOC OBJ          LINE  SOURCE
F154 893E4E00    3496      MOV   CRT_START,DI          ; START ADDRESS SAVED IN GLOBAL
F158 C60620000   3497      MOV   ACTIVE_PAGE,0        ; SET PAGE VALUE
F15D B90020      3498      MOV   CX,8192              ; NUMBER OF WORDS IN COLOR CARD
F160 80FC04      3499      CMP   AH,4                  ; TEST FOR GRAPHICS
F163 720B        3500      JC    M12                   ; NO_GRAPHICS_INIT
F165 80FC07      3501      CMP   AH,7                  ; TEST FOR BM CARD
F168 7404        3502      JE    M11                   ; BM_CARD_INIT
F16A 33C0        3503      XOR   AX,AX                 ; FILL FOR GRAPHICS MODE
F16C EB05        3504      JMP   SHORT M13             ; CLEAR_BUFFER
F16E          3505      M11:                        ; BM_CARD_INIT
F16E B508        3506      MOV   CH,08H               ; BUFFER SIZE ON BM CARD
F170          3507      M12:                        ; NO_GRAPHICS_INIT
F170 B82007      3508      MOV   AX,' '4*256          ; FILL CHAR FOR ALPHA
F173          3509      M13:                        ; CLEAR_BUFFER
F173 F3          3510      REP   STOSM                ; FILL THE REGEN BUFFER WITH BLANKS
F174 AB
3511
3512          ;----- ENABLE VIDEO AND CORRECT PORT SETTING
3513
F175 C70660000706 3514      MOV   CURSOR_MODE,607H     ; SET CURRENT CURSOR MODE
F17B A04900      3515      MOV   AL,CRT_MODE         ; GET THE MODE
F17E 32E4        3516      XOR   AH,AH                ; INTO AX REGISTER
F180 8BF0        3517      MOV   SI,AX                ; TABLE POINTER, INDEXED BY MODE
F182 8B166300    3518      MOV   DX,ADDR_6845        ; PREPARE TO OUTPUT TO
3519          ; VIDEO ENABLE PORT
F186 83C204      3520      ADD   DX,4
F189 2E8A84F4F0 3521      MOV   AL,CS:[SI+OFFSET M7] ; SET VIDEO ENABLE PORT
F18E EE          3522      OUT  DX,AL
F18F A26500      3523      MOV   CRT_MODE_SET,AL     ; SAVE THAT VALUE
3524
3525          ;----- DETERMINE NUMBER OF COLUMNS, BOTH FOR ENTIRE DISPLAY
3526          ;----- AND THE NUMBER TO BE USED FOR TTY INTERFACE
3527
F192 2E8A84ECF0 3528      MOV   AL,CS:[SI+OFFSET M6] ;
F197 32E4        3529      XOR   AH,AH
F199 A34A00      3530      MOV   CRT_COLS,AX         ; NUMBER OF COLUMNS IN THIS SCREEN
3531
3532          ;----- SET CURSOR POSITIONS
3533
F19C 81E60E00    3534      AND   SI,0EH              ; WORD OFFSET INTO CLEAR LENGTH TABLE
F1A0 2E8B0CE4F0 3535      MOV   CX,CS:[SI+OFFSET M5] ; LENGTH TO CLEAR
F1A5 890E4C00    3536      MOV   CRT_LEN,CX          ; SAVE LENGTH OF CRT -- NOT USED FOR BM
F1A9 B90800      3537      MOV   CX,8                 ; CLEAR ALL CURSOR POSITIONS
F1AC BF5000      3538      MOV   DI,OFFSET CURSOR_POSN
F1AF 1E          3539      PUSH DS                    ; ESTABLISH SEGMENT
F1B0 07          3540      POP  ES                    ; ADDRESSING
F1B1 33C0        3541      XOR   AX,AX
F1B3 F3          3542      REP   STOSM                ; FILL WITH ZEROS
F1B4 AB
3543
3544          ;----- SET UP OVERSCAN REGISTER
3545
F1B5 42          3546      INC  DX                    ; SET OVERSCAN PORT TO A DEFAULT
F1B6 B030      3547      MOV   AL,30H              ; VALUE OF 30H FOR ALL MODES
3548          ; EXCEPT 640X200
F1B8 803E490006 3549      CMP   CRT_MODE,6          ; SEE IF THE MODE IS 640X200 BM
F1BD 7502      3550      JNZ  M14                   ; IF IT ISNT 640X200, THEN GOTO REGULAR
F1BF B03F      3551      MOV   AL,3FH              ; IF IT IS 640X200, THEN PUT IN 3FH
F1C1          3552      M14:                        ;
F1C1 EE          3553      OUT  DX,AL                 ; OUTPUT THE CORRECT VALUE TO 3D9 PORT
F1C2 A26600      3554      MOV   CRT_PALETTE,AL     ; SAVE THE VALUE FOR FUTURE USE
3555
3556          ;----- NORMAL RETURN FROM ALL VIDEO RETURNS
3557
F1C5          3558      VIDEO_RETURN:
F1C5 5F          3559      POP  DI
F1C6 5E          3560      POP  SI
F1C7 5B          3561      POP  BX
F1C8          3562      M15:                        ; VIDED_RETURN_C
F1C8 59          3563      POP  CX
F1C9 5A          3564      POP  DX
F1CA 1F          3565      POP  DS
F1CB 07          3566      POP  ES                    ; RECOVER SEGMENTS
F1CC CF          3567      IRET                       ; ALL DONE
3568      SET_MODE      ENDP
3569          ;-----
3570          ; SET_CTYPE

```

Appendix A

```

LOC OBJ          LINE   SOURCE

3571             ;           THIS ROUTINE SETS THE CURSOR VALUE           :
3572             ; INPUT                                           :
3573             ;           (CX) HAS CURSOR VALUE CH-START LINE, CL-STOP LINE :
3574             ; OUTPUT                                           :
3575             ;           NONE                                       :
3576             ;-----
F1CD             SET_CTYPE      PROC      NEAR
F1CD B40A        3577             MOV      AH,10             ; 6845 REGISTER FOR CURSOR SET
F1CF 890E6000    3578             MOV      CURSOR_MODE,CX          ; SAVE IN DATA AREA
F1D3 E80200      3579             CALL     M16              ; OUTPUT CX REG
F1D6 EBED        3581             JMP      VIDEO_RETURN
3582
3583             ;----- THIS ROUTINE OUTPUTS THE CX REGISTER TO THE 6845 REGS NAMED IN AH
3584
F1D8             M16:
F1D8 8B166300    3586             MOV      DX,ADDR_6845          ; ADDRESS REGISTER
F1DC 8AC4        3587             MOV      AL,AH               ; GET VALUE
F1DE EE         3588             OUT     DX,AL               ; REGISTER SET
F1DF 42         3589             INC     DX                   ; DATA REGISTER
F1E0 8AC5        3590             MOV      AL,CH               ; DATA
F1E2 EE         3591             OUT     DX,AL
F1E3 4A         3592             DEC     DX
F1E4 8AC4        3593             MOV      AL,AH
F1E6 FEC0        3594             INC     AL                   ; POINT TO OTHER DATA REGISTER
F1E8 EE         3595             OUT     DX,AL               ; SET FOR SECOND REGISTER
F1E9 42         3596             INC     DX
F1EA 8AC1        3597             MOV      AL,CL               ; SECOND DATA VALUE
F1EC EE         3598             OUT     DX,AL
F1ED C3         3599             RET                          ; ALL DONE
3600             SET_CTYPE      ENDP
3601             ;-----
3602             ; SET_CPOS
3603             ;           THIS ROUTINE SETS THE CURRENT CURSOR       :
3604             ;           POSITION TO THE NEW X-Y VALUES PASSED     :
3605             ; INPUT                                           :
3606             ;           DX - ROW,COLUMN OF NEW CURSOR           :
3607             ;           BH - DISPLAY PAGE OF CURSOR             :
3608             ; OUTPUT                                           :
3609             ;           CURSOR IS SET AT 6845 IF DISPLAY PAGE   :
3610             ;           IS CURRENT DISPLAY                     :
3611             ;-----
F1EE             SET_CPOS      PROC      NEAR
F1EE 8ACF        3613             MOV      CL,BH               ; ESTABLISH LOOP COUNT
F1F0 32E0        3614             XOR     CH,CH               ; WORD OFFSET
F1F2 D1E1        3615             SAL     CX,1                ; USE INDEX REGISTER
F1F4 8BF1        3616             MOV      SI,CX               ; SAVE THE POINTER
F1F6 895450      3617             MOV     [SI+OFFSET_CURSOR_POSN],DX
F1F9 383E6200    3618             CMP     ACTIVE_PAGE,BH
F1FD 7505        3619             JNZ     M17                 ; SET_CPOS_RETURN
F1FF 8BC2        3620             MOV     AX,DX               ; GET ROW/COLUMN TO AX
F201 E80200      3621             CALL    M18                 ; CURSOR_SET
F204           3622             M17:
F204 EBBF        3623             JMP     VIDEO_RETURN
3624             SET_CPOS      ENDP
3625
3626             ;----- SET CURSOR POSITION, AX HAS ROW/COLUMN FOR CURSOR
3627
F206             M18  PROC      NEAR
F206 E87C00      3629             CALL    POSITION             ; DETERMINE LOCATION IN REGEN BUFFER
F209 8BC8        3630             MOV     CX,AX
F20B 030E4E00    3631             ADD     CX,CRT_START        ; ADD IN THE START ADDR FOR THIS PAGE
F20F D1F9        3632             SAR     CX,1                ; DIVIDE BY 2 FOR CHAR ONLY COUNT
F211 B40E        3633             MOV     AH,14               ; REGISTER NUMBER FOR CURSOR
F213 E8C2FF      3634             CALL    M16                 ; OUTPUT THE VALUE TO THE 6845
F216 C3         3635             RET
3636             M18  ENDP
3637             ;-----
3638             ; ACT_DISP_PAGE
3639             ;           THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING THE :
3640             ;           FULL USE OF THE RAM SET ASIDE FOR THE VIDEO ATTACHMENT :
3641             ; INPUT                                           :
3642             ;           AL HAS THE NEW ACTIVE DISPLAY PAGE       :
3643             ; OUTPUT                                           :
3644             ;           THE 6845 IS RESET TO DISPLAY THAT PAGE   :
3645             ;-----
F217             ACT_DISP_PAGE  PROC      NEAR
F217 A26200      3647             MOV     ACTIVE_PAGE,AL      ; SAVE ACTIVE PAGE VALUE

```


LOC OBJ	LINE	SOURCE	
F21A 800E4C00	3648	MOV	CX,CRT_LEN ; GET SAVED LENGTH OF REGEN BUFFER
F21E 98	3649	CBW	; CONVERT AL TO WORD
F21F 50	3650	PUSH	AX ; SAVE PAGE VALUE
F220 F7E1	3651	MUL	CX ; DISPLAY PAGE TIMES REGEN LENGTH
F222 A34E00	3652	MOV	CRT_START,AX ; SAVE START ADDRESS FOR
	3653		; LATER REQUIREMENTS
F225 8BC8	3654	MOV	CX,AX ; START ADDRESS TO CX
F227 D1F9	3655	SAR	CX,1 ; DIVIDE BY 2 FOR 6845 HANDLING
F229 B40C	3656	MOV	AH,12 ; 6845 REGISTER FOR START ADDRESS
F22B E8AAFF	3657	CALL	M16
F22E 5B	3658	POP	BX ; RECOVER PAGE VALUE
F22F D1E3	3659	SAL	BX,1 ; *2 FOR WORD OFFSET
F231 8B4750	3660	MOV	AX,(BX + OFFSET CURSOR_POSN) ; GET CURSOR FOR THIS PAGE
F234 E8CFFF	3661	CALL	M18 ; SET THE CURSOR POSITION
F237 EB8C	3662	JMP	SHORT VIDEO_RETURN
	3663	ACT_DISP_PAGE	ENDP
	3664		-----
	3665	READ_CURSOR	;
	3666	;	THIS ROUTINE READS THE CURRENT CURSOR VALUE FROM THE ;
	3667	;	6845, FORMATS IT, AND SENDS IT BACK TO THE CALLER ;
	3668	;	INPUT ;
	3669	;	BH - PAGE OF CURSOR ;
	3670	;	OUTPUT ;
	3671	;	DX - ROW, COLUMN OF THE CURRENT CURSOR POSITION ;
	3672	;	CX - CURRENT CURSOR MODE ;
	3673		-----
F239	3674	READ_CURSOR	PROC NEAR
F239 8ADF	3675	MOV	BL,BH
F23B 32FF	3676	XOR	BH,BH
F23D D1E3	3677	SAL	BX,1 ; WORD OFFSET
F23F 8B5750	3678	MOV	DX,(BX+OFFSET CURSOR_POSN)
F242 8B0E6000	3679	MOV	CX,CURSOR_MODE
F246 5F	3680	POP	DI
F247 5E	3681	POP	SI
F248 5B	3682	POP	BX
F249 58	3683	POP	AX ; DISCARD SAVED CX AND DX
F24A 58	3684	POP	AX
F24B 1F	3685	POP	DS
F24C 0F	3686	POP	ES
F24D CF	3687	IRET	
	3688	READ_CURSOR	ENDP
	3689		-----
	3690	;	SET COLOR ;
	3691	;	THIS ROUTINE WILL ESTABLISH THE BACKGROUND COLOR, THE OVERSCAN ;
	3692	;	COLOR, AND THE FOREGROUND COLOR SET FOR MEDIUM RESOLUTION ;
	3693	;	GRAPHICS ;
	3694	;	INPUT ;
	3695	;	(BH) HAS COLOR ID ;
	3696	;	IF BH=0, THE BACKGROUND COLOR VALUE IS SET ;
	3697	;	FROM THE LOW BITS OF BL (0-31) ;
	3698	;	IF BH=1, THE PALETTE SELECTION IS MADE ;
	3699	;	BASED ON THE LOW BIT OF BL: ;
	3700	;	0=GREEN, RED, YELLOW FOR COLORS 1,2,3 ;
	3701	;	1=BLUE, CYAN, MAGENTA FOR COLORS 1,2,3 ;
	3702	;	(BL) HAS THE COLOR VALUE TO BE USED ;
	3703	;	OUTPUT ;
	3704	;	THE COLOR SELECTION IS UPDATED ;
	3705		-----
F24E	3706	SET_COLOR	PROC NEAR
F24E 8B166300	3707	MOV	DX,ADDR_6845 ; I/O PORT FOR PALETTE
F252 83C205	3708	ADD	DX,5 ; OVERSCAN PORT
F255 A06600	3709	MOV	AL,CRT_PALETTE ; GET THE CURRENT PALETTE VALUE
F258 0AFF	3710	OR	BH,BH ; IS THIS COLOR 0?
F25A 750E	3711	JNZ	M20 ; OUTPUT COLOR 1
	3712		
	3713	i-----	HANDLE COLOR 0 BY SETTING THE BACKGROUND COLOR
	3714		
F25C 24E0	3715	AND	AL,0E0H ; TURN OFF LOW 5 BITS OF CURRENT
F25E 80E31F	3716	AND	BL,01FH ; TURN OFF HIGH 3 BITS OF INPUT VALUE
F261 0AC3	3717	OR	AL,BL ; PUT VALUE INTO REGISTER
F263	3718		;
F263 EE	3719	M19:	OUT DX,AL ; OUTPUT THE PALETTE
F264 A26600	3720	MOV	CRT_PALETTE,AL ; OUTPUT COLOR SELECTION TO 3D9 PORT
F267 E95BFF	3721	JMP	VIDEO_RETURN ; SAVE THE COLOR VALUE
	3722		
	3723	i-----	HANDLE COLOR 1 BY SELECTING THE PALETTE TO BE USED
	3724		

LOC OBJ	LINE	SOURCE
F26A	3725	M20:
F26A 24DF	3726	AND AL,0DFH ; TURN OFF PALETTE SELECT BIT
F26C D0EB	3727	SHR BL,1 ; TEST THE LOW ORDER BIT OF BL
F26E 73F3	3728	JNC M19 ; ALREADY DONE
F270 0C20	3729	OR AL,20H ; TURN ON PALETTE SELECT BIT
F272 EBEF	3730	JMP M19 ; GO DO IT
	3731	SET_COLOR ENDP
	3732	-----
	3733	; VIDEO STATE :
	3734	; RETURNS THE CURRENT VIDEO STATE IN AX :
	3735	; AH = NUMBER OF COLUMNS ON THE SCREEN :
	3736	; AL = CURRENT VIDEO MODE :
	3737	; BH = CURRENT ACTIVE PAGE :
	3738	-----
F274	3739	VIDEO_STATE PROC NEAR
F274 8A264A00	3740	MOV AH,BYTE PTR CRT_COLS ; GET NUMBER OF COLUMNS
F278 A04900	3741	MOV AL,CRT_MODE ; CURRENT MODE
F27B 8A3E6200	3742	MOV BH,ACTIVE_PAGE ; GET CURRENT ACTIVE PAGE
F27F 5F	3743	POP DI ; RECOVER REGISTERS
F280 5E	3744	POP SI
F281 59	3745	POP CX ; DISCARD SAVED BX
F282 E943FF	3746	JMP M15 ; RETURN TO CALLER
	3747	VIDEO_STATE ENDP
	3748	-----
	3749	; POSITION :
	3750	; THIS SERVICE ROUTINE CALCULATES THE REGEN :
	3751	; BUFFER ADDRESS OF A CHARACTER IN THE ALPHA MODE :
	3752	; INPUT :
	3753	; AX = ROW, COLUMN POSITION :
	3754	; OUTPUT :
	3755	; AX = OFFSET OF CHAR POSITION IN REGEN BUFFER :
	3756	-----
F285	3757	POSITION PROC NEAR
F285 53	3758	PUSH BX ; SAVE REGISTER
F286 8B08	3759	MOV BX,AX
F288 8AC4	3760	MOV AL,AH ; ROWS TO AL
F28A F6264A00	3761	MJL BYTE PTR CRT_COLS ; DETERMINE BYTES TO ROW
F28E 32FF	3762	XOR BH,BH
F290 03C3	3763	ADD AX,BX ; ADD IN COLUMN VALUE
F292 D1E0	3764	SAL AX,1 ; * 2 FOR ATTRIBUTE BYTES
F294 5B	3765	POP BX
F295 C3	3766	RET
	3767	POSITION ENDP
	3768	-----
	3769	; SCROLL UP :
	3770	; THIS ROUTINE MOVES A BLOCK OF CHARACTERS UP :
	3771	; ON THE SCREEN :
	3772	; INPUT :
	3773	; (AH) = CURRENT CRT MODE :
	3774	; (AL) = NUMBER OF ROWS TO SCROLL :
	3775	; (CX) = ROW/COLUMN OF UPPER LEFT CORNER :
	3776	; (DX) = ROW/COLUMN OF LOWER RIGHT CORNER :
	3777	; (BH) = ATTRIBUTE TO BE USED ON BLANKED LINE :
	3778	; (DS) = DATA SEGMENT :
	3779	; (ES) = REGEN BUFFER SEGMENT :
	3780	; OUTPUT :
	3781	; NONE -- THE REGEN BUFFER IS MODIFIED :
	3782	-----
	3783	ASSUME CS:CODE,DS:DATA,ES:DATA
F296	3784	SCROLL_UP PROC NEAR
F296 8AD8	3785	MOV BL,AL ; SAVE LINE COUNT IN BL
F298 80FC04	3786	CMF AH,4 ; TEST FOR GRAPHICS MODE
F29B 7208	3787	JC N1 ; HANDLE SEPARATELY
F29D 80FC07	3788	CMF AH,7 ; TEST FOR BM CARD
F2A0 7403	3789	JE N1
F2A2 E9F001	3790	JMP GRAPHICS_UP
F2A5	3791	N1: ; UP_CONTINUE
F2A5 53	3792	PUSH BX ; SAVE FILL ATTRIBUTE IN BH
F2A6 8BC1	3793	MOV AX,CX ; UPPER LEFT POSITION
F2A8 E83700	3794	CALL SCROLL_POSITION ; DO SETUP FOR SCROLL
F2AB 7431	3795	JZ N7 ; BLANK_FIELD
F2AD 03F0	3796	ADD SI,AX ; FROM ADDRESS
F2AF 8AE6	3797	MOV AH,DH ; # ROWS IN BLOCK
F2B1 2AE3	3798	SUB AH,BL ; # ROWS TO BE MOVED
F2B3	3799	N2: ; ROW_LOOP
F2B3 E87200	3800	CALL N10 ; MOVE ONE ROW
F2B6 03F5	3801	ADD SI,BP

LOC OBJ	LINE	SOURCE			
F2B5 03FD	3602	ADD	DI,BP		; POINT TO NEXT LINE IN BLOCK
F2BA FECC	3603	DEC	AH		; COUNT OF LINES TO MOVE
F2BC 75F5	3604	JNZ	N2		; ROW_LOOP
F2BE	3605	N3:			; CLEAR_ENTRY
F2BE 58	3606	POP	AX		; RECOVER ATTRIBUTE IN AH
F2BF B020	3607	MOV	AL,' '		; FILL WITH BLANKS
F2C1	3608	N4:			; CLEAR_LOOP
F2C1 E86D00	3609	CALL	N11		; CLEAR THE ROW
F2C4 03FD	3610	ADD	DI,BP		; POINT TO NEXT LINE
F2C6 FECC	3611	DEC	BL		; COUNTER OF LINES TO SCROLL
F2C8 75F7	3612	JNZ	N4		; CLEAR_LOOP
F2CA	3613	N5:			; SCROLL_END
F2CA E88C07	3614	CALL	DDS		
F2CD 803E490007	3615	CHP	CRT_MODE,7		; IS THIS THE BLACK AND WHITE CARD
F2D2 7407	3616	JE	N6		; IF SO, SKIP THE MODE RESET
F2D4 A06500	3617	MOV	AL,CRT_MODE_SET		; GET THE VALUE OF THE MODE SET
F2D7 BAD803	3618	MOV	DX,03D8H		; ALWAYS SET COLOR CARD PORT
F2DA EE	3619	OUT	DX,AL		
F2DB	3620	N6:			; VIDEO_RET_HERE
F2DB E9E7FE	3621	JMP	VIDEO_RETURN		
F2DE	3622	N7:			; BLANK_FIELD
F2DE 8ADE	3623	MOV	BL,DH		; GET ROW COUNT
F2E0 E8DC	3624	JMP	N3		; GO CLEAR THAT AREA
	3625	SCROLL_UP	ENDP		
	3626				
	3627				;----- HANDLE COMMON SCROLL SET UP HERE
	3628				
F2E2	3629	SCROLL_POSITION PROC	NEAR		
F2E2 803E490002	3630	CHP	CRT_MODE,2		; TEST FOR SPECIAL CASE HERE
F2E7 7218	3631	JB	N9		; HAVE TO HANDLE 80X25 SEPARATELY
F2E9 803E490003	3632	CHP	CRT_MODE,3		
F2EE 7711	3633	JA	N9		
	3634				
	3635				;----- 80X25 COLOR CARD SCROLL
	3636				
F2F0 52	3637	PUSH	DX		
F2F1 BADA03	3638	MOV	DX,3DAH		; GUARANTEED TO BE COLOR CARD HERE
F2F4 50	3639	PUSH	AX		
F2F5	3640	N8:			; WAIT_DISP_ENABLE
F2F5 EC	3641	IN	AL,DX		; GET PORT
F2F6 A808	3642	TEST	AL,8		; WAIT FOR VERTICAL RETRACE
F2F8 74FB	3643	JZ	N8		; WAIT_DISP_ENABLE
F2FA B025	3644	MOV	AL,25H		
F2FC B2D8	3645	MOV	DL,0D8H		; DX=308
F2FE EE	3646	OUT	DX,AL		; TURN OFF VIDEO
F2FF 58	3647	POP	AX		; DURING VERTICAL RETRACE
F300 5A	3648	POP	DX		
F301	3649	N9:			
F301 E881FF	3650	CALL	POSITION		; CONVERT TO REGEN POINTER
F304 03064E00	3651	ADD	AX,CRT_START		; OFFSET OF ACTIVE PAGE
F308 8BF8	3652	MOV	DI,AX		; TO ADDRESS FOR SCROLL
F30A 8BF0	3653	MOV	SI,AX		; FROM ADDRESS FOR SCROLL
F30C 2BD1	3654	SUB	DX,CX		; DX = # ROWS, #COLS IN BLOCK
F30E FEC6	3655	INC	DH		
F310 FEC2	3656	INC	DL		; INCREMENT FOR 0 ORIGIN
F312 32ED	3657	XOR	CH,CH		; SET HIGH BYTE OF COUNT TO ZERO
F314 8B2E4A00	3658	MOV	BP,CRT_COLS		; GET NUMBER OF COLUMNS IN DISPLAY
F318 03ED	3659	ADD	BP,BP		; TIMES 2 FOR ATTRIBUTE BYTE
F31A 8AC3	3660	MOV	AL,BL		; GET LINE COUNT
F31C F6264A00	3661	MUL	BYTE PTR CRT_COLS		; DETERMINE OFFSET TO FROM ADDRESS
F320 03C0	3662	ADD	AX,AX		; *2 FOR ATTRIBUTE BYTE
F322 06	3663	PUSH	ES		; ESTABLISH ADDRESSING TO REGEN BUFFER
F323 1F	3664	POP	DS		; FOR BOTH POINTERS
F324 80FB00	3665	CHP	BL,0		; 0 SCROLL MEANS BLANK FIELD
F327 C3	3666	RET			; RETURN WITH FLAGS SET
	3667	SCROLL_POSITION ENDP			
	3668				
	3669				;----- MOVE_ROW
	3670				
F328	3671	N10	PROC	NEAR	
F328 8ACA	3672	MOV	CL,DL		; GET # OF COLS TO MOVE
F32A 56	3673	PUSH	SI		
F32B 57	3674	PUSH	DI		; SAVE START ADDRESS
F32C F3	3675	REP	MOVSM		; MOVE THAT LINE ON SCREEN
F32D A5					
F32E 5F	3676	POP	DI		
F32F 5E	3677	POP	SI		; RECOVER ADDRESSES

```

LOC OBJ          LINE   SOURCE
F330 C3          3078      RET
                 3079      N10   ENDP
                 3080
                 3081      ;----- CLEAR_ROW
                 3082
F331             3083      N11   PROC   NEAR
F331 8ACA        3084          MOV   CL,DL           ; GET # COLUMNS TO CLEAR
F333 57          3085          PUSH  DI
F334 F3          3086          REP   STOSW           ; STORE THE FILL CHARACTER
F335 AB
F336 5F          3087          POP   DI
F337 C3          3088      RET
                 3089      N11   ENDP
                 3090
                 3091      ;-----
                 3091      ; SCROLL_DOWN                               :
                 3092      ; THIS ROUTINE MOVES THE CHARACTERS WITHIN A   :
                 3093      ; DEFINED BLOCK DOWN ON THE SCREEN, FILLING THE :
                 3094      ; TOP LINES WITH A DEFINED CHARACTER         :
                 3095      ; INPUT                                         :
                 3096      ; (AH) = CURRENT CRT MODE                     :
                 3097      ; (AL) = NUMBER OF LINES TO SCROLL          :
                 3098      ; (CX) = UPPER LEFT CORNER OF REGION       :
                 3099      ; (DX) = LOWER RIGHT CORNER OF REGION      :
                 3100      ; (BH) = FILL CHARACTER                       :
                 3101      ; (DS) = DATA SEGMENT                       :
                 3102      ; (ES) = REGEN SEGMENT                     :
                 3103      ; OUTPUT                                         :
                 3104      ; NONE -- SCREEN IS SCROLLED                :
                 3105      ;-----
F338             3106      SCROLL_DOWN  PROC   NEAR
F338 FD          3107          STD                               ; DIRECTION FOR SCROLL DOWN
F339 8AD8        3108          MOV   BL,AL           ; LINE COUNT TO BL
F33B 80FC04     3109          CMP   AH,4           ; TEST FOR GRAPHICS
F33E 7208       3110          JC    N12
F340 80FC07     3111          CMP   AH,7           ; TEST FOR BW CARD
F343 7403       3112          JE    N12
F345 E9A601     3113          JMP   GRAPHICS_DOWN
F346            3114      N12:                               ; CONTINUE_DOWN
F346 53          3115          PUSH  BX           ; SAVE ATTRIBUTE IN BH
F349 0BC2       3116          MOV   AX,DX         ; LOWER RIGHT CORNER
F34B E894FF     3117          CALL  SCROLL_POSITION ; GET REGEN LOCATION
F34E 7420       3118          JZ    N16
F350 2BF0       3119          SUB   SI,AX         ; SI IS FROM ADDRESS
F352 8AE6       3120          MOV   AH,DH         ; GET TOTAL # ROWS
F354 2AE3       3121          SUB   AH,BL         ; COUNT TO MOVE IN SCROLL
F356            3122      N13:
F356 E0CFFF     3123          CALL  N10           ; MOVE ONE ROW
F359 2BF5       3124          SUB   SI,BP
F35B 2BFD       3125          SUB   DI,BP
F35D FECC       3126          DEC   AH
F35F 75F5       3127          JNZ  N13
F361            3128      N14:
F361 58          3129          POP   AX           ; RECOVER ATTRIBUTE IN AH
F362 B020       3130          MOV   AL,' '
F364            3131      N15:
F364 E8CAFF     3132          CALL  N11           ; CLEAR ONE ROW
F367 2BFD       3133          SUB   DI,BP         ; GO TO NEXT ROW
F369 FECB       3134          DEC   BL
F36B 75F7       3135          JNZ  N15
F36D E95AFF     3136          JMP   N5           ; SCROLL_END
F370            3137      N16:
F370 8ADE        3138          MOV   BL,DH
F372 EBED       3139          JMP   N14
                 3140      SCROLL_DOWN  ENDP
                 3141      ;-----
                 3142      ; READ_AC_CURRENT                               :
                 3143      ; THIS ROUTINE READS THE ATTRIBUTE AND CHARACTER :
                 3144      ; AT THE CURRENT CURSOR POSITION AND RETURNS THEN :
                 3145      ; TO THE CALLER                               :
                 3146      ; INPUT                                         :
                 3147      ; (AH) = CURRENT CRT MODE                     :
                 3148      ; (BH) = DISPLAY PAGE ( ALPHA MODES ONLY )    :
                 3149      ; (DS) = DATA SEGMENT                       :
                 3150      ; (ES) = REGEN SEGMENT                     :
                 3151      ; OUTPUT                                         :
                 3152      ; (AL) = CHAR READ                           :
                 3153      ; (AH) = ATTRIBUTE READ                       :
                 3154      ;-----

```

```

3955          ASSUME  CS:CODE,DS:DATA,ES:DATA
F374          3956  READ_AC_CURRENT PROC  NEAR
F374 80FC04   3957      CMP  AH,4          ; IS THIS GRAPHICS
F377 7208     3958      JC   P1
F379 80FC07   3959      CMP  AH,7          ; IS THIS BW CARD
F37C 7403     3960      JE   P1
F37E E9A802   3961      JMP  GRAPHICS_READ
F381          3962      P1:          ; READ_AC_CONTINUE
F381 E81A00   3963      CALL  FIND_POSITION
F384 8BF3     3964      MOV  SI,BX          ; ESTABLISH ADDRESSING IN SI
3965
3966          ;----- WAIT FOR HORIZONTAL RETRACE
3967
F386 8B166300 3968      MOV  DX,ADDR_6845    ; GET BASE ADDRESS
F38A 83C206   3969      ADD  DX,6            ; POINT AT STATUS PORT
F38D 06       3970      PUSH ES
F38E 1F       3971      POP  DS            ; GET SEGMENT FOR QUICK ACCESS
F38F          3972      P2:          ; WAIT FOR RETRACE LOW
F38F EC       3973      IN   AL,DX           ; GET STATUS
F390 A801     3974      TEST  AL,1            ; IS HORZ RETRACE LOW
F392 75FB     3975      JNZ  P2              ; WAIT UNTIL IT IS
F394 FA       3976      CLI
3995          3977      P3:          ; WAIT FOR RETRACE HIGH
F395 EC       3978      IN   AL,DX           ; GET STATUS
F396 A801     3979      TEST  AL,1            ; IS IT HIGH
F398 74FB     3980      JZ   P3              ; WAIT UNTIL IT IS
F39A AD       3981      LODSB
F39B E927FE   3982      JMP  VIDEO_RETURN
3983          3983      READ_AC_CURRENT ENDP
3984
F39E          3985      FIND_POSITION PROC  NEAR
F39E 8ACF     3986      MOV  CL,BH            ; DISPLAY PAGE TO CX
F3A0 32ED     3987      XOR  CH,CH
F3A2 8BF1     3988      MOV  SI,CX            ; MOVE TO SI FOR INDEX
F3A4 D1E6     3989      SAL  SI,1            ; * 2 FOR WORD OFFSET
F3A6 8B4450   3990      MOV  AX,[SI+OFFSET CURSOR_POSN] ; GET ROW/COLUMN OF THAT PAGE
F3A9 330B     3991      XOR  BX,BX            ; SET START ADDRESS TO ZERO
F3AB E306     3992      JCXZ P5              ; NO_PAGE
F3AD          3993      P4:          ; PAGE_LOOP
F3AD 031E4C00 3994      ADD  BX,CRT_LEN      ; LENGTH OF BUFFER
F3B1 E2FA     3995      LOOP P4
F3B3          3996      P5:          ; NO_PAGE
F3B3 E8CFFE   3997      CALL POSITION          ; DETERMINE LOCATION IN REGEN
F3B6 03DB     3998      ADD  BX,AX            ; ADD TO START OF REGEN
F3B8 C3       3999      RET
4000          4000      FIND_POSITION  ENDP
4001          ;-----
4002      ; WRITE_AC_CURRENT          :
4003      ; THIS ROUTINE WRITES THE ATTRIBUTE :
4004      ; AND CHARACTER AT THE CURRENT CURSOR :
4005      ; POSITION :
4006      ; INPUT :
4007      ; (AH) = CURRENT CRT MODE :
4008      ; (BH) = DISPLAY PAGE :
4009      ; (CX) = COUNT OF CHARACTERS TO WRITE :
4010      ; (AL) = CHAR TO WRITE :
4011      ; (BL) = ATTRIBUTE OF CHAR TO WRITE :
4012      ; (DS) = DATA SEGMENT :
4013      ; (ES) = REGEN SEGMENT :
4014      ; OUTPUT :
4015      ; NONE :
4016      ;-----
F3B9          4017      WRITE_AC_CURRENT PROC  NEAR
F3B9 80FC04   4018      CMP  AH,4          ; IS THIS GRAPHICS
F3BC 7208     4019      JC   P6
F3BE 80FC07   4020      CMP  AH,7          ; IS THIS BW CARD
F3C1 7403     4021      JE   P6
F3C3 E9B201   4022      JMP  GRAPHICS_WRITE
F3C6          4023      P6:          ; WRITE_AC_CONTINUE
F3C6 8AE3     4024      MOV  AH,BL            ; GET ATTRIBUTE TO AH
F3C8 50       4025      PUSH AX              ; SAVE ON STACK
F3C9 51       4026      PUSH CX              ; SAVE WRITE COUNT
F3CA E8D1FF   4027      CALL  FIND_POSITION
F3CD 8BF8     4028      MOV  DI,BX            ; ADDRESS TO DI REGISTER
F3CF 59       4029      POP  CX              ; WRITE COUNT
F3D0 5B       4030      POP  BX              ; CHARACTER IN BX REG

```

```

LOC OBJ          LINE    SOURCE

F3D1             4031    P7:                ; WRITE_LOOP
                4032
                4033    ;----- WAIT FOR HORIZONTAL RETRACE
                4034
F3D1 8B166300    4035        MOV    DX,ADDR_6845    ; GET BASE ADDRESS
F3D5 83C206      4036        ADD    DX,6            ; POINT AT STATUS PORT
F3D8             4037    P8:
F3D8 EC          4038        IN     AL,DX          ; GET STATUS
F3D9 A801        4039        TEST   AL,1            ; IS IT LOW
F3DB 75FB        4040        JNZ   P8              ; WAIT UNTIL IT IS
F3DD FA          4041        CLI    P8              ; NO MORE INTERRUPTS
F3DE             4042    P9:
F3DE EC          4043        IN     AL,DX          ; GET STATUS
F3DF A801        4044        TEST   AL,1            ; IS IT HIGH
F3E1 74FB        4045        JZ    P9              ; WAIT UNTIL IT IS
F3E3 8BC3        4046        MOV    AX,BX          ; RECOVER THE CHAR/ATTR
F3E5 AB          4047        STOSW                ; PUT THE CHAR/ATTR
F3E6 FB          4048        STI    P8              ; INTERRUPTS BACK ON
F3E7 E2E8        4049        LOOP  P7              ; AS MANY TIMES AS REQUESTED
F3E9 E9D9FD      4050        JMP    VIDEO_RETURN
                4051    WRITE_AC_CURRENT    ENDP
                4052    ;-----
                4053    ; WRITE_C_CURRENT    :
                4054    ;     THIS ROUTINE WRITES THE CHARACTER AT    :
                4055    ;     THE CURRENT CURSOR POSITION, ATTRIBUTE  :
                4056    ;     UNCHANGED                                :
                4057    ;     INPUT                                    :
                4058    ;     (AH) = CURRENT CRT MODE                    :
                4059    ;     (BH) = DISPLAY PAGE                          :
                4060    ;     (CX) = COUNT OF CHARACTERS TO WRITE      :
                4061    ;     (AL) = CHAR TO WRITE                          :
                4062    ;     (DS) = DATA SEGMENT                            :
                4063    ;     (ES) = REGEN SEGMENT                            :
                4064    ;     OUTPUT                                    :
                4065    ;     NONE                                        :
                4066    ;-----
F3EC             4067    WRITE_C_CURRENT PROC    NEAR
F3EC 80FC04      4068        CMP    AH,4          ; IS THIS GRAPHICS
F3EF 7208        4069        JC    P10
F3F1 80FC07      4070        CMP    AH,7          ; IS THIS BW CARD
F3F4 7403        4071        JE    P10
F3F6 E97F01      4072        JMP    GRAPHICS_WRITE
F3F9             4073    P10:
F3F9 50          4074        PUSH   AX            ; SAVE ON STACK
F3FA 51          4075        PUSH   CX            ; SAVE WRITE COUNT
F3FB 8BA0FF      4076        CALL  FIND_POSITION
F3FE 8BFB        4077        MOV    DI,BX          ; ADDRESS TO DI
F400 59          4078        POP    CX            ; WRITE COUNT
F401 5B          4079        POP    BX            ; BL HAS CHAR TO WRITE
F402             4080    P11:                ; WRITE_LOOP
                4081
                4082    ;----- WAIT FOR HORIZONTAL RETRACE
                4083
F402 8B166300    4084        MOV    DX,ADDR_6845    ; GET BASE ADDRESS
F406 83C206      4085        ADD    DX,6            ; POINT AT STATUS PORT
F409             4086    P12:
F409 EC          4087        IN     AL,DX          ; GET STATUS
F40A A801        4088        TEST   AL,1            ; IS IT LOW
F40C 75FB        4089        JNZ   P12            ; WAIT UNTIL IT IS
F40E FA          4090        CLI    P12            ; NO MORE INTERRUPTS
F40F             4091    P13:
F40F EC          4092        IN     AL,DX          ; GET STATUS
F410 A801        4093        TEST   AL,1            ; IS IT HIGH
F412 74FB        4094        JZ    P13            ; WAIT UNTIL IT IS
F414 8AC3        4095        MOV    AL,BL          ; RECOVER CHAR
F416 AA          4096        STOSB                ; PUT THE CHAR/ATTR
F417 FB          4097        STI    P12            ; INTERRUPTS BACK ON
F418 47          4098        INC    DI            ; BUMP POINTER PAST ATTRIBUTE
F419 E2E7        4099        LOOP  P11            ; AS MANY TIMES AS REQUESTED
F41B E9A7FD      4100        JMP    VIDEO_RETURN
                4101    WRITE_C_CURRENT    ENDP
                4102    ;-----
                4103    ; READ DOT -- WRITE DOT    :
                4104    ;     THESE ROUTINES WILL WRITE A DOT, OR READ THE DOT AT    :
                4105    ;     THE INDICATED LOCATION                                :
                4106    ;     ENTRY --                                            :
                4107    ;     DX = ROW (0-199)    (THE ACTUAL VALUE DEPENDS ON THE MODE) :

```

```

4108 ; CX = COLUMN ( 0-639 ) ( THE VALUES ARE NOT RANGE CHECKED ) :
4109 ; AL = DOT VALUE TO WRITE (1,2 OR 4 BITS DEPENDING ON MODE, :
4110 ; REQ'D FOR WRITE DOT ONLY, RIGHT JUSTIFIED) :
4111 ; BIT 7 OF AL=1 INDICATES XOR THE VALUE INTO THE LOCATION :
4112 ; DS = DATA SEGMENT :
4113 ; ES = REGEN SEGMENT :
4114 ; :
4115 ; EXIT :
4116 ; AL = DOT VALUE READ, RIGHT JUSTIFIED, READ ONLY :
4117 ;-----
4118 ASSUME CS:CODE,DS:DATA,ES:DATA
4119 READ_DOT PROC NEAR
4120 CALL R3 ; DETERMINE BYTE POSITION OF DOT
4121 MOV AL,ES:[SI] ; GET THE BYTE
4122 AND AL,AH ; MASK OFF THE OTHER BITS IN THE BYTE
4123 SHL AL,CL ; LEFT JUSTIFY THE VALUE
4124 MOV CL,DH ; GET NUMBER OF BITS IN RESULT
4125 ROL AL,CL ; RIGHT JUSTIFY THE RESULT
4126 JMP VIDEO_RETURN ; RETURN FROM VIDEO IO
4127 READ_DOT ENDP
4128
4129 WRITE_DOT PROC NEAR
4130 PUSH AX ; SAVE DOT VALUE
4131 PUSH AX ; TWICE
4132 CALL R3 ; DETERMINE BYTE POSITION OF THE DOT
4133 SHR AL,CL ; SHIFT TO SET UP THE BITS FOR OUTPUT
4134 AND AL,AH ; STRIP OFF THE OTHER BITS
4135 MOV CL,ES:[SI] ; GET THE CURRENT BYTE
4136 POP BX ; RECOVER XOR FLAG
4137 TEST BL,80H ; IS IT ON
4138 JNZ R2 ; YES, XOR THE DOT
4139 NOT AH ; SET THE MASK TO REMOVE THE
4140 AND CL,AH ; INDICATED BITS
4141 OR AL,CL ; OR IN THE NEW VALUE OF THOSE BITS
4142 R1: ; FINISH_DOT
4143 MOV ES:[SI],AL ; RESTORE THE BYTE IN MEMORY
4144 POP AX
4145 JMP VIDEO_RETURN ; RETURN FROM VIDEO IO
4146 R2: ; XOR_DOT
4147 XOR AL,CL ; EXCLUSIVE OR THE DOTS
4148 JMP R1 ; FINISH UP THE WRITING
4149 WRITE_DOT ENDP
4150 ;-----
4151 ; THIS SUBROUTINE DETERMINES THE REGEN BYTE LOCATION :
4152 ; OF THE INDICATED ROW COLUMN VALUE IN GRAPHICS MODE. :
4153 ; ENTRY -- :
4154 ; DX = ROW VALUE (0-199) :
4155 ; CX = COLUMN VALUE (0-639) :
4156 ; EXIT -- :
4157 ; SI = OFFSET INTO REGEN BUFFER FOR BYTE OF INTEREST :
4158 ; AH = MASK TO STRIP OFF THE BITS OF INTEREST :
4159 ; CL = BITS TO SHIFT TO RIGHT JUSTIFY THE MASK IN AH :
4160 ; DH = # BITS IN RESULT :
4161 ;-----
4162 R3 PROC NEAR
4163 PUSH BX ; SAVE BX DURING OPERATION
4164 PUSH AX ; WILL SAVE AL DURING OPERATION
4165
4166 ;---- DETERMINE 1ST BYTE IN DICATED ROW BY MULTIPLYING ROW VALUE BY 40
4167 ;---- ( LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW
4168
4169 MOV AL,40
4170 PUSH DX ; SAVE ROW VALUE
4171 AND DL,0FEH ; STRIP OFF ODD/EVEN BIT
4172 MUL DL ; AX HAS ADDRESS OF 1ST BYTE
4173 ; OF INDICATED ROW
4174 POP DX ; RECOVER IT
4175 TEST DL,1 ; TEST FOR EVEN/ODD
4176 JZ R4 ; JUMP IF EVEN ROW
4177 ADD AX,2000H ; OFFSET TO LOCATION OF ODD ROWS
4178 R4: ; EVEN_ROW
4179 MOV SI,AX ; MOVE POINTER TO SI
4180 POP AX ; RECOVER AL VALUE
4181 MOV DX,CX ; COLUMN VALUE TO DX
4182
4183 ;---- DETERMINE GRAPHICS MODE CURRENTLY IN EFFECT
4184

```

```

LOC OBJ          LINE  SOURCE
4185             |-----|
4186             ; SET UP THE REGISTERS ACCORDING TO THE MODE          :
4187             ; CH = MASK FOR LOW OF COLUMN ADDRESS ( 7/3 FOR HIGH/MED RES) :
4188             ; CL = # OF ADDRESS BITS IN COLUMN VALUE ( 3/2 FOR H/M)      :
4189             ; BL = MASK TO SELECT BITS FROM POINTED BYTE ( 80H/COH FOR H/M) :
4190             ; BH = NUMBER OF VALID BITS IN POINTED BYTE ( 1/2 FOR H/M)    :
4191             |-----|
4192
F46A BBC002     4193             MOV     BX,2C0H
F46D B90203     4194             MOV     CX,302H             ; SET PARMS FOR MED RES
F470 803E490006 4195             CMP     CRT_MODE,6
F475 7206       4196             JC      R5                 ; HANDLE IF MED ARES
F477 BB8001     4197             MOV     BX,180H
F47A B90307     4198             MOV     CX,703H             ; SET PARMS FOR HIGH RES
4199
4200             ;----- DETERMINE BIT OFFSET IN BYTE FROM COLUMN MASK
4201
F47D             4202             R5:
F47D 22EA       4203             AND     CH,DL             ; ADDRESS OF PEL WITHIN BYTE TO CH
4204
4205             ;----- DETERMINE BYTE OFFSET FOR THIS LOCATION IN COLUMN
4206
F47F D3EA       4207             SHR     DX,CL             ; SHIFT BY CORRECT AMOUNT
F481 03F2       4208             ADD     SI,DX            ; INCREMENT THE POINTER
F483 0AF7       4209             MOV     DH,BH            ; GET THE # OF BITS IN RESULT TO DH
4210
4211             ;----- MULTIPLY BH (VALID BITS IN BYTE) BY CH (BIT OFFSET)
4212
F485 2AC9       4213             SUB     CL,CL            ; ZERO INTO STORAGE LOCATION
F487             4214             R6:
F487 D0C8       4215             ROR     AL,1             ; LEFT JUSTIFY THE VALUE
4216             ; IN AL (FOR WRITE)
F489 02CD       4217             ADD     CL,CH            ; ADD IN THE BIT OFFSET VALUE
F48B FECF       4218             DEC     BH               ; LOOP CONTROL
F48D 75F8       4219             JNZ    R6               ; ON EXIT, CL HAS SHIFT COUNT
4220             ; TO RESTORE BITS
F48F 8AE3       4221             MOV     AH,BL            ; GET MASK TO AH
F491 D2EC       4222             SHR     AH,CL            ; MOVE THE MASK TO CORRECT LOCATION
F493 5B         4223             POP     BX               ; RECOVER REG
F494 C3         4224             RET                    ; RETURN WITH EVERYTHING SET UP
4225             R3      ENDP
4226             |-----|
4227             ; SCROLL UP
4228             ; THIS ROUTINE SCROLLS UP THE INFORMATION ON THE CRT
4229             ; ENTRY
4230             ; CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
4231             ; DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
4232             ; BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
4233             ; BH = FILL VALUE FOR BLANKED LINES
4234             ; AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE
4235             ; FIELD)
4236             ; DS = DATA SEGMENT
4237             ; ES = REGEN SEGMENT
4238             ; EXIT
4239             ; NOTHING, THE SCREEN IS SCROLLED
4240             |-----|
F495             4241             GRAPHICS_UP   PROC   NEAR
F495 8AD8       4242             MOV     BL,AL            ; SAVE LINE COUNT IN BL
F497 8BC1       4243             MOV     AX,CX            ; GET UPPER LEFT POSITION INTO AX REG
4244
4245             ;----- USE CHARACTER SUBROUTINE FOR POSITIONING
4246             ;----- ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
4247
F499 E86902     4248             CALL    GRAPH_POSN
F49C 8BF8       4249             MOV     DI,AX            ; SAVE RESULT AS DESTINATION ADDRESS
4250
4251             ;----- DETERMINE SIZE OF WINDOW
4252
F49E 2BD1       4253             SUB     DX,CX
F4A0 81C20101   4254             ADD     DX,101H          ; ADJUST VALUES
F4A4 D0E6       4255             SAL     DH,1             ; MULTIPLY # ROWS BY 4
4256             ; SINCE 8 VERT DOTS/CHAR
F4A6 D0E6       4257             SAL     DH,1             ; AND EVEN/ODD ROWS
4258
4259             ;----- DETERMINE CRT MODE
4260
F4A8 803E490006 4261             CMP     CRT_MODE,6       ; TEST FOR MEDIUM RES

```



```

LOC OBJ          LINE  SOURCE
F4AD 7304        4262          JNC   R7              ; FIND_SOURCE
4263
4264          ;----- MEDIUM RES UP
4265
F4AF D0E2        4266          SAL   DL,1             ; # COLUMNS * 2, SINCE 2 BYTES/CHAR
F4B1 D1E7        4267          SAL   DI,1             ; OFFSET #2 SINCE 2 BYTES/CHAR
4268
4269          ;----- DETERMINE THE SOURCE ADDRESS IN THE BUFFER
4270
F4B3            4271          R7:              ; FIND_SOURCE
F4B3 06          4272          PUSH  ES              ; GET SEGMENTS BOTH POINTING TO REGEN
F4B4 1F          4273          POP   DS
F4B5 2AED        4274          SUB   CH,CH            ; ZERO TO HIGH OF COUNT REG
F4B7 D0E3        4275          SAL   BL,1             ; MULTIPLY NUMBER OF LINES BY 4
F4B9 D0E3        4276          SAL   BL,1
F4BB 742D        4277          JZ    R11              ; IF ZERO, THEN BLANK ENTIRE FIELD
F4BD 8AC3        4278          MOV   AL,BL            ; GET NUMBER OF LINES IN AL
F4BF B450        4279          MOV   AH,80            ; 80 BYTES/ROW
F4C1 F6E4        4280          MUL   AH               ; DETERMINE OFFSET TO SOURCE
F4C3 8BF7        4281          MOV   SI,DI            ; SET UP SOURCE
F4C5 03F0        4282          ADD   SI,AX            ; ADD IN OFFSET TO IT
F4C7 8AE6        4283          MOV   AH,DH            ; NUMBER OF ROWS IN FIELD
F4C9 2AE3        4284          SUB   AH,BL            ; DETERMINE NUMBER TO MOVE
4285
4286          ;----- LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS
4287
F4CB            4288          R8:              ; ROW_LOOP
F4CB E88000      4289          CALL  R17              ; MOVE ONE ROW
F4CE 81EEB01F    4290          SUB   SI,2000H-80      ; MOVE TO NEXT ROW
F4D2 81EFB01F    4291          SUB   DI,2000H-80
F4D6 FECC        4292          DEC   AH               ; NUMBER OF ROWS TO MOVE
F4D8 75F1        4293          JNZ   R8               ; CONTINUE TILL ALL MOVED
4294
4295          ;----- FILL IN THE VACATED LINE(S)
4296
F4DA            4297          R9:              ; CLEAR_ENTRY
F4DA 8AC7        4298          MOV   AL,BH            ; ATTRIBUTE TO FILL WITH
F4DC            4299          R10:
F4DC E88000      4300          CALL  R18              ; CLEAR THAT ROW
F4DF 81EFB01F    4301          SUB   DI,2000H-80      ; POINT TO NEXT LINE
F4E3 FECC        4302          DEC   BL               ; NUMBER OF LINES TO FILL
F4E5 75F5        4303          JNZ   R10              ; CLEAR_LOOP
F4E7 E90BFC      4304          JMP   VIDEO_RETURN     ; EVERYTHING DONE
F4EA            4305          R11:
F4EA 8ADE        4306          MOV   BL,DH            ; SET BLANK COUNT TO
4307          ; EVERYTHING IN FIELD
F4EC EBEC        4308          JMP   R9               ; CLEAR THE FIELD
4309          GRAPHICS_UP      ENDP
4310          ;-----
4311          ; SCROLL DOWN
4312          ; THIS ROUTINE SCROLLS DOWN THE INFORMATION ON THE CRT
4313          ; ENTRY
4314          ; CH,CL = UPPER LEFT CORNER OF REGION TO SCROLL
4315          ; DH,DL = LOWER RIGHT CORNER OF REGION TO SCROLL
4316          ; BOTH OF THE ABOVE ARE IN CHARACTER POSITIONS
4317          ; BH = FILL VALUE FOR BLANKED LINES
4318          ; AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE
4319          ; FIELD)
4320          ; DS = DATA SEGMENT
4321          ; ES = REGEN SEGMENT
4322          ; EXIT
4323          ; NOTHING, THE SCREEN IS SCROLLED
4324          ;-----
F4EE            4325          GRAPHICS_DOWN      PROC  NEAR
F4EE FD          4326          STD
F4EF 8ADB        4327          MOV   BL,AL            ; SAVE LINE COUNT IN BL
F4F1 8BC2        4328          MOV   AX,DX            ; GET LOWER RIGHT POSITION INTO AX REG
4329
4330          ;----- USE CHARACTER SUBROUTINE FOR POSITIONING
4331          ;----- ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE
4332
F4F3 E80F02      4333          CALL  GRAPH_POSN
F4F6 8BF8        4334          MOV   DI,AX            ; SAVE RESULT AS DESTINATION ADDRESS
4335
4336          ;----- DETERMINE SIZE OF WINDOW
4337
F4F8 2BD1        4338          SUB   DX,CX

```

LOC OBJ	LINE	SOURCE	
F4FA 81C20101	4339	ADD DX,101H	; ADJUST VALUES
F4FE D0E6	4340	SAL DH,1	; MULTIPLY # ROWS BY 4
	4341		; SINCE 8 VERT DOTS/CHAR
F500 D0E6	4342	SAL DH,1	; AND EVEN/ODD ROWS
	4343		
	4344	;----- DETERMINE CRT MODE	
	4345		
F502 803E490006	4346	CMP CRT_MODE,6	; TEST FOR MEDIUM RES
F507 7305	4347	JNC R12	; FIND_SOURCE_DOWN
	4348		
	4349	;----- MEDIUM RES DOWN	
	4350		
F509 D0E2	4351	SAL DL,1	; # COLUMNS * 2, SINCE
	4352		; 2 BYTES/CHAR (OFFSET OK)
F50B D1E7	4353	SAL DI,1	; OFFSET #2 SINCE 2 BYTES/CHAR
F50D 47	4354	INC DI	; POINT TO LAST BYTE
	4355		
	4356	;----- DETERMINE THE SOURCE ADDRESS IN THE BUFFER	
	4357		
F50E	4358	R12:	; FIND_SOURCE_DOWN
F50E 06	4359	PUSH ES	; BOTH SEGMENTS TO REGEN
F50F 1F	4360	POP DS	
F510 2AED	4361	SUB CH,CH	; ZERO TO HIGH OF COUNT REG
F512 81C7F000	4362	ADD DI,240	; POINT TO LAST ROW OF PIXELS
F516 D0E3	4363	SAL BL,1	; MULTIPLY NUMBER OF LINES BY 4
F518 D0E3	4364	SAL BL,1	
F51A 742E	4365	JZ R16	; IF ZERO, THEN BLANK ENTIRE FIELD
F51C 8AC3	4366	MOV AL,BL	; GET NUMBER OF LINES IN AL
F51E B450	4367	MOV AH,80	; 80 BYTES/ROW
F520 F6E4	4368	MUL AH	; DETERMINE OFFSET TO SOURCE
F522 8BF7	4369	MOV SI,DI	; SET UP SOURCE
F524 2BF0	4370	SUB SI,AX	; SUBTRACT THE OFFSET
F526 8AE6	4371	MOV AH,DH	; NUMBER OF ROWS IN FIELD
F528 2AE3	4372	SUB AH,BL	; DETERMINE NUMBER TO MOVE
	4373		
	4374	;----- LOOP THROUGH, MOVING ONE ROW AT A TIME, BOTH EVEN AND ODD FIELDS	
	4375		
F52A	4376	R13:	; ROW_LOOP_DOWN
F52A E82100	4377	CALL R17	; MOVE ONE ROW
F52D 81EE5020	4378	SUB SI,2000H+80	; MOVE TO NEXT ROW
F531 81EF5020	4379	SUB DI,2000H+80	
F535 FECC	4380	DEC AH	; NUMBER OF ROWS TO MOVE
F537 75F1	4381	JNZ R13	; CONTINUE TILL ALL MOVED
	4382		
	4383	;----- FILL IN THE VACATED LINE(S)	
	4384		
F539	4385	R14:	; CLEAR_ENTRY_DOWN
F539 8AC7	4386	MOV AL,BH	; ATTRIBUTE TO FILL WITH
F53B	4387	R15:	; CLEAR_LOOP_DOWN
F53B E82900	4388	CALL R1C	; CLEAR A ROW
F53E 81EF5020	4389	SUB DI,2000H+80	; POINT TO NEXT LINE
F542 FECC	4390	DEC BL	; NUMBER OF LINES TO FILL
F544 75F5	4391	JNZ R15	; CLEAR_LOOP_DOWN
F546 FC	4392	CLD	; RESET THE DIRECTION FLAG
F547 E97BFC	4393	JMP VIDEO_RETURN	; EVERYTHING DONE
F54A	4394	R16:	; BLANK_FIELD_DOWN
F54A 8ADE	4395	MOV BL,DH	; SET BLANK COUNT TO EVERYTHING
	4396		; IN FIELD
F54C EBEB	4397	JMP R14	; CLEAR THE FIELD
	4398	GRAPHICS_DOWN	ENDP
	4399		
	4400	;----- ROUTINE TO MOVE ONE ROW OF INFORMATION	
	4401		
F54E	4402	R17	PROC NEAR
F54E 8ACA	4403	MOV CL,DL	; NUMBER OF BYTES IN THE ROW
F550 56	4404	PUSH SI	
F551 57	4405	PUSH DI	; SAVE POINTERS
F552 F3	4406	REP MOVSB	; MOVE THE EVEN FIELD
F553 A4			
F554 5F	4407	POP DI	
F555 5E	4408	POP SI	
F556 81C60020	4409	ADD SI,2000H	
F55A 81C70020	4410	ADD DI,2000H	; POINT TO THE ODD FIELD
F55E 56	4411	PUSH SI	
F55F 57	4412	PUSH DI	; SAVE THE POINTERS
F560 8ACA	4413	MOV CL,DL	; COUNT BACK
F562 F3	4414	REP MOVSB	; MOVE THE ODD FIELD

LOC OBJ	LINE	SOURCE
F563 A4		
F564 5F	4415	POP DI
F565 5E	4416	POP SI ; POINTERS BACK
F566 C3	4417	RET ; RETURN TO CALLER
	4418	R17 ENDP
	4419	
	4420	;----- CLEAR A SINGLE ROM
	4421	
F567	4422	R18 PROC NEAR
F567 8ACA	4423	MOV CL,DL ; NUMBER OF BYTES IN FIELD
F569 57	4424	PUSH DI ; SAVE POINTER
F56A F3	4425	REP STOSB ; SAVE THE NEW VALUE
F56B AA		
F56C 5F	4426	POP DI ; POINTER BACK
F56D 81C70020	4427	ADD DI,2000H ; POINT TO ODD FIELD
F571 57	4428	PUSH DI
F572 8ACA	4429	MOV CL,DL
F574 F3	4430	REP STOSB ; FILL THE ODD FILED
F575 AA		
F576 5F	4431	POP DI
F577 C3	4432	RET ; RETURN TO CALLER
	4433	R18 ENDP
	4434	;------
	4435	; GRAPHICS WRITE ;
	4436	; THIS ROUTINE WRITES THE ASCII CHARACTER TO THE ;
	4437	; CURRENT POSITION ON THE SCREEN. ;
	4438	; ENTRY ;
	4439	; AL = CHARACTER TO WRITE ;
	4440	; BL = COLOR ATTRIBUTE TO BE USED FOR FOREGROUND COLOR ;
	4441	; IF BIT 7 IS SET, THE CHAR IS XOR'D INTO THE REGEN ;
	4442	; BUFFER (0 IS USED FOR THE BACKGROUND COLOR) ;
	4443	; CX = NUMBER OF CHARS TO WRITE ;
	4444	; DS = DATA SEGMENT ;
	4445	; ES = REGEN SEGMENT ;
	4446	; EXIT ;
	4447	; NOTHING IS RETURNED ;
	4448	; ;
	4449	; GRAPHICS READ ;
	4450	; THIS ROUTINE READS THE ASCII CHARACTER AT THE CURRENT ;
	4451	; CURSOR POSITION ON THE SCREEN BY MATCHING THE DOTS ON ;
	4452	; THE SCREEN TO THE CHARACTER GENERATOR CODE POINTS ;
	4453	; ENTRY ;
	4454	; NONE (0 IS ASSUMED AS THE BACKGROUND COLOR ;
	4455	; EXIT ;
	4456	; AL = CHARACTER READ AT THAT POSITION (0 RETURNED IF ;
	4457	; NONE FOUND) ;
	4458	; ;
	4459	; FOR BOTH ROUTINES, THE IMAGES USED TO FORM CHARS ARE ;
	4460	; CONTAINED IN ROM FOR THE 1ST 128 CHARS. TO ACCESS CHARS ;
	4461	; IN THE SECOND HALF, THE USER MUST INITIALIZE THE VECTOR AT ;
	4462	; INTERRUPT 1FH (LOCATION 0007CH) TO POINT TO THE USER ;
	4463	; SUPPLIED TABLE OF GRAPHIC IMAGES (8X8 BOXES). ;
	4464	; FAILURE TO DO SO WILL CAUSE IN STRANGE RESULTS ;
	4465	;------
	4466	ASSUME CS:CODE,DS:DATA,ES:DATA
F578	4467	GRAPHICS_WRITE PROC NEAR
F578 B400	4468	MOV AH,0 ; ZERO TO HIGH OF CODE POINT
F57A 50	4469	PUSH AX ; SAVE CODE POINT VALUE
	4470	
	4471	;----- DETERMINE POSITION IN REGEN BUFFER TO PUT CODE POINTS
	4472	
F57B E88401	4473	CALL S26 ; FIND LOCATION IN REGEN BUFFER
F57E 8BF8	4474	MOV DI,AX ; REGEN POINTER IN DI
	4475	
	4476	;----- DETERMINE REGION TO GET CODE POINTS FROM
	4477	
F580 58	4478	POP AX ; RECOVER CODE POINT
F581 3C80	4479	CMPL AL,80H ; IS IT IN SECOND HALF
F583 7306	4480	JAE SI ; YES
	4481	
	4482	;----- IMAGE IS IN FIRST HALF, CONTAINED IN ROM
	4483	
F585 BE6EFA	4484	MOV SI,0FA6EH ; CRT_CHAR_GEN (OFFSET OF IMAGES)
F588 0E	4485	PUSH CS ; SAVE SEGMENT ON STACK
F589 EBF0	4486	JMP SHORT S2 ; DETERMINE_MODE
	4487	
	4488	;----- IMAGE IS IN SECOND HALF, IN USER RAM

LOC OBJ	LINE	SOURCE	
	4489		
F5B8	4490	S1:	; EXTEND_CHAR
F5B8 2C80	4491	SUB AL,80H	; ZERO ORIGIN FOR SECOND HALF
F5B8 1E	4492	PUSH DS	; SAVE DATA POINTER
F5B8 2BF6	4493	SUB SI,SI	
F590 8EDE	4494	MOV DS,SI	; ESTABLISH VECTOR ADDRESSING
	4495	ASSUME DS:ABS0	
F592 C5367C00	4496	LDS SI,EXT_PTR	; GET THE OFFSET OF THE TABLE
F596 8CDA	4497	MOV DX,DS	; GET THE SEGMENT OF THE TABLE
	4498	ASSUME DS:DATA	
F598 1F	4499	POP DS	; RECOVER DATA SEGMENT
F599 52	4500	PUSH DX	; SAVE TABLE SEGMENT ON STACK
	4501		
	4502	;----- DETERMINE GRAPHICS MODE IN OPERATION	
	4503		
F59A	4504	S2:	; DETERMINE_MODE
F59A D1E0	4505	SAL AX,1	; MULTIPLY CODE POINT
F59C D1E0	4506	SAL AX,1	; VALUE BY 8
F59E D1E0	4507	SAL AX,1	
F5A0 03F0	4508	ADD SI,AX	; SI HAS OFFSET OF DESIRED CODES
F5A2 803E490006	4509	CMP CRT_MODE,6	
F5A7 1F	4510	POP DS	; RECOVER TABLE POINTER SEGMENT
F5A8 722C	4511	JC S7	; TEST FOR MEDIUM RESOLUTION MODE
	4512		
	4513	;----- HIGH RESOLUTION MODE	
	4514		
F5AA	4515	S3:	; HIGH_CHAR
F5AA 57	4516	PUSH DI	; SAVE REGEN POINTER
F5AB 56	4517	PUSH SI	; SAVE CODE POINTER
F5AC B604	4518	MOV DH,4	; NUMBER OF TIMES THROUGH LOOP
F5AE	4519	S4:	
F5AE AC	4520	LODSB	; GET BYTE FROM CODE POINTS
F5AF F6C380	4521	TEST BL,80H	; SHOULD WE USE THE FUNCTION
F5B2 7516	4522	JNZ S6	; TO PUT CHAR IN
F5B4 AA	4523	STOSB	; STORE IN REGEN BUFFER
F5B5 AC	4524	LODSB	
F5B6	4525	S5:	
F5B6 268885FF1F	4526	MOV ES:[DI+2000H-1],AL	; STORE IN SECOND HALF
F5B8 83C74F	4527	ADD DI,79	; MOVE TO NEXT ROW IN REGEN
F5BE FECE	4528	DEC DH	; DONE WITH LOOP
F5C0 75EC	4529	JNZ S4	
F5C2 5E	4530	POP SI	
F5C3 5F	4531	POP DI	; RECOVER REGEN POINTER
F5C4 47	4532	INC DI	; POINT TO NEXT CHAR POSITION
F5C5 E2E3	4533	LODP S3	; MORE CHARS TO WRITE
F5C7 E9FBFB	4534	JMP VIDEO_RETURN	
F5CA	4535	S6:	
F5CA 263205	4536	XOR AL,ES:[DI]	; EXCLUSIVE OR WITH CURRENT
F5CD AA	4537	STOSB	; STORE THE CODE POINT
F5CE AC	4538	LODSB	; AGAIN FOR ODD FIELD
F5CF 263285FF1F	4539	XOR AL,ES:[DI+2000H-1]	
F5D4 EBE0	4540	JMP S5	; BACK TO MAINSTREAM
	4541		
	4542	;----- MEDIUM RESOLUTION WRITE	
	4543		
F5D6	4544	S7:	; MED_RES_WRITE
F5D6 8AD3	4545	MOV DL,BL	; SAVE HIGH COLOR BIT
F5D8 D1E7	4546	SAL DI,1	; OFFSET*2 SINCE 2 BYTES/CHAR
F5DA E8D100	4547	CALL S19	; EXPAND BL TO FULL WORD OF COLOR
F5DD	4548	S8:	; MED_CHAR
F5DD 57	4549	PUSH DI	; SAVE REGEN POINTER
F5DE 56	4550	PUSH SI	; SAVE THE CODE POINTER
F5DF B604	4551	MOV DH,4	; NUMBER OF LOOPS
F5E1	4552	S9:	
F5E1 AC	4553	LODSB	; GET CODE POINT
F5E2 E8D000	4554	CALL S21	; DOUBLE UP ALL THE BITS
F5E5 23C3	4555	AND AX,BX	; CONVERT THEM TO FOREGROUND
	4556		; COLOR (0 BACK)
F5E7 F6C280	4557	TEST DL,80H	; IS THIS XOR FUNCTION
F5EA 7407	4558	JZ S10	; NO, STORE IT IN AS IT IS
F5EC 263225	4559	XOR AH,ES:[DI]	; DO FUNCTION WITH HALF
F5EF 26324501	4560	XOR AL,ES:[DI+1]	; AND WITH OTHER HALF
F5F3	4561	S10:	
F5F3 268025	4562	MOV ES:[DI],AH	; STORE FIRST BYTE
F5F6 26804501	4563	MOV ES:[DI+1],AL	; STORE SECOND BYTE
F5FA AC	4564	LODSB	; GET CODE POINT
F5FB E8C500	4565	CALL S21	

LOC OBJ	LINE	SOURCE			
F5FE 23C3	4566	AND	AX,BX		; CONVERT TO COLOR
F600 F6C280	4567	TEST	DL,80H		; AGAIN, IS THIS XOR FUNCTION
F603 740A	4568	JZ	S11		; NO, JUST STORE THE VALUES
F605 2632A50020	4569	XOR	AH,ES:[DI+2000H]		; FUNCTION WITH FIRST HALF
F60A 2632850120	4570	XOR	AL,ES:[DI+2001H]		; AND WITH SECOND HALF
F60F	4571	S11:			
F60F 2688A50020	4572	MOV	ES:[DI+2000H],AH		
F614 2688850120	4573	MOV	ES:[DI+2000H+1],AL		; STORE IN SECOND PORTION OF BUFFER
F619 83C750	4574	ADD	DI,80		; POINT TO NEXT LOCATION
F61C FECE	4575	DEC	DH		
F61E 75C1	4576	JNZ	S9		; KEEP GOING
F620 5E	4577	POP	SI		; RECOVER CODE POINTER
F621 5F	4578	POP	DI		; RECOVER REGEN POINTER
F622 47	4579	INC	DI		; POINT TO NEXT CHAR POSITION
F623 47	4580	INC	DI		
F624 E2B7	4581	LOOP	S8		; MORE TO WRITE
F626 E99CFB	4582	JMP	VIDEO_RETURN		
	4583	GRAPHICS_WRITE	ENDP		
	4584				;-----
	4585	; GRAPHICS_READ	:		
	4586				;-----
F629	4587	GRAPHICS_READ	PROC NEAR		
F629 E8D600	4588	CALL	S26		; CONVERTED TO OFFSET IN REGEN
F62C 8BF0	4589	MOV	SI,AX		; SAVE IN SI
F62E 83EC08	4590	SUB	SP,8		; ALLOCATE SPACE TO SAVE THE
	4591				; READ CODE POINT
F631 8BEC	4592	MOV	BP,SP		; POINTER TO SAVE AREA
	4593				
	4594	;---- DETERMINE GRAPHICS MODES			
	4595				
F633 803E490006	4596	CMP	CRT_MODE,6		
F638 06	4597	PUSH	ES		
F639 1F	4598	POP	DS		; POINT TO REGEN SEGMENT
F63A 721A	4599	JC	S13		; MEDIUM RESOLUTION
	4600				
	4601	;---- HIGH RESOLUTION READ			
	4602				
	4603	;---- GET VALUES FROM REGEN BUFFER AND CONVERT TO CODE POINT			
	4604				
F63C B604	4605	MOV	DH,4		; NUMBER OF PASSES
F63E	4606	S12:			
F63E	4607	MOV	AL,[SI]		; GET FIRST BYTE
F640 884600	4608	MOV	[BP],AL		; SAVE IN STORAGE AREA
F643 45	4609	INC	BP		; NEXT LOCATION
F644 8A840020	4610	MOV	AL,[SI+2000H]		; GET LOWER REGION BYTE
F648 884600	4611	MOV	[BP],AL		; ADJUST AND STORE
F64B 45	4612	INC	BP		
F64C 83C650	4613	ADD	SI,80		; POINTER INTO REGEN
F64F FECE	4614	DEC	DH		; LOOP CONTROL
F651 75EB	4615	JNZ	S12		; DO IT SOME MORE
F653 EB1790	4616	JMP	S15		; GO MATCH THE SAVED CODE POINTS
	4617				
	4618	;---- MEDIUM RESOLUTION READ			
	4619				
F656	4620	S13:			; MED_RES_READ
F656 D1E6	4621	SAL	SI,1		; OFFSET*2 SINCE 2 BYTES/CHAR
F658 B604	4622	MOV	DH,4		; NUMBER OF PASSES
F65A	4623	S14:			
F65A E88800	4624	CALL	S23		; GET PAIR BYTES FROM REGEN
	4625				; INTO SINGLE SAVE
F65D 81C60020	4626	ADD	SI,2000H		; GO TO LOWER REGION
F661 E88100	4627	CALL	S23		; GET THIS PAIR INTO SAVE
F664 81EEB01F	4628	SUB	SI,2000H-80		; ADJUST POINTER BACK INTO UPPER
F668 FECE	4629	DEC	DH		
F66A 75EE	4630	JNZ	S14		; KEEP GOING UNTIL ALL 8 DONE
	4631				
	4632	;---- SAVE AREA HAS CHARACTER IN IT, MATCH IT			
	4633				
F66C	4634	S15:			; FIND_CHAR
F66C BF6EFA90	4635	MOV	DI,OFFSET CRT_CHAR_GEN		; ESTABLISH ADDRESSING
F670 0E	4636	PUSH	CS		
F671 07	4637	POP	ES		; CODE POINTS IN CS
F672 83ED08	4638	SUB	BP,8		; ADJUST POINTER TO BEGINNING
	4639				; OF SAVE AREA
F675 8BF5	4640	MOV	SI,BP		
F677 FC	4641	CLD			; ENSURE DIRECTION
F678 B000	4642	MOV	AL,0		; CURRENT CODE POINT BEING MATCHED

LOC OBJ	LINE	SOURCE
F67A	4643	S16:
F67A 16	4644	PUSH SS ; ESTABLISH ADDRESSING TO STACK
F67B 1F	4645	POP DS ; FOR THE STRING COMPARE
F67C BA8000	4646	MOV DX,128 ; NUMBER TO TEST AGAINST
F67F	4647	S17:
F67F 56	4648	PUSH SI ; SAVE SAVE AREA POINTER
F680 57	4649	PUSH DI ; SAVE CODE POINTER
F681 B90800	4650	MOV CX,8 ; NUMBER OF BYTES TO MATCH
F684 F3	4651	REPE CMPSB ; COMPARE THE 8 BYTES
F685 A6		
F686 5F	4652	POP DI ; RECOVER THE POINTERS
F687 5E	4653	POP SI
F688 741E	4654	JZ S18 ; IF ZERO FLAG SET, THEN MATCH OCCURRED
F68A FEC0	4655	INC AL ; NO MATCH, MOVE ON TO NEXT
F68C 83C708	4656	ADD DI,8 ; NEXT CODE POINT
F68F 4A	4657	DEC DX ; LOOP CONTROL
F690 75ED	4658	JNZ S17 ; DO ALL OF THEM
	4659	
	4660	;----- CHAR NOT MATCHED, MIGHT BE IN USER SUPPLIED SECOND HALF
	4661	
F692 3C00	4662	CMP AL,0 ; AL <= 0 IF ONLY 1ST HALF SCANNED
F694 7412	4663	JE S18 ; IF = 0, THEN ALL HAS BEEN SCANNED
F696 2BC0	4664	SUB AX,AX
F698 8ED8	4665	MOV DS,AX ; ESTABLISH ADDRESSING TO VECTOR
	4666	ASSUME DS:ABS0
F69A C43E7C00	4667	LES DI,EXT_PTR ; GET POINTER
F69E 8CC0	4668	MOV AX,ES ; SEE IF THE POINTER REALLY EXISTS
F6A0 0BC7	4669	OR AX,DI ; IF ALL 0, THEN DOESN'T EXIST
F6A2 7404	4670	JZ S18 ; NO SENSE LOOKING
F6A4 B080	4671	MOV AL,128 ; ORIGIN FOR SECOND HALF
F6A6 EBD2	4672	JMP S16 ; GO BACK AND TRY FOR IT
	4673	ASSUME DS:DATA
	4674	
	4675	;----- CHARACTER IS FOUND (AL=0 IF NOT FOUND)
	4676	
F6A8	4677	S18:
F6A8 83C408	4678	ADD SP,8 ; READJUST THE STACK, THROW AWAY SAVE
F6AB E917FB	4679	JMP VIDEO_RETURN ; ALL DONE
	4680	GRAPHICS_READ ENDP
	4681	;-----
	4682	; EXPAND_MED_COLOR :
	4683	; THIS ROUTINE EXPANDS THE LOW 2 BITS IN BL TO :
	4684	; FILL THE ENTIRE BX REGISTER :
	4685	; ENTRY :
	4686	; BL = COLOR TO BE USED (LOW 2 BITS) :
	4687	; EXIT :
	4688	; BX = COLOR TO BE USED (8 REPLICATIONS OF THE :
	4689	; 2 COLOR BITS) :
	4690	;-----
F6AE	4691	S19 PROC NEAR
F6AE 80E303	4692	AND BL,3 ; ISOLATE THE COLOR BITS
F6B1 8AC3	4693	MOV AL,BL ; COPY TO AL
F6B3 51	4694	PUSH CX ; SAVE REGISTER
F6B4 B90300	4695	MOV CX,3 ; NUMBER OF TIMES TO DO THIS
F6B7	4696	S20:
F6B7 D0E0	4697	SAL AL,1
F6B9 D0E0	4698	SAL AL,1 ; LEFT SHIFT BY 2
F6BB 0AD8	4699	OR BL,AL ; ANOTHER COLOR VERSION INTO BL
F6BD E2F8	4700	LOOP S20 ; FILL ALL OF BL
F6BF 8AFB	4701	MOV BH,BL ; FILL UPPER PORTION
F6C1 59	4702	POP CX ; REGISTER BACK
F6C2 C3	4703	RET ; ALL DONE
	4704	S19 ENDP
	4705	;-----
	4706	; EXPAND_BYTE :
	4707	; THIS ROUTINE TAKES THE BYTE IN AL AND DOUBLES :
	4708	; ALL OF THE BITS, TURNING THE 8 BITS INTO :
	4709	; 16 BITS. THE RESULT IS LEFT IN AX :
	4710	;-----
F6C3	4711	S21 PROC NEAR
F6C3 52	4712	PUSH DX ; SAVE REGISTERS
F6C4 51	4713	PUSH CX
F6C5 53	4714	PUSH BX
F6C6 2BD2	4715	SUB DX,DX ; RESULT REGISTER
F6C8 B90100	4716	MOV CX,1 ; MASK REGISTER
F6CB	4717	S22:
F6CB 8BD8	4718	MOV BX,AX ; BASE INTO TEMP

LOC OBJ	LINE	SOURCE	
F6CD 2309	4719	AND BX,CX	; USE MASK TO EXTRACT A BIT
F6CF 0B03	4720	OR DX,BX	; PUT INTO RESULT REGISTER
F6D1 D1E0	4721	SHL AX,1	
F6D3 D1E1	4722	SHL CX,1	; SHIFT BASE AND MASK BY 1
F6D5 8B08	4723	MOV BX,AX	; BASE TO TEMP
F6D7 23D9	4724	AND BX,CX	; EXTRACT THE SAME BIT
F6D9 0B03	4725	OR DX,BX	; PUT INTO RESULT
F6DB D1E1	4726	SHL CX,1	; SHIFT ONLY MASK NOW,
	4727		; MOVING TO NEXT BASE
F6DD 73EC	4728	JNC S22	; USE MASK BIT COMING OUT TO TERMINATE
F6DF 8BC2	4729	MOV AX,DX	; RESULT TO PARAM REGISTER
F6E1 5B	4730	POP BX	
F6E2 59	4731	POP CX	; RECOVER REGISTERS
F6E3 5A	4732	POP DX	
F6E4 C3	4733	RET	; ALL DONE
	4734	S21 ENDP	
	4735	;-----	
	4736	; MED_READ_BYTE	;
	4737	; THIS ROUTINE WILL TAKE 2 BYTES FROM THE REGEN	;
	4738	; BUFFER, COMPARE AGAINST THE CURRENT FOREGROUND	;
	4739	; COLOR, AND PLACE THE CORRESPONDING ON/OFF BIT	;
	4740	; PATTERN INTO THE CURRENT POSITION IN THE SAVE	;
	4741	; AREA	;
	4742	; ENTRY	;
	4743	; SI,DS = POINTER TO REGEN AREA OF INTEREST	;
	4744	; BX = EXPANDED FOREGROUND COLOR	;
	4745	; BP = POINTER TO SAVE AREA	;
	4746	; EXIT	;
	4747	; BP IS INCREMENT AFTER SAVE	;
	4748	;-----	
F6E5	4749	S23 PROC NEAR	
F6E5 8A24	4750	MOV AH,[SI]	; GET FIRST BYTE
F6E7 8A4401	4751	MOV AL,[SI+1]	; GET SECOND BYTE
F6EA B900C0	4752	MOV CX,0C000H	; 2 BIT MASK TO TEST THE ENTRIES
F6ED B200	4753	MOV DL,0	; RESULT REGISTER
F6EF	4754	S24:	
F6EF 85C1	4755	TEST AX,CX	; IS THIS SECTION BACKGROUND?
F6F1 F8	4756	CLC	; CLEAR CARRY IN HOPES THAT IT IS
F6F2 7401	4757	JZ S25	; IF ZERO, IT IS BACKGROUND
F6F4 F9	4758	STC	; WASN'T, SO SET CARRY
F6F5 D0D2	4759	S25: RCL DL,1	; MOVE THAT BIT INTO THE RESULT
F6F7 D1E9	4760	SHR CX,1	
F6F9 D1E9	4761	SHR CX,1	; MOVE THE MASK TO THE RIGHT BY 2 BITS
F6FB 73F2	4762	JNC S24	; DO IT AGAIN IF MASK DIDN'T FALL OUT
F6FD 885600	4763	MOV [BP],DL	; STORE RESULT IN SAVE AREA
F700 45	4764	INC BP	; ADJUST POINTER
F701 C3	4765	RET	; ALL DONE
	4766	S23 ENDP	
	4767	;-----	
	4768	; V4_POSITION	;
	4769	; THIS ROUTINE TAKES THE CURSOR POSITION	;
	4770	; CONTAINED IN THE MEMORY LOCATION, AND	;
	4771	; CONVERTS IT INTO AN OFFSET INTO THE	;
	4772	; REGEN BUFFER, ASSUMING ONE BYTE/CHAR.	;
	4773	; FOR MEDIUM RESOLUTION GRAPHICS,	;
	4774	; THE NUMBER MUST BE DOUBLED.	;
	4775	; ENTRY	;
	4776	; NO REGISTERS, MEMORY LOCATION	;
	4777	; CURSOR_POSN IS USED	;
	4778	; EXIT	;
	4779	; AX CONTAINS OFFSET INTO REGEN BUFFER	;
	4780	;-----	
F702	4781	S26 PROC NEAR	
F702 A15000	4782	MOV AX,CURSOR_POSN	; GET CURRENT CURSOR
F705	4783	GRAPH_POSN LABEL NEAR	
F705 53	4784	PUSH BX	; SAVE REGISTER
F706 8B08	4785	MOV BX,AX	; SAVE A COPY OF CURRENT CURSOR
F708 8AC4	4786	MOV AL,AH	; GET ROWS TO AL
F70A F6264A00	4787	MUL BYTE PTR CRT_COLS	; MULTIPLY BY BYTES/COLUMN
F70E D1E0	4788	SHL AX,1	; MULTIPLY * 4 SINCE 4 ROWS/BYTE
F710 D1E0	4789	SHL AX,1	
F712 2AFF	4790	SUB BH,BH	; ISOLATE COLUMN VALUE
F714 03C3	4791	ADD AX,BX	; DETERMINE OFFSET
F716 5B	4792	POP BX	; RECOVER POINTER
F717 C3	4793	RET	; ALL DONE
	4794	S26 ENDP	

```

4795 ;-----
4796 ; WRITE_TTY :
4797 ; THIS INTERFACE PROVIDES A TELETYPE LIKE INTERFACE TO THE VIDEO :
4798 ; CARD. THE INPUT CHARACTER IS WRITTEN TO THE CURRENT CURSOR :
4799 ; POSITION, AND THE CURSOR IS MOVED TO THE NEXT POSITION. IF THE :
4800 ; CURSOR LEAVES THE LAST COLUMN OF THE FIELD, THE COLUMN IS SET :
4801 ; TO ZERO, AND THE ROW VALUE IS INCREMENTED. IF THE ROW VALUE :
4802 ; LEAVES THE FIELD, THE CURSOR IS PLACED ON THE LAST ROW, FIRST :
4803 ; COLUMN, AND THE ENTIRE SCREEN IS SCROLLED UP ONE LINE. WHEN :
4804 ; THE SCREEN IS SCROLLED UP, THE ATTRIBUTE FOR FILLING THE NEWLY :
4805 ; BLANKED LINE IS READ FROM THE CURSOR POSITION ON THE PREVIOUS :
4806 ; LINE BEFORE THE SCROLL, IN CHARACTER MODE. IN GRAPHICS MODE, :
4807 ; THE 0 COLOR IS USED. :
4808 ; ENTRY :
4809 ; (AH) = CURRENT CRT MODE :
4810 ; (AL) = CHARACTER TO BE WRITTEN :
4811 ; NOTE THAT BACK SPACE, CAR RET, BELL AND LINE FEED ARE HANDLED :
4812 ; AS COMMANDS RATHER THAN AS DISPLAYABLE GRAPHICS :
4813 ; (BL) = FOREGROUND COLOR FOR CHAR WRITE IF CURRENTLY IN A :
4814 ; GRAPHICS MODE :
4815 ; EXIT :
4816 ; ALL REGISTERS SAVED :
4817 ;-----
4818 ASSUME CS:CODE,DS:DATA
F718 4819 WRITE_TTY PROC NEAR
F718 50 4820 PUSH AX ; SAVE REGISTERS
F719 50 4821 PUSH AX ; SAVE CHAR TO WRITE
F71A B403 4822 MOV AH,3
F71C 8A3E6200 4823 MOV BH,ACTIVE_PAGE ; GET THE CURRENT ACTIVE PAGE
F720 CD10 4824 INT 10H ; READ THE CURRENT CURSOR POSITION
F722 58 4825 POP AX ; RECOVER CHAR
4826
4827 ;----- DX NOW HAS THE CURRENT CURSOR POSITION
4828
F723 3C08 4829 CMP AL,8 ; IS IT A BACKSPACE
F725 7452 4830 JE U8 ; BACK_SPACE
F727 3C0D 4831 CMP AL,0DH ; IS IT CARRIAGE RETURN
F729 7457 4832 JE U9 ; CAR_RET
F72B 3C0A 4833 CMP AL,0AH ; IS IT A LINE FEED
F72D 7457 4834 JE U10 ; LINE_FEED
F72F 3C07 4835 CMP AL,07H ; IS IT A BELL
F731 745A 4836 JE U11 ; BELL
4837
4838 ;----- WRITE THE CHAR TO THE SCREEN
4839
4840
F733 B40A 4841 MOV AH,10 ; WRITE CHAR ONLY
F735 B90100 4842 MOV CX,1 ; ONLY ONE CHAR
F738 CD10 4843 INT 10H ; WRITE THE CHAR
4844
4845 ;----- POSITION THE CURSOR FOR NEXT CHAR
4846
F73A FEC2 4847 INC DL
F73C 3A164A00 4848 CMP DL,BYTE PTR CRT_COLS ; TEST FOR COLUMN OVERFLOW
F740 7533 4849 JNZ U7 ; SET_CURSOR
F742 B200 4850 MOV DL,0 ; COLUMN FOR CURSOR
F744 80FE10 4851 CMP DH,24
F747 752A 4852 JNZ U6 ; SET_CURSOR_INC
4853
4854 ;----- SCROLL REQUIRED
4855
F749 4856 U1:
F749 B402 4857 MOV AH,2
F74B CD10 4858 INT 10H ; SET THE CURSOR
4859
4860 ;----- DETERMINE VALUE TO FILL WITH DURING SCROLL
4861
F74D A04900 4862 MOV AL,CRT_MODE ; GET THE CURRENT MODE
F750 3C04 4863 CMP AL,4
F752 7206 4864 JC U2 ; READ-CURSOR
F754 3C07 4865 CMP AL,7
F756 B700 4866 MOV BH,0 ; FILL WITH BACKGROUND
F758 7506 4867 JNE U3 ; SCROLL-UP
F75A 4868 U2: ; READ-CURSOR
F75A B408 4869 MOV AH,8
F75C CD10 4870 INT 10H ; READ CHAR/ATTR AT CURRENT CURSOR
F75E 8AFC 4871 MOV BH,AH ; STORE IN BH
F760 4872 U3: ; SCROLL-UP

```


LOC OBJ	LINE	SOURCE
F760 B80106	4873	MOV AX,601H ; SCROLL ONE LINE
F763 2BC9	4874	SUB CX,CX ; UPPER LEFT CORNER
F765 B618	4875	MOV DH,24 ; LOWER RIGHT ROW
F767 8A164A00	4876	MOV DL,BYTE PTR CRT_COLS ; LOWER RIGHT COLUMN
F76B FECA	4877	DEC DL
F76D	4878	U4: ; VIDEO-CALL-RETURN
F76D CD10	4879	INT 10H ; SCROLL UP THE SCREEN
F76F	4880	U5: ; TTY-RETURN
F76F 58	4881	POP AX ; RESTORE THE CHARACTER
F770 E952FA	4882	JMP VIDEO_RETURN ; RETURN TO CALLER
F773	4883	U6: ; SET-CURSOR-INC
F773 FECA	4884	INC DH ; NEXT ROW
F775	4885	U7: ; SET-CURSOR
F775 B402	4886	MOV AH,2
F777 EBF4	4887	JMP U4 ; ESTABLISH THE NEW CURSOR
	4888	
	4889	;---- BACK SPACE FOUND
	4890	
F779	4891	U8:
F779 80FA00	4892	CHP DL,0 ; ALREADY AT END OF LINE
F77C 74F7	4893	JE U7 ; SET_CURSOR
F77E FECA	4894	DEC DL ; NO -- JUST MOVE IT BACK
F780 EBF3	4895	JMP U7 ; SET_CURSOR
	4896	
	4897	;---- CARRIAGE RETURN FOUND
	4898	
F782	4899	U9:
F782 B200	4900	MOV DL,0 ; MOVE TO FIRST COLUMN
F784 EBEF	4901	JMP U7 ; SET_CURSOR
	4902	
	4903	;---- LINE FEED FOUND
	4904	
F786	4905	U10:
F786 80FE18	4906	CHP DH,24 ; BOTTOM OF SCREEN
F789 75E8	4907	JNE U6 ; YES, SCROLL THE SCREEN
F78B EBBC	4908	JMP U1 ; NO, JUST SET THE CURSOR
	4909	
	4910	;---- BELL FOUND
	4911	
F78D	4912	U11:
F78D B302	4913	MOV BL,2 ; SET UP COUNT FOR BEEP
F78F E87602	4914	CALL BEEP ; SOUND THE PDD BELL
F792 EBDB	4915	JMP U5 ; TTY_RETURN
	4916	WRITE_TTY ENDP
	4917	;-----
	4918	; LIGHT PEN ;
	4919	; THIS ROUTINE TESTS THE LIGHT PEN SWITCH AND THE LIGHT ;
	4920	; PEN TRIGGER. IF BOTH ARE SET, THE LOCATION OF THE LIGHT ;
	4921	; PEN IS DETERMINED. OTHERWISE, A RETURN WITH NO ;
	4922	; INFORMATION IS MADE. ;
	4923	; ON EXIT ;
	4924	; (AH) = 0 IF NO LIGHT PEN INFORMATION IS AVAILABLE ;
	4925	; BX,CX,DX ARE DESTROYED ;
	4926	; (AH) = 1 IF LIGHT PEN IS AVAILABLE ;
	4927	; (DH,DL) = ROW,COLUMN OF CURRENT LIGHT PEN ;
	4928	; POSITION ;
	4929	; (CH) = RASTER POSITION ;
	4930	; (BX) = BEST GUESS AT PIXEL HORIZONTAL POSITION ;
	4931	;-----
	4932	ASSUME CS:CODE,DS:DATA
	4933	;---- SUBTRACT_TABLE
F794	4934	V1 LABEL BYTE
F794 03	4935	DB 3,3,5,5,3,3,3,4 ;
F795 03		
F796 05		
F797 05		
F798 03		
F799 03		
F79A 03		
F79B 04		
F79C	4936	READ_LPEN PROC NEAR
	4937	
	4938	;---- WAIT FOR LIGHT PEN TO BE DEPRESSED
	4939	
F79C B400	4940	MOV AH,0 ; SET NO LIGHT PEN RETURN CODE
F79E 8B166300	4941	MOV DX,ADDR_6845 ; GET BASE ADDRESS OF 6845
F7A2 83C206	4942	ADD DX,6 ; POINT TO STATUS REGISTER

Appendix A

LOC OBJ	LINE	SOURCE		
F7A5 EC	4943	IN	AL,DX	; GET STATUS REGISTER
F7A6 A804	4944	.TEST	AL,4	; TEST LIGHT PEN SWITCH
F7A8 757E	4945	JNZ	V6	; NOT SET, RETURN
	4946			
	4947	;----- NOW TEST FOR LIGHT PEN TRIGGER		
	4948			
F7AA A802	4949	TEST	AL,2	; TEST LIGHT PEN TRIGGER
F7AC 7503	4950	JNZ	V7A	; RETURN WITHOUT RESETTING TRIGGER
F7AE E98100	4951	JMP	V7	
	4952			
	4953	;----- TRIGGER HAS BEEN SET, READ THE VALUE IN		
	4954			
F7B1	4955	V7A:		
F7B1 B410	4956	MOV	AH,16	; LIGHT PEN REGISTERS ON 6845
	4957			
	4958	;----- INPUT REGS POINTED TO BY AH, AND CONVERT TO ROW COLUMN IN DX		
	4959			
F7B3 8B166300	4960	MOV	DX,ADDR_6845	; ADDRESS REGISTER FOR 6845
F7B7 8AC4	4961	MOV	AL,AH	; REGISTER TO READ
F7B9 EE	4962	OUT	DX,AL	; SET IT UP
F7BA 42	4963	INC	DX	; DATA REGISTER
F7BB EC	4964	IN	AL,DX	; GET THE VALUE
F7BC 8AE8	4965	MOV	CH,AL	; SAVE IN CX
F7BE 4A	4966	DEC	DX	; ADDRESS REGISTER
F7BF FEC4	4967	INC	AH	
F7C1 8AC4	4968	MOV	AL,AH	; SECOND DATA REGISTER
F7C3 EE	4969	OUT	DX,AL	
F7C4 42	4970	INC	DX	; POINT TO DATA REGISTER
F7C5 EC	4971	IN	AL,DX	; GET SECOND DATA VALUE
F7C6 8AE5	4972	MOV	AH,CH	; AX HAS INPUT VALUE
	4973			
	4974	;----- AX HAS THE VALUE READ IN FROM THE 6845		
	4975			
F7C8 8A1E4900	4976	MOV	BL,CRT_MODE	
F7CC 2AFF	4977	SUB	BH,BH	; MODE VALUE TO BX
F7CE 2E8A9F94F7	4978	MOV	BL,CS-VI[BX]	; DETERMINE AMOUNT TO SUBTRACT
F7D3 2BC3	4979	SUB	AX,BX	; TAKE IT AWAY
F7D5 8B1E4E00	4980	MOV	BX,CRT_START	
F7D9 D1EB	4981	SHR	BX,1	
F7DB 2BC3	4982	SUB	AX,BX	
F7DD 7902	4983	JNS	V2	; IF POSITIVE, DETERMINE MODE
F7DF 2BC0	4984	SUB	AX,AX	; <0 PLAYS AS 0
	4985			
	4986	;----- DETERMINE MODE OF OPERATION		
	4987			
F7E1	4988	V2:		; DETERMINE_MODE
F7E1 B103	4989	MOV	CL,3	; SET #8 SHIFT COUNT
F7E3 803E490004	4990	CMP	CRT_MODE,4	; DETERMINE IF GRAPHICS OR ALPHA
F7E8 722A	4991	JB	V4	; ALPHA_PEN
F7EA 803E490007	4992	CMP	CRT_MODE,7	
F7EF 7423	4993	JE	V4	; ALPHA_PEN
	4994			
	4995	;----- GRAPHICS MODE		
	4996			
F7F1 B228	4997	MOV	DL,40	; DIVISOR FOR GRAPHICS
F7F3 F6F2	4998	DIV	DL	; DETERMINE ROW(AL) AND COLUMN(AH)
	4999			; AL RANGE 0-99, AH RANGE 0-39
	5000			
	5001	;----- DETERMINE GRAPHIC ROW POSITION		
	5002			
F7F5 8AE8	5003	MOV	CH,AL	; SAVE ROW VALUE IN CH
F7F7 02ED	5004	ADD	CH,CH	; *2 FOR EVEN/ODD FIELD
F7F9 8ADC	5005	MOV	BL,AH	; COLUMN VALUE TO BX
F7FB 2AFF	5006	SUB	BH,BH	; MULTIPLY BY 8 FOR MEDIUM RES
F7FD 803E490006	5007	CMP	CRT_MODE,6	; DETERMINE MEDIUM OR HIGH RES
F802 7504	5008	JNE	V3	; NOT_HIGH_RES
F804 B104	5009	MOV	CL,4	; SHIFT VALUE FOR HIGH RES
F806 D0E4	5010	SAL	AH,1	; COLUMN VALUE TIMES 2 FOR HIGH RES
F808	5011	V3:		; NOT_HIGH_RES
F808 D3E3	5012	SHL	BX,CL	; MULTIPLY *16 FOR HIGH RES
	5013			
	5014	;----- DETERMINE ALPHA CHAR POSITION		
	5015			
F80A 8AD4	5016	MOV	DL,AH	; COLUMN VALUE FOR RETURN
F80C 8AF0	5017	MOV	DH,AL	; ROW VALUE
F80E D0EE	5018	SHR	DH,1	; DIVIDE BY 4
F810 D0EE	5019	SHR	DH,1	; FOR VALUE IN 0-24 RANGE

```

LOC OBJ          LINE    SOURCE

F812 EB12        5020          JMP     SHORT V5          ; LIGHT_PEN_RETURN_SET
5021
5022          ;---- ALPHA MODE ON LIGHT PEN
5023
F814             5024          V4:          ; ALPHA_PEN
F814 F6364A00    5025          DIV     BYTE PTR CRT_COLS ; DETERMINE ROW,COLUMN VALUE
F818 8AF0        5026          MOV     DH,AL             ; ROWS TO DH
F81A 8AD4        5027          MOV     DL,AH             ; COLS TO DL
F81C D2E0        5028          SAL     AL,CL             ; MULTIPLY ROWS * 8
F81E 8AE8        5029          MOV     CH,AL             ; GET RASTER VALUE TO RETURN REG
F820 8ADC        5030          MOV     BL,AH             ; COLUMN VALUE
F822 32FF        5031          XOR     BH,BH             ; TO BX
F824 D3E3        5032          SAL     BX,CL
F826             5033          V5:          ; LIGHT_PEN_RETURN_SET
F826 B401        5034          MOV     AH,1              ; INDICATE EVERTHING SET
F828             5035          V6:          ; LIGHT_PEN_RETURN
F828 52             5036          PUSH   DX                 ; SAVE RETURN VALUE (IN CASE)
F829 8B166300    5037          MOV     DX,ADDR_6845      ; GET BASE ADDRESS
F82D 83C207      5038          ADD     DX,7               ; POINT TO RESET PARM
F830 EE          5039          OUT     DX,AL             ; ADDRESS, NOT DATA, IS IMPORTANT
F831 5A          5040          POP     DX                 ; RECOVER VALUE
F832             5041          V7:          ; RETURN_NO_RESET
F832 5F             5042          POP     DI
F833 5E             5043          POP     SI
F834 1F             5044          POP     DS                 ; DISCARD SAVED BX,CX,DX
F835 1F             5045          POP     DS
F836 1F             5046          POP     DS
F837 1F             5047          POP     DS
F838 07             5048          POP     ES
F839 CF             5049          IRET
5050          READ_LPEN      ENDP
5051
5052          ;--- INT 12 -----
5053          ; MEMORY_SIZE_DET
5054          ; THIS ROUTINE DETERMINES THE AMOUNT OF MEMORY IN THE SYSTEM
5055          ; AS REPRESENTED BY THE SWITCHES ON THE PLANAR. NOTE THAT THE
5056          ; SYSTEM MAY NOT BE ABLE TO USE I/O MEMORY UNLESS THERE IS A FULL
5057          ; COMPLEMENT OF 64K BYTES ON THE PLANAR.
5058          ; INPUT
5059          ; NO REGISTERS
5060          ; THE MEMORY_SIZE VARIABLE IS SET DURING POWER ON DIAGNOSTICS
5061          ; ACCORDING TO THE FOLLOWING HARDWARE ASSUMPTIONS:
5062          ; PORT 60 BITS 3,2 = 00 - 16K BASE RAM
5063          ; 01 - 32K BASE RAM
5064          ; 10 - 48K BASE RAM
5065          ; 11 - 64K BASE RAM
5066          ; PORT 62 BITS 3-0 INDICATE AMOUNT OF I/O RAM IN 32K INCREMENTS
5067          ; E.G., 0000 - NO RAM IN I/O CHANNEL
5068          ; 0010 - 64K RAM IN I/O CHANNEL, ETC.
5069          ; OUTPUT
5070          ; (AX) = NUMBER OF CONTIGUOUS 1K BLOCKS OF MEMORY
5071          ;-----
5072          ASSUME CS:CODE,DS:DATA
F841             5073          ORG     0F841H
F841             5074          MEMORY_SIZE_DET PROC FAR
F841 FB             5075          STI          ; INTERRUPTS BACK ON
F842 1E             5076          PUSH   DS      ; SAVE SEGMENT
F843 E81302        5077          CALL   DDS
F846 A11300        5078          MOV     AX,MEMORY_SIZE    ; GET VALUE
F849 1F             5079          POP     DS      ; RECOVER SEGMENT
F84A CF             5080          IRET          ; RETURN TO CALLER
5081          MEMORY_SIZE_DET ENDP
5082
5083          ;--- INT 11 -----
5084          ; EQUIPMENT DETERMINATION
5085          ; THIS ROUTINE ATTEMPTS TO DETERMINE WHAT OPTIONAL
5086          ; DEVICES ARE ATTACHED TO THE SYSTEM.
5087          ; INPUT
5088          ; NO REGISTERS
5089          ; THE EQUIP_FLAG VARIABLE IS SET DURING THE POWER ON
5090          ; DIAGNOSTICS USING THE FOLLOWING HARDWARE ASSUMPTIONS:
5091          ; PORT 60 = LOW ORDER BYTE OF EQUIPMENT
5092          ; PORT 3FA = INTERRUPT ID REGISTER OF 8250
5093          ; BITS 7-3 ARE ALWAYS 0
5094          ; PORT 378 = OUTPUT PORT OF PRINTER -- 8255 PORT THAT
5095          ; CAN BE READ AS WELL AS WRITTEN
5096          ; OUTPUT

```

LOC OBJ

LINE SOURCE

```

5097 ; (AX) IS SET, BIT SIGNIFICANT, TO INDICATE ATTACHED I/O :
5098 ; BIT 15,14 = NUMBER OF PRINTERS ATTACHED :
5099 ; BIT 13 NOT USED :
5100 ; BIT 12 = GAME I/O ATTACHED :
5101 ; BIT 11,10,9 = NUMBER OF RS232 CARDS ATTACHED :
5102 ; BIT 8 UNUSED :
5103 ; BIT 7,6 = NUMBER OF DISKETTE DRIVES :
5104 ; 00=1, 01=2, 10=3, 11=4 ONLY IF BIT 0 = 1 :
5105 ; BIT 5,4 = INITIAL VIDEO MODE :
5106 ; 00 - UNUSED :
5107 ; 01 - 40X25 BW USING COLOR CARD :
5108 ; 10 - 80X25 BW USING COLOR CARD :
5109 ; 11 - 80X25 BW USING BW CARD :
5110 ; BIT 3,2 = PLANAR RAM SIZE (00=16K,01=32K,10=48K,11=64K) :
5111 ; BIT 1 NOT USED :
5112 ; BIT 0 = IPL FROM DISKETTE -- THIS BIT INDICATES THAT :
5113 ; THERE ARE DISKETTE DRIVES ON THE SYSTEM :
5114 ;
5115 ; NO OTHER REGISTERS AFFECTED :
5116 ;-----
5117 ; ASSUME CS:CODE,DS:DATA
F84D 5118 ORG 0F84DH
F84D 5119 EQUIPMENT PROC FAR
F84D FB 5120 STI ; INTERRUPTS BACK ON
F84E 1E 5121 PUSH DS ; SAVE SEGMENT REGISTER
F84F E0702 5122 CALL DDS
F852 A11000 5123 MOV AX,EQUIP_FLAG ; GET THE CURRENT SETTINGS
F855 1F 5124 POP DS ; RECOVER SEGMENT
F856 CF 5125 IRET ; RETURN TO CALLER
5126 EQUIPMENT ENDP
5127
5128 ;--- INT 15 -----
5129 ; DUMMY CASSETTE IO ROUTINE--RETURNS 'INVALID CMD' IF THE ROUTINE IS :
5130 ; IS EVER CALLED BY ACCIDENT (AH=86H, CARRY FLAG=1) :
5131 ;-----
F859 5132 ORG 0F859H
F859 5133 CASSETTE_IO PROC FAR
F859 F9 5134 STC ; CARRY INDICATOR=1
F85A B486 5135 MOV AH,86H
F85C CA0200 5136 RET 2
5137 CASSETTE_IO ENDP
5138
5139 ;-----
5140 ; NON-MASKABLE INTERRUPT ROUTINE:
5141 ; THIS ROUTINE WILL PRINT A PARITY CHECK 1 OR 2 MESSAGE :
5142 ; AND ATTEMPT TO FIND THE STORAGE LOCATION CONTAINING THE :
5143 ; BAD PARITY. IF FOUND, THE SEGMENT ADDRESS WILL BE :
5144 ; PRINTED. IF NO PARITY ERROR CAN BE FOUND (INTERMITTANT :
5145 ; READ PROBLEM) ?????<-WILL BE PRINTED WHERE THE ADDRESS :
5146 ; WOULD NORMALLY GO. :
5147 ; IF ADDRESS IN ERROR IS IN THE I/O EXPANSION BOX, THE :
5148 ; ADDRESS WILL BE FOLLOWED BY A '(E)', IF IN SYSTEM UNIT, :
5149 ; A '(S)' WILL FOLLOW THE ADDRESS :
5150 ;-----
F85F 5151 NMI_INT PROC NEAR
5152 ASSUME DS:DATA
F85F 50 5153 PUSH AX ; SAVE ORIG CONTENTS OF AX
F860 E462 5154 IN AL,PORT_C
F862 A8C0 5155 TEST AL,00H ; PARITY CHECK?
F864 7503 5156 JNZ NMI_1
F866 E98700 5157 JMP D14 ; NO, EXIT FROM ROUTINE
F869 5158 NMI_1:
F869 BA4000 5159 MOV DX,DATA
F86C 0EDA 5160 MOV DS,DX
F86E BE15F990 5161 MOV SI,OFFSET D1 ; ADDR OF ERROR MSG
F872 A840 5162 TEST AL,40H ; I/O PARITY CHECK
F874 7504 5163 JNZ D13 ; DISPLAY ERROR MSG
F876 BE25F990 5164 MOV SI,OFFSET D2 ; MUST BE PLANAR
F87A 5165 D13:
F87A B400 5166 MOV AH,0 ; INIT AND SET MODE FOR VIDEO
F87C A04900 5167 MOV AL,CRT_MODE
F87F CD10 5168 INT 10H ; CALL VIDEO_IO PROCEDURE
F881 E04601 5169 CALL P_MSG ; PRINT ERROR MSG
5170
5171 ;----- SEE IF LOCATION THAT CAUSED PARITY CHECK CAN BE FOUND
5172
F884 B000 5173 MOV AL,00H ; DISABLE TRAP

```

LOC OBJ	LINE	SOURCE	
F886 E6A0	5174	OUT	0A0H,AL
F888 E461	5175	IN	AL,PORT_B
F88A 0C30	5176	OR	AL,00110000B ; TOGGLE PARITY CHECK ENABLES
F88C E661	5177	OUT	PORT_B,AL
F88E 24CF	5178	AND	AL,11001111B
F890 E661	5179	OUT	PORT_B,AL
F892 8B1E1300	5180	MOV	BX,HEMORY_SIZE ; GET MEMORY SIZE WORD
F896 FC	5181	CLD	; SET DIR FLAG TO INCRIMENT
F897 2802	5182	SUB	DX,DX ; POINT DX AT START OF MEM
F899	5183		NMI_LOOP:
F899 8EDA	5184	MOV	DS,DX
F89B 8EC2	5185	MOV	ES,DX
F89D B90040	5186	MOV	CX,4000H ; SET FOR 16KB SCAN
F8A0 2BF6	5187	SUB	SI,SI ; SET SI TO BE REALTIME TO
	5188		; START OF ES
F8A2 F3	5189	REP	LODSB ; READ 16KB OF MEMORY
F8A3 AC			
F8A4 E462	5190	IN	AL,PORT_C ; SEE IF PARITY CHECK HAPPENED
F8A6 24C0	5191	AND	AL,11000000B
F8A8 7512	5192	JNZ	PRT_NMI ; GO PRINT ADDRESS IF IT DID
F8AA 81C20004	5193	ADD	DX,0400H ; POINT TO NEXT 16K BLOCK
F8AE 83EB10	5194	SUB	BX,16D
F8B1 75E6	5195	JNZ	NMI_LOOP
F8B3 BE35F990	5196	MOV	SI,(OFFSET D2A) ; PRINT ROW OF ????? IF PARITY
F8B7 E81001	5197	CALL	P_MSG ; CHECK COULD NOT BE RE-CREATED
F8BA FA	5198	CLI	
F8BB F4	5199	HLT	; HALT SYSTEM
F8BC	5200		PRT_NMI:
F8BC 8CDA	5201	MOV	DX,DS
F8BE E81907	5202	CALL	PRT_SEG ; PRINT SEGMENT VALUE
F8C1 BA1302	5203	MOV	DX,0213H
F8C4 B000	5204	MOV	AL,00 ; DISABLE EXPANSION BOX
F8C6 EE	5205	OUT	DX,AL ; (CAN'T WRITE TO MEM)
F8C7 B028	5206	MOV	AL,'C'
F8C9 E80000	5207	CALL	PRT_HEX
F8CC B85AA5	5208	MOV	AX,0A55AH
F8CF 8BC8	5209	MOV	CX,AX
F8D1 2B08	5210	SUB	BX,BX
F8D3 8907	5211	MOV	IBX,AX ; WRITE A WORD TO SEGMENT THAT
F8D5 90	5212	NOP	
F8D6 90	5213	NOP	
F8D7 8B07	5214	MOV	AX,[BX] ; HAD THE ERROR
F8D9 3BC1	5215	CMF	AX,CX ; IS IT THERE?
F8DB 7407	5216	JE	SYS_BOX_ERR ; YES- MUST BE SYS UNIT
F8DD B045	5217	MOV	AL,'E' ; NO-MUST BE IN EXP. BOX
F8DF E8BA00	5218	CALL	PRT_HEX
F8E2 EB05	5219	JMP	SHORT HLT_NMI
F8E4	5220		SYS_BOX_ERR:
F8E4 B053	5221	MOV	AL,'S'
F8E6 E8B300	5222	CALL	PRT_HEX
F8E9	5223		HLT_NMI:
F8E9 B029	5224	MOV	AL,')'
F8EB E8AE00	5225	CALL	PRT_HEX
F8EE FA	5226	CLI	; HALT SYSTEM
F8EF F4	5227	HLT	
F8F0	5228		D14:
F8F0 58	5229	POP	AX ; RESTORE ORIG CONTENTS OF AX
F8F1 CF	5230	IRET	
	5231		NMI_INT ENDP
	5232		
	5233);-----
	5234		; ROS CHECKSUM SUBROUTINE ;
	5235);-----
F8F2	5236	ROS_CHECKSUM PROC NEAR	; NEXT_ROS_MODULE
F8F2 B90020	5237	MOV	CX,8192 ; NUMBER OF BYTES TO ADD
F8F5	5238	ROS_CHECKSUM_CNT:	; ENTRY FOR OPTIONAL ROS TEST
F8F5 32C0	5239	XOR	AL,AL
F8F7	5240		C26:
F8F7 0207	5241	ADD	AL,DS:[BX]
F8F9 43	5242	INC	BX ; POINT TO NEXT BYTE
F8FA E2FB	5243	LOOP	C26 ; ADD ALL BYTES IN ROS MODULE
F8FC 0AC0	5244	OR	AL,AL ; SUM = 0?
F8FE C3	5245	RET	
	5246	ROS_CHECKSUM ENDP	
	5247);-----
	5248		; MESSAGE AREA FOR POST ;
	5249);-----

LOC OBJ	LINE	SOURCE		
F8FF 313031	5250	E0 DB	'101',13,10	; SYSTEM BOARD ERROR
F902 0D				
F903 0A				
F904 20323031	5251	E1 DB	' 201',13,10	; MEMORY ERROR
F908 0D				
F909 0A				
F90A 524F4D	5252	F3A DB	'ROM',13,10	; ROM CHECKSUM ERROR
F90D 0D				
F90E 0A				
F90F 31383031	5253	F3C DB	'1801',13,10	; EXPANSION IO BOX ERROR
F913 0D				
F914 0A				
F915 50415249545920	5254	D1 DB	'PARITY CHECK 2',13,10	
43484543482032				
F923 0D				
F924 0A				
F925 50415249545920	5255	D2 DB	'PARITY CHECK 1',13,10	
43484543482031				
F933 0D				
F934 0A				
F935 3F3F3F3F3F	5256	D2A DB	'?????',13,10	
F93B 0A				
	5257			
	5258			
	5259			
	5260			
	5261			
	5262			
	5263			
F93C				
F93C FB	5264	STI		
F93D 50	5265	PUSH AX		; SAVE AX REG CONTENTS
F93E E461	5266	IN AL,PORT_B		; READ CURRENT VAL OF PORT B
F940 8AE0	5267	MOV AH,AL		
F942 F6D0	5268	NOT AL		; FLIP ALL BITS
F944 2440	5269	AND AL,01000000B		; ISOLATE CONTROL BIT
F946 80E4BF	5270	AND AH,10111111B		; MASK OUT OF ORIGINAL VAL
F949 0AC4	5271	OR AL,AH		; OR NEW CONTROL BIT IN
F94B E661	5272	OUT PORT_B,AL		
F94D B020	5273	MOV AL,E01		
F94F E620	5274	OUT INTA00,AL		
F951 58	5275	POP AX		; RESTORE AX REG
F952 CF	5276	IRET		
	5277	BLINK_INT	ENDP	
	5278			
	5279			
	5280			
	5281			
	5282			
F953	5283			
F953 B84000	5284	MOV AX,DATA		; POINT ES TO DATA AREA
F956 8EC0	5285	MOV ES,AX		
F958 2AE4	5286	SUB AH,AH		; ZERO OUT AH
F95A 8A4702	5287	MOV AL,[BX+2]		; GET LENGTH INDICATOR
F95D B109	5288	MOV CL,09H		; MULTIPLY BY 512
F95F D3E0	5289	SHL AX,CL		
F961 8BC8	5290	MOV CX,AX		; SET COUNT
F963 51	5291	PUSH CX		; SAVE COUNT
F964 B90400	5292	MOV CX,4		; ADJUST
F967 D3E8	5293	SHR AX,CL		
F969 0300	5294	ADD DX,AX		; SET POINTER TO NEXT MODULE
F96B 59	5295	POP CX		; RETRIVE COUNT
F96C E886FF	5296	CALL ROM_CHECKSUM_CNT		; DO CHECKSUM
F96F 7A06	5297	JZ ROM_CHECK_1		
F971 E857E0	5298	CALL ROM_ERR		; POST CHECKSUM ERROR
F974 EB1490	5299	JMP ROM_CHECK_END		; AND EXIT
F977	5300	ROM_CHECK_1:		
F977 52	5301	PUSH DX		; SAVE POINTER
F978 26C70667000300	5302	MOV ES:IO_ROM_INIT,0003H		; LOAD OFFSET
F97F 268C1E6900	5303	MOV ES:IO_ROM_SEG,DS		; LOAD SEGMENT
F984 26FF1E6700	5304	CALL DWORD PTR ES:IO_ROM_INIT		; CALL INIT./TEST ROUTINE
F989 5A	5305	POP DX		
F98A	5306	ROM_CHECK_END:		
F98A C3	5307	RET		; RETURN TO CALLER
	5308	ROM_CHECK	ENDP	
	5309			

```

5310 ;-----
5311 ; CONVERT AND PRINT ASCII CODE          :
5312 ; AL MUST CONTAIN NUMBER TO BE CONVERTED. :
5313 ; AX AND BX DESTROYED.                  :
5314 ;-----
F98B 5315 XPC_BYTE PROC NEAR
F98B 50 5316 PUSH AX ; SAVE FOR LOW NIBBLE DISPLAY
F98C B104 5317 MOV CL,4 ; SHIFT COUNT
F98E D2E8 5318 SHR AL,CL ; NYBBLE SWAP
F990 E80300 5319 CALL XLAT_PR ; DO THE HIGH NIBBLE DISPLAY
F993 58 5320 POP AX ; RECOVER THE NIBBLE
F994 240F 5321 AND AL,0FH ; ISOLATE TO LOW NIBBLE
5322 ; FALL INTO LOW NIBBLE CONVERSION
F996 5323 XLAT_PR PROC NEAR ; CONVERT 00-0F TO ASCII CHARACTER
F996 0490 5324 ADD AL,090H ; ADD FIRST CONVERSION FACTOR
F998 27 5325 DAA ; ADJUST FOR NUMERIC AND ALPHA RANGE
F999 1440 5326 ADC AL,040H ; ADD CONVERSION AND ADJUST LOW NIBBLE
F99B 27 5327 DAA ; ADJUST HIGH NIBBLE TO ASCII RANGE
F99C 5328 PRT_HEX PROC NEAR
F99C B40E 5329 MOV AH,14 ; DISPLAY CHARACTER IN AL
F99E B700 5330 MOV BH,0
F9A0 CD10 5331 INT 10H ; CALL VIDEO_IO
F9A2 C3 5332 RET
5333 PRT_HEX ENDP
5334 XLAT_PR ENDP
5335 XPC_BYTE ENDP
5336
F9A3 5337 F4 LABEL WORD ; PRINTER SOURCE TABLE
F9A3 BC03 5338 DW 3BCH
F9A5 7803 5339 DW 378H
F9A7 7802 5340 DW 278H
F9A9 5341 F4E LABEL WORD
5342
5343 ;-----
5344 ; THIS SUBROUTINE WILL PRINT A MESSAGE ON THE DISPLAY :
5345 ; :
5346 ; ENTRY REQUIREMENTS: :
5347 ; SI = OFFSET(ADDRESS) OF MESSAGE BUFFER :
5348 ; CX = MESSAGE BYTE COUNT :
5349 ; MAXIMUM MESSAGE LENGTH IS 36 CHARACTERS :
5350 ;-----
F9A9 5351 E_MSG PROC NEAR
F9A9 8BEE 5352 MOV BP,SI ; SET BP NON-ZERO TO FLAG ERR
F9AB E81C00 5353 CALL P_MSG ; PRINT MESSAGE
F9AE 1E 5354 PUSH DS
F9AF E8A700 5355 CALL DDS
F9B2 A01000 5356 MOV AL,BYTE PTR EQUIP_FLAG ; LOOP/HALT ON ERROR
F9B5 2401 5357 AND AL,01H ; SWITCH ON?
F9B7 750F 5358 JNZ G12 ; NO - RETURN
F9B9 5359 MFG_HALT:
F9B9 FA 5360 CLI ; YES - HALT SYSTEM
F9BA B089 5361 MOV AL,89H
F9BC E663 5362 OUT CHD_PORT,AL
F9BE B085 5363 MOV AL,10000101B ; DISABLE KB
F9C0 E661 5364 OUT PORT_B,AL
F9C2 A01500 5365 MOV AL,MFG_ERR_FLAG ; RECOVER ERROR INDICATOR
F9C5 E660 5366 OUT PORT_A,AL ; SET INTO 8255 REG
F9C7 F4 5367 HLT ; HALT SYS
F9C8 5368 G12:
F9CB 1F 5369 POP DS ; WRITE_MSG:
F9C9 C3 5370 RET
5371 E_MSG ENDP
5372
F9CA 5373 P_MSG PROC NEAR
F9CA 5374 G12A:
F9CA 2E8A04 5375 MOV AL,CS:[SI] ; PUT CHAR IN AL
F9CD 46 5376 INC SI ; POINT TO NEXT CHAR
F9CE 50 5377 PUSH AX ; SAVE PRINT CHAR
F9CF E8CAFF 5378 CALL PRT_HEX ; CALL VIDEO_IO
F9D2 58 5379 POP AX ; RECOVER PRINT CHAR
F9D3 3C0A 5380 CMP AL,10 ; HAS IT LINE FEED?
F9D5 75F3 5381 JNE G12A ; NO,KEEP PRINTING STRING
F9D7 C3 5382 RET
5383 P_MSG ENDP
5384
5385 ;-----
5386 ; INITIAL RELIABILITY TEST -- SUBROUTINES :
5387 ;-----
5388 ASSUME CS:CODE,DS:DATA

```

LOC OBJ

LINE SOURCE

```

5389 ;-----
5390 ; SUBROUTINES FOR POWER ON DIAGNOSTICS :
5391 ;-----
5392 ; THIS PROCEDURE WILL ISSUE ONE LONG TONE (3 SECS) AND ONE OR :
5393 ; MORE SHORT TONES (1 SEC) TO INDICATE A FAILURE ON THE PLANAR :
5394 ; BOARD, A BAD RAM MODULE, OR A PROBLEM WITH THE CRT. :
5395 ; ENTRY PARAMETERS: :
5396 ; DH = NUMBER OF LONG TONES TO BEEP :
5397 ; DL = NUMBER OF SHORT TONES TO BEEP. :
5398 ;-----
F908 5399 ERR_BEEP PROC NEAR
F908 9C 5400 PUSHF ; SAVE FLAGS
F909 FA 5401 CLI ; DISABLE SYSTEM INTERRUPTS
F90A 1E 5402 PUSH DS ; SAVE DS REG CONTENTS
F90B E87B00 5403 CALL DDS
F90E 0AF6 5404 OR DH,DH ; ANY LONG ONES TO BEEP
F90E 7414 5405 JZ G3 ; NO, DO THE SHORT ONES
F9E2 5406 G1: ; LONG_BEEP:
F9E2 B306 5407 MOV BL,6 ; COUNTER FOR BEEPS
F9E4 E82100 5408 CALL BEEP ; DO THE BEEP
F9E7 5409 G2:
F9E7 E2FE 5410 LOOP G2 ; DELAY BETWEEN BEEPS
F9E9 FECE 5411 DEC DH ; ANY MORE TO DO
F9EB 75F5 5412 JNZ G1 ; DO IT
F9ED 803E120001 5413 CMP HFG_TST,1 ; HFG TEST MODE?
F9F2 7502 5414 JNE G3 ; YES - CONTINUE BEEPING SPEAKER
F9F4 EBC3 5415 JMP HFG_HALT ; STOP BLINKING LED
F9F6 5416 G3: ; SHORT_BEEP:
F9F6 B301 5417 MOV BL,1 ; COUNTER FOR A SHORT BEEP
F9F8 E80D00 5418 CALL BEEP ; DO THE SOUND
F9FB 5419 G4:
F9FB E2FE 5420 LOOP G4 ; DELAY BETWEEN BEEPS
F9FD FECA 5421 DEC DL ; DONE WITH SHORTS
F9FF 75F5 5422 JNZ G3 ; DO SOME MORE
FA01 5423 G5:
FA01 E2FE 5424 LOOP G5 ; LONG DELAY BEFORE RETURN
FA03 5425 G6:
FA03 E2FE 5426 LOOP G6
FA05 1F 5427 POP DS ; RESTORE ORIG CONTENTS OF DS
FA06 90 5428 POPF ; RESTORE FLAGS TO ORIG SETTINGS
FA07 C3 5429 RET ; RETURN TO CALLER
5430 ERR_BEEP ENDP
5431
5432 ;---- ROUTINE TO SOUND BEEPER
5433
FA08 5434 BEEP PROC NEAR
FA08 B0B6 5435 MOV AL,10110110B ; SEL TIM 2,LSB,MSB,BINARY
FA0A E643 5436 OUT TIMER+3,AL ; WRITE THE TIMER MODE REG
FA0C B83305 5437 MOV AX,533H ; DIVISOR FOR 1000 HZ
FA0F E642 5438 OUT TIMER+2,AL ; WRITE TIMER 2 CNT - LSB
FA11 8AC4 5439 MOV AL,AH
FA13 E642 5440 OUT TIMER+2,AL ; WRITE TIMER 2 CNT - MSB
FA15 E461 5441 IN AL,PORT_B ; GET CURRENT SETTING OF PORT
FA17 8AE0 5442 MOV AH,AL ; SAVE THAT SETTING
FA19 0C03 5443 OR AL,03 ; TURN SPEAKER ON
FA1B E661 5444 OUT PORT_B,AL
FA1D 2BC9 5445 SUB CX,CX ; SET CNT TO WAIT 500 MS
FA1F 5446 G7:
FA1F E2FE 5447 LOOP G7 ; DELAY BEFORE TURNING OFF
FA21 FECC 5448 DEC BL ; DELAY CNT EXPIRED?
FA23 75FA 5449 JNZ G7 ; NO - CONTINUE BEEPING SPK
FA25 8AC4 5450 MOV AL,AH ; RECOVER VALUE OF PORT
FA27 E661 5451 OUT PORT_B,AL
FA29 C3 5452 RET ; RETURN TO CALLER
5453 BEEP ENDP
5454
5455 ;-----
5456 ; THIS PROCEDURE WILL SEND A SOFTWARE RESET TO THE KEYBOARD. :
5457 ; SCAN CODE 'AA' SHOULD BE RETURNED TO THE CPU. :
5458 ;-----
FA2A 5459 KBD_RESET PROC NEAR
FA2A B008 5460 ASSUME DS:ABS0
FA2C E661 5461 MOV AL,06H ; SET KBD CLK LINE LOW
FA2E B95629 5462 OUT PORT_B,AL ; WRITE 8255 PORT B
FA31 5463 MOV CX,10582 ; HOLD KBD CLK LOW FOR 20 MS
FA31 E2FE 5464 G8:
FA31 E2FE 5465 LOOP G8 ; LOOP FOR 20 MS

```


LOC OBJ	LINE	SOURCE
FA33 B0C8	5466	MOV AL,0C0H ; SET CLK, ENABLE LINES HIGH
FA35 E661	5467	OUT PORT_B,AL
FA37	5468	SP_TEST: ; ENTRY FOR MANUFACTURING TEST 2
FA37 B048	5469	MOV AL,48H ; SET KBD CLK HIGH, ENABLE LOW
FA39 E661	5470	OUT PORT_B,AL
FA3B B0FD	5471	MOV AL,0FDH ; ENABLE KEYBOARD INTERRUPTS
FA3D E621	5472	OUT INTA01,AL ; WRITE 8259 IHR
FA3F C6066B0400	5473	MOV DATA_AREA[OFFSET INTR_FLAG] ; RESET INTERRUPT INDICATOR
FA44 FB	5474	STI ; ENABLE INTERRUPTS
FA45 2BC9	5475	SUB CX,CX ; SETUP INTERRUPT TIMEOUT CNT
FA47	5476	G9:
FA47 F6066B0402	5477	JNZ DATA_AREA[OFFSET INT_R_FLAG],02H ; DID A KEYBOARD INTR OCCUR?
FA48 7502	5478	TEST JNZ ; YES - READ SCAN CODE RETURNED
FA4E E2F7	5479	LOOP G9 ; NO - LOOP TILL TIMEOUT
FA50	5480	G10:
FA50 E460	5481	IN AL,PORT_A ; READ KEYBOARD SCAN CODE
FA52 8AD8	5482	MOV BL,AL ; SAVE SCAN CODE JUST READ
FA54 B0C8	5483	MOV AL,0C8H ; CLEAR KEYBOARD
FA56 E661	5484	OUT PORT_B,AL
FA58 C3	5485	KBD_RET ; RETURN TO CALLER
	5486	KBD_RESET ENDP
	5487	
FA59	5488	DDS PROC NEAR
FA59 50	5489	PUSH AX ; SAVE AX
FA5A B84000	5490	MOV AX,DATA
FA5D 8ED8	5491	MOV DS,AX ; SET SEGMENT
FA5F 58	5492	POP AX ; RESTORE AX
FA60 C3	5493	RET
	5494	DDS ENDP
	5495	
	5496	
	5497	;------
	5498	; CHARACTER GENERATOR GRAPHICS FOR 320X200 AND 640X200 GRAPHICS ;
	5499	;------
FA6E	5499	ORG 0F46EH
FA6E	5500	CRT_CHAR_BYTE LABEL BYTE
FA6E 0000000000000000	5501	DB 000H,000H,000H,000H,000H,000H,000H,000H ; D_00
FA76 7E81A501BD99817E	5502	DB 07EH,081H,0A5H,081H,08DH,099H,081H,07EH ; D_01
FA7E 7EFFDFBFC3E7FF7E	5503	DB 07EH,0FFH,0DBH,0FFH,0C3H,0E7H,0FFH,0E0H ; D_02
FA86 6CFEFEF7C381000	5504	DB 06CH,0FEH,0FEH,0FEH,07CH,038H,010H,000H ; D_03
FA8E 10387CFE7C381000	5505	DB 010H,038H,07CH,0FEH,07CH,038H,010H,000H ; D_04
FA96 387C38FE7C387C	5506	DB 038H,07CH,038H,0FEH,0FEH,07CH,038H,07CH ; D_05
FA9E 1010387CFE7C387C	5507	DB 010H,010H,038H,07CH,0FEH,07CH,038H,07CH ; D_06
FAA6 000183C3CE180000	5508	DB 000H,000H,018H,03CH,03CH,018H,000H,000H ; D_07
FAAE FFFFE7C3CE7FFFFF	5509	DB 0FFH,0FFH,0E7H,0C3H,0C3H,0E7H,0FFH,0FFH ; D_08
FAB6 003C664242663C00	5510	DB 000H,03CH,066H,042H,042H,066H,03CH,000H ; D_09
FABE FFC99BBD099C3FF	5511	DB 0FFH,0C3H,099H,08DH,08DH,099H,0C3H,0FFH ; D_0A
FAC6 0F070F7DC0CC0C78	5512	DB 00FH,007H,00FH,070H,0CCH,0CCH,0CCH,078H ; D_0B
FACE 3C6666663187E18	5513	DB 03CH,066H,066H,066H,03CH,018H,07EH,018H ; D_0C
FAD6 3F333F303070F0E0	5514	DB 03FH,033H,03FH,030H,030H,070H,0F0H,0E0H ; D_0D
FADE 7F637F63637E6C0	5515	DB 07FH,063H,07FH,063H,063H,067H,0E6H,0C0H ; D_0E
FAE6 995A3CE7E73C5A99	5516	DB 099H,05AH,03CH,0E7H,0E7H,03CH,05AH,099H ; D_0F
FAEE 80E0F8FEF8E08000	5517	DB 080H,0E0H,0F8H,0FEH,0F8H,0E0H,080H,000H ; D_10
FAF6 0203FEF3E0E02000	5518	DB 020H,0DEH,03EH,0FEH,03EH,0E0H,002H,000H ; D_11
FAFE 183C7E18187E3C18	5519	DB 018H,03CH,07EH,018H,018H,07EH,03CH,018H ; D_12
FB06 6666666666606600	5520	DB 066H,066H,066H,066H,066H,000H,066H,000H ; D_13
FB0E 7F0B0B7B1B1B1B00	5521	DB 07FH,0DBH,0DBH,07BH,01BH,01BH,01BH,000H ; D_14
FB16 3E6336C6C638CC78	5522	DB 03EH,063H,038H,06CH,06CH,038H,0CCH,078H ; D_15
FB1E 000000007E7E7E00	5523	DB 000H,000H,000H,000H,07EH,07EH,07EH,000H ; D_16
FB26 183C7E187E3C18FF	5524	DB 018H,03CH,07EH,018H,07EH,03CH,018H,0FFH ; D_17
FB2E 183C7E1818181800	5525	DB 018H,03CH,07EH,018H,018H,018H,018H,000H ; D_18
FB36 181818187E3C1800	5526	DB 018H,018H,018H,018H,07EH,03CH,018H,000H ; D_19
FB3E 00180CFE0C180000	5527	DB 000H,018H,00CH,0FEH,00CH,018H,000H,000H ; D_1A
FB46 00306FE603000000	5528	DB 000H,030H,060H,0FEH,060H,030H,000H,000H ; D_1B
FB4E 00000C0C0FE00000	5529	DB 000H,000H,0C0H,0C0H,0C0H,0FEH,000H,000H ; D_1C
FB56 00246FF662400000	5530	DB 000H,024H,066H,0FFH,066H,024H,000H,000H ; D_1D
FB5E 00183CEFFFF00000	5531	DB 000H,018H,03CH,07EH,0FFH,0FFH,000H,000H ; D_1E
FB66 00FFFF7E3C180000	5532	DB 000H,0FFH,0FFH,07EH,03CH,018H,000H,000H ; D_1F
FB6E 0000000000000000	5533	DB 000H,000H,000H,000H,000H,000H,000H ; SP_D_20
FB76 3078787830003000	5534	DB 030H,078H,078H,030H,030H,000H,030H,000H ; ! D_21
FB7E 6C6C6C0000000000	5535	DB 06CH,06CH,06CH,000H,000H,000H,000H,000H ; " D_22
FB86 6C6CFE6CFE6C6C00	5536	DB 06CH,06CH,0FEH,06CH,0FEH,06CH,06CH,000H ; # D_23
FB8E 307CC780CF8F3000	5537	DB 030H,07CH,0C0H,078H,00CH,0F8H,030H,000H ; \$ D_24
FB96 00C66C183066C600	5538	DB 000H,0C6H,0CCH,018H,030H,066H,0C6H,000H ; PER CENT D_25
FB9E 386C3878DC0CC7600	5539	DB 038H,06CH,038H,076H,0DCH,0CCH,076H,000H ; & D_26
FBAA 6060C00000000000	5540	DB 060H,060H,0C0H,000H,000H,000H,000H,000H ; ' D_27
FBAE 1830606060301800	5541	DB 018H,030H,060H,060H,060H,030H,018H,000H ; (D_28
FBBA 6030181818306000	5542	DB 060H,030H,018H,018H,018H,030H,060H,000H ;) D_29

LOC OBJ	LINE	SOURCE
FBBE 00663CFF3C660000	5543	DB 000H,066H,03CH,0FFH,03CH,066H,000H,000H ; # D_2A
FBCE 003030FC30300000	5544	DB 000H,030H,030H,0FCH,030H,030H,000H,000H ; + D_2B
FBCE 0000000000303060	5545	DB 000H,000H,000H,000H,000H,030H,030H,060H ; D_2C
FBD6 000000FC00000000	5546	DB 000H,000H,000H,0FCH,000H,000H,000H,000H ; - D_2D
FBD6 0000000000303000	5547	DB 000H,000H,000H,000H,000H,000H,030H,030H,000H ; - D_2E
FBE6 060C183060C08000	5548	DB 060H,00CH,018H,030H,060H,0C0H,080H,000H ; / D_2F
FBE6 7CC6CEDEF6E67C00	5549	DB 07CH,0C6H,0CEH,0DEH,0F6H,0E6H,07CH,000H ; 0 D_30
FBF6 3070303030709FC0	5550	DB 030H,070H,030H,030H,030H,030H,0FCH,000H ; 1 D_31
FBFE 78CC0C3860CCFC00	5551	DB 078H,0CCH,00CH,038H,060H,0CCH,0FCH,000H ; 2 D_32
FC06 78CC0C380CC78000	5552	DB 078H,0CCH,00CH,038H,00CH,0CCH,0CCH,078H,000H ; 3 D_33
FC0E 1C3C6CCFC0E1C000	5553	DB 01CH,03CH,06CH,0CCH,0FEH,0CCH,01EH,000H ; 4 D_34
FC16 FCC0F800CC0C7800	5554	DB 0FCH,0C0H,0F8H,00CH,00CH,0CCH,078H,000H ; 5 D_35
FC1E 3860CF8C8CC78000	5555	DB 038H,060H,0C0H,0F8H,0CCH,0CCH,078H,000H ; 6 D_36
FC26 FCC0C18303030000	5556	DB 0FCH,0CCH,00CH,018H,030H,030H,030H,000H ; 7 D_37
FC2E 78CC0C78CC780000	5557	DB 078H,0CCH,0CCH,078H,0CCH,0CCH,078H,000H ; 8 D_38
FC36 78CC0C7C0C187000	5558	DB 078H,0CCH,0CCH,07CH,0CCH,018H,070H,000H ; 9 D_39
FC3E 0030300000303000	5559	DB 000H,030H,030H,000H,000H,030H,030H,000H ; : D_3A
FC46 0030300000303060	5560	DB 000H,030H,030H,000H,000H,030H,030H,060H ; ; D_3B
FC4E 183060C060301800	5561	DB 018H,030H,060H,0C0H,060H,000H,018H,000H ; < D_3C
FC56 0000FC0000FC0000	5562	DB 000H,000H,0FCH,000H,000H,0FCH,000H,000H ; = D_3D
FC5E 6030180C18306000	5563	DB 060H,030H,018H,00CH,018H,030H,000H,000H ; > D_3E
FC66 78CC0C1830003000	5564	DB 078H,0CCH,00CH,018H,030H,000H,030H,000H ; ? D_3F
FC6E 7CC6DEDEEC078000	5565	DB 07CH,0C6H,0DEH,0DEH,0DEH,0C0H,078H,000H ; @ D_40
FC76 3078CCFC0CC00000	5566	DB 030H,078H,0CCH,0CCH,0FCH,0CCH,0CCH,000H ; A D_41
FC7E FC66667C6666FC00	5567	DB 0FCH,066H,066H,07CH,066H,066H,066H,0FCH,000H ; B D_42
FC86 3C66C0C0663C0000	5568	DB 03CH,066H,0C0H,0C0H,0C0H,066H,03CH,000H ; C D_43
FC8E F86C6666666C0000	5569	DB 0F8H,06CH,066H,066H,066H,06CH,0F8H,000H ; D D_44
FC96 FE268786862FE000	5570	DB 0FEH,062H,068H,078H,068H,062H,0FEH,000H ; E D_45
FC9E FE268786860F0000	5571	DB 0FEH,062H,068H,078H,068H,060H,0F0H,000H ; F D_46
FCAA 3C66C0C0663E0000	5572	DB 03CH,066H,0C0H,0C0H,0C0H,066H,03EH,000H ; G D_47
FCBA CCCCCFC000000000	5573	DB 0CCH,0CCH,0CCH,0FCH,0CCH,0CCH,0CCH,000H ; H D_48
FCCE 7830303030307800	5574	DB 078H,030H,030H,030H,030H,030H,078H,000H ; I D_49
FCBE 1E0C0C0C0C0C7800	5575	DB 01EH,00CH,00CH,00CH,0CCH,0CCH,078H,000H ; J D_4A
FCC6 E66666786C66E600	5576	DB 0E6H,066H,06CH,078H,06CH,066H,0E6H,000H ; K D_4B
FCC6 F0606060626FE000	5577	DB 0F0H,060H,060H,060H,062H,066H,0FEH,000H ; L D_4C
FCD6 C6EEFED666C60000	5578	DB 0C6H,0EEH,0FEH,0FEH,0D6H,0C6H,0C6H,000H ; M D_4D
FCD6 C6E6F6DE0CC6C600	5579	DB 0C6H,0E6H,0F6H,0DEH,0C6H,0C6H,0C6H,000H ; N D_4E
FCE6 3B6CC6C6C66C3800	5580	DB 038H,06CH,0C6H,0C6H,0C6H,06CH,038H,000H ; O D_4F
FCEE FC66667C6060F000	5581	DB 0FCH,066H,066H,07CH,060H,060H,0F0H,000H ; P D_50
FCF6 78CCCCC0C781C000	5582	DB 078H,0CCH,0CCH,0CCH,00CH,078H,01CH,000H ; Q D_51
FCFE FC66667C6C66E600	5583	DB 0FCH,066H,066H,07CH,06CH,066H,0E6H,000H ; R D_52
FD06 78CC0701C0C78000	5584	DB 078H,0CCH,0E0H,070H,01CH,0CCH,078H,000H ; S D_53
FD0E FCB4303030307800	5585	DB 0FCH,0B4H,030H,030H,030H,030H,078H,000H ; T D_54
FD16 CCCCCCCCC0000000	5586	DB 0CCH,0CCH,0CCH,0CCH,0CCH,0CCH,0FCH,000H ; U D_55
FD1E CCCCCCCCC0783000	5587	DB 0CCH,0CCH,0CCH,0CCH,0CCH,078H,030H,000H ; V D_56
FD26 C6C6C066FE000000	5588	DB 0C6H,0C6H,0C6H,0D6H,0FEH,0EEH,0C6H,000H ; W D_57
FD2E C6C6C0383B6CC600	5589	DB 0C6H,0C6H,06CH,038H,038H,06CH,0C6H,000H ; X D_58
FD36 CCCCC07830307800	5590	DB 0CCH,0CCH,0CCH,078H,030H,030H,078H,000H ; Y D_59
FD3E FEC68C18326FE000	5591	DB 0FEH,0C6H,08CH,018H,032H,066H,0FEH,000H ; Z D_5A
FD46 7860606060607800	5592	DB 078H,060H,060H,060H,060H,060H,078H,000H ; [D_5B
FD4E C06030180C060200	5593	DB 0C0H,060H,030H,018H,00CH,066H,002H,000H ; BACKSLASH D_5C
FD56 7818181818187800	5594	DB 078H,018H,018H,018H,018H,018H,078H,000H ;] D_5D
FD5E 10386C6000000000	5595	DB 010H,038H,06CH,0C6H,000H,000H,000H,000H ; CIRCUMFLEX D_5E
FD66 00000000000000FF	5596	DB 000H,000H,000H,000H,000H,000H,000H,0FFH ; _ D_5F
FD6E 3030180000000000	5597	DB 030H,030H,018H,000H,000H,000H,000H,000H ; ` D_60
FD76 0000780C7CC7E000	5598	DB 000H,000H,078H,00CH,07CH,0CCH,076H,000H ; LOWER CASE A D_61
FD7E E0660607C6666C00	5599	DB 0E0H,060H,060H,07CH,066H,066H,0CCH,000H ; L.C. B D_62
FD86 000078CC0CC78000	5600	DB 000H,000H,078H,0CCH,0C0H,0CCH,078H,000H ; L.C. C D_63
FD8E 1C0C0C07CC0C7E00	5601	DB 01CH,00CH,00CH,07CH,0CCH,0CCH,076H,000H ; L.C. D D_64
FD96 000078CFC0078000	5602	DB 000H,000H,078H,0CCH,0FCH,0C0H,078H,000H ; L.C. E D_65
FD9E 386C0F06060F0000	5603	DB 038H,06CH,060H,0F0H,060H,060H,0F0H,000H ; L.C. F D_66
FDA6 000076CCCC7C0C78	5604	DB 000H,000H,076H,0CCH,0CCH,07CH,0CCH,0F8H ; L.C. G D_67
FDAE E0606C786666E600	5605	DB 0E0H,060H,06CH,076H,066H,066H,0E6H,000H ; L.C. H D_68
FDB6 3000703030307800	5606	DB 030H,000H,070H,030H,030H,030H,078H,000H ; L.C. I D_69
FDBE 0C0B0C0C0C0C0C78	5607	DB 0CCH,000H,0CCH,00CH,00CH,0CCH,0CCH,078H ; L.C. J D_6A
FDC6 E060666C786CE000	5608	DB 0E0H,060H,066H,06CH,078H,06CH,0E6H,000H ; L.C. K D_6B
FDD6 7030303030307800	5609	DB 070H,030H,030H,030H,030H,030H,078H,000H ; L.C. L D_6C
FDD6 0000CFE0E66C6000	5610	DB 000H,000H,0CCH,0F0H,0FEH,0D6H,0C6H,000H ; L.C. M D_6D
FDD6 0000F8CCCC000000	5611	DB 000H,000H,0F8H,0CCH,0CCH,0CCH,0CCH,000H ; L.C. N D_6E
FDE6 000078CCCC0C7800	5612	DB 000H,000H,078H,0CCH,0CCH,0CCH,078H,000H ; L.C. O D_6F
FDEF 0000D66667C0F000	5613	DB 000H,000H,0DCH,066H,066H,07CH,060H,0F0H ; L.C. P D_70
FDF6 000076CCCC7C0C1E	5614	DB 000H,000H,076H,0CCH,0CCH,07CH,0CCH,01EH ; L.C. Q D_71
FDFE 0000D666660F0000	5615	DB 000H,000H,0DCH,076H,066H,060H,0F8H,000H ; L.C. R D_72
FDE6 00007C0780CF8000	5616	DB 000H,000H,07CH,0C0H,078H,00CH,0F8H,000H ; L.C. S D_73
FE0E 10307C3030341800	5617	DB 010H,030H,07CH,030H,030H,034H,018H,000H ; L.C. T D_74
FE16 0000CCCC0C0C7E00	5618	DB 000H,000H,0CCH,0CCH,0CCH,0CCH,076H,000H ; L.C. U D_75
FE1E 0000CCCC0C780000	5619	DB 000H,000H,0CCH,0CCH,0CCH,0CCH,078H,030H,000H ; L.C. V D_76

```

FE26 0000C6D6FEFE6C00 5620 DB 000H,000H,0C6H,0D6H,0FEH,0FEH,06CH,000H ; L.C. M_D_77
FE2E 0000C6C6386CC600 5621 DB 000H,000H,0C6H,06CH,038H,06CH,0C6H,000H ; L.C. X_D_78
FE36 0000CCCC7C0CF8 5622 DB 000H,000H,0CCH,0CCH,0CCH,07CH,0CCH,0FBH ; L.C. Y_D_79
FE3E 0000FC9B3064FC00 5623 DB 000H,000H,0FCH,098H,030H,064H,0FCH,000H ; L.C. Z_D_7A
FE46 1C3030E030301C00 5624 DB 01CH,030H,030H,0E0H,030H,030H,01CH,000H ; [ D_7B
FE4E 1818180018181800 5625 DB 018H,018H,018H,000H,018H,018H,018H,000H ; | D_7C
FE56 E030301C3030E000 5626 DB 0E0H,030H,030H,01CH,030H,030H,0E0H,000H ; ] D_7D
FE5E 760C000000000000 5627 DB 076H,0DCH,000H,000H,000H,000H,000H,000H ; TILDE D_7E
FE66 0010386CC6C6FE00 5628 DB 000H,010H,038H,06CH,0C6H,0C6H,0FEH,000H ; DELTA D_7F
5629
5630 ;--- INT 1A -----
5631 ; TIME_OF_DAY :
5632 ; THIS ROUTINE ALLOWS THE CLOCK TO BE SET/READ :
5633 ; :
5634 ; INPUT :
5635 ; (AH) = 0 READ THE CURRENT CLOCK SETTING :
5636 ; RETURNS CX = HIGH PORTION OF COUNT :
5637 ; DX = LOW PORTION OF COUNT :
5638 ; AL = 0 IF TIMER HAS NOT PASSED :
5639 ; 24 HOURS SINCE LAST READ :
5640 ; <>0 IF ON ANOTHER DAY :
5641 ; (AH) = 1 SET THE CURRENT CLOCK :
5642 ; CX = HIGH PORTION OF COUNT :
5643 ; DX = LOW PORTION OF COUNT :
5644 ; NOTE: COUNTS OCCUR AT THE RATE OF :
5645 ; 1193180/65536 COUNTS/SEC :
5646 ; (OR ABOUT 18.2 PER SECOND -- SEE EQUATES BELOW) :
5647 ;-----
5648 ASSUME CS:CODE,DS:DATA
FE6E 5649 ORG 0FE5EH
FE6E 5650 TIME_OF_DAY PROC FAR
FE6E FB 5651 STI ; INTERRUPTS BACK ON
FE6F 1E 5652 PUSH DS ; SAVE SEGMENT
FE70 E8E6FB 5653 CALL DDS
FE73 0AE4 5654 OR AH,AH ; AH=0
FE75 7407 5655 JZ T2 ; READ_TIME
FE77 FECC 5656 DEC AH ; AH=1
FE79 7416 5657 JZ T3 ; SET_TIME
FE7B 5658 T1: ; TOD_RETURN
FE7B FB 5659 STI ; INTERRUPTS BACK ON
FE7C 1F 5660 POP DS ; RECOVER SEGMENT
FE7D CF 5661 IRET ; RETURN TO CALLER
FE7E 5662 T2: ; READ_TIME
FE7E FA 5663 CLI ; NO TIMER INTERRUPTS WHILE READING
FE7F A07000 5664 MOV AL,TIMER_OFL
FE82 C06700000 5665 MOV TIMER_OFL,0 ; GET OVERFLOW, AND RESET THE FLAG
FE87 8B0E6E00 5666 MOV CX,TIMER_HIGH
FE8B 8B166C00 5667 MOV DX,TIMER_LOW
FE8F EBEA 5668 JMP T1 ; TOD_RETURN
FE91 5669 T3: ; SET_TIME
FE91 FA 5670 CLI ; NO INTERRUPTS WHILE WRITING
FE92 89166C00 5671 MOV TIMER_LOW,DX
FE96 890E6E00 5672 MOV TIMER_HIGH,CX ; SET THE TIME
FE9A C06700000 5673 MOV TIMER_OFL,0 ; RESET OVERFLOW
FE9F EBD4 5674 JMP T1 ; TOD_RETURN
5675 TIME_OF_DAY ENDP
5676
5677 ;-----
5678 ; THIS ROUTINE HANDLES THE TIMER INTERRUPT FROM :
5679 ; CHANNEL 0 OF THE 8253 TIMER. INPUT FREQUENCY :
5680 ; IS 1.19318 MHZ AND THE DIVISOR IS 65536, RESULTING :
5681 ; IN APPROX. 18.2 INTERRUPTS EVERY SECOND. :
5682 ; :
5683 ; THE INTERRUPT HANDLER MAINTAINS A COUNT OF INTERRUPTS :
5684 ; SINCE POWER ON TIME, WHICH MAY BE USED TO ESTABLISH :
5685 ; TIME OF DAY. :
5686 ; THE INTERRUPT HANDLER ALSO DECREASES THE MOTOR :
5687 ; CONTROL COUNT OF THE DISKETTE, AND WHEN IT EXPIRES, :
5688 ; WILL TURN OFF THE DISKETTE MOTOR, AND RESET THE :
5689 ; MOTOR RUNNING FLAGS. :
5690 ; THE INTERRUPT HANDLER WILL ALSO INVOKE A USER ROUTINE :
5691 ; THROUGH INTERRUPT 1CH AT EVERY TIME TICK. THE USER :
5692 ; MUST CODE A ROUTINE AND PLACE THE CORRECT ADDRESS IN :
5693 ; THE VECTOR TABLE. :
5694 ;-----
FEA5 5695 ORG 0FEA5H
FEA5 5696 TIMER_INT PROC FAR

```

```

LOC OBJ          LINE    SOURCE

FEA5 FB          5697      STI              ; INTERRUPTS BACK ON
FEA6 1E          5698      PUSH DS
FEA7 50          5699      PUSH AX
FEA8 52          5700      PUSH DX          ; SAVE MACHINE STATE
FEA9 E8ADFB      5701      CALL DDS
FEAC FF06C000    5702      INC TIMER_LOW   ; INCREMENT TIME
FEB0 7504        5703      JNZ T4          ; TEST_DAY
FEB2 FF06E000    5704      INC TIMER_HIGH  ; INCREMENT HIGH WORD OF TIME
FEB4             5705      T4:            ; TEST_DAY
FEB6 833E6E0018  5706      CMP TIMER_HIGH,010H ; TEST FOR COUNT EQUALING 24 HOURS
FEBB 7515        5707      JNZ T5          ; DISKETTE_CTL
FEBD 813E6C00B000 5708      CMP TIMER_LOW,0B0H
FEC3 750D        5709      JNZ T5          ; DISKETTE_CTL
5710
5711      ;----- TIMER HAS GONE 24 HOURS
5712
FEC5 2BC0        5713      SUB AX,AX
FEC7 A36E00      5714      MOV TIMER_HIGH,AX
FECA A36C00      5715      MOV TIMER_LOW,AX
FECD C06E700001  5716      MOV TIMER_OFL,1
5717
5718      ;----- TEST FOR DISKETTE TIME OUT
5719
FED2             5720      T5:            ; DISKETTE_CTL
FED2 FE0E4000    5721      DEC MOTOR_COUNT
FED6 750B        5722      JNZ T6          ; RETURN IF COUNT NOT OUT
FED8 80263F00F0  5723      AND MOTOR_STATUS,0F0H ; TURN OFF MOTOR RUNNING BITS
FEDD B00C        5724      MOV AL,0CH
FEDF BAF203      5725      MOV DX,03F2H   ; FDC CTL PORT
FEE2 EE          5726      OUT DX,AL      ; TURN OFF THE MOTOR
FEE3             5727      T6:            ; TIMER_RET:
FEE3 CD1C        5728      INT 1CH        ; TRANSFER CONTROL TO A USER ROUTINE
FEE5 B020        5729      MOV AL,EDI
FEE7 E620        5730      OUT 020H,AL   ; END OF INTERRUPT TO 8259
FEE9 5A          5731      POP DX
FEEA 58          5732      POP AX
FEEB 1F          5733      POP DS        ; RESET MACHINE STATE
FEEC CF          5734      IRET          ; RETURN FROM INTERRUPT
5735      TIMER_INT   ENDP
5736
5737      ;-----
5738      ; THESE ARE THE VECTORS WHICH ARE MOVED INTO :
5739      ; THE 8086 INTERRUPT AREA DURING POWER ON. :
5740      ; ONLY THE OFFSETS ARE DISPLAYED HERE, CODE :
5741      ; SEGMENT WILL BE ADDED FOR ALL OF THEM, EXCEPT :
5742      ; WHERE NOTED. :
5743      ;-----
5744      ASSUME CS:CODE
5745      ORG OFE3H
5746      VECTOR_TABLE LABEL WORD ; VECTOR TABLE FOR MOVE TO INTERRUPTS
5747      DW OFFSET TIMER_INT ; INTERRUPT 8
5748      DW OFFSET KB_INT ; INTERRUPT 9
5749      DW OFFSET D11 ; INTERRUPT A
5750      DW OFFSET D11 ; INTERRUPT B
5751      DW OFFSET D11 ; INTERRUPT C
5752      DW OFFSET D11 ; INTERRUPT D
5753      DW OFFSET DISK_INT ; INTERRUPT E
5754      DW OFFSET D11 ; INTERRUPT F
5755      DW OFFSET VIDEO_IO ; INTERRUPT 10H
5756      DW OFFSET EQUIPMENT ; INTERRUPT 11H
5757      DW OFFSET MEMORY_SIZE_DET ; INTERRUPT 12H
5758      DW OFFSET DISKETTE_IO ; INTERRUPT 13H
5759      DW OFFSET RS232_IO ; INTERRUPT 14H
5760      DW CASSETTE_IO ; INTERRUPT 15H(FORMER CASSETTE IO)
5761      DW OFFSET KEYBOARD_IO ; INTERRUPT 16H
5762      DW OFFSET PRINTER_IO ; INTERRUPT 17H
5763
5764      FF13 0000    5764      DW 00000H ; INTERRUPT 18H
5765      DW 0F600H ; MUST BE INSERTED INTO TABLE LATER
5766
5767      FF15 F2E6    5767      DW OFFSET BOOT_STRAP ; INTERRUPT 19H
5768      FF17 6EFE    5768      DW TIME_OF_DAY ; INTERRUPT 1AH -- TIME OF DAY
5769      FF19 48FF    5769      DW DUMMY_RETURN ; INTERRUPT 1BH -- KEYBOARD BREAK ADDR
5770      FF1B 48FF    5770      DW DUMMY_RETURN ; INTERRUPT 1C -- TIMER BREAK ADDR
5771      FF1D A6F0    5771      DW VIDEO_PARMS ; INTERRUPT 1D -- VIDEO PARAMETERS
5772      FF1F C7EF    5772      DW OFFSET DISK_BASE ; INTERRUPT 1E -- DISK PARMS
5773      FF21 0000    5773      DW 0 ; INTERRUPT 1F -- POINTER TO VIDEO EXT

```

```

5774 ;-----
5775 ;
5776 ; TEMPORARY INTERRUPT SERVICE ROUTINE :
5777 ; 1. THIS ROUTINE IS ALSO LEFT IN PLACE AFTER THE :
5778 ; POWER ON DIAGNOSTICS TO SERVICE UNUSED :
5779 ; INTERRUPT VECTORS. LOCATION 'INTR_FLAG' WILL :
5780 ; CONTAIN EITHER: 1. LEVEL OF HARDWARE INT. THAT :
5781 ; CAUSED CODE TO BE EXEC. :
5782 ; 2. 'FF' FOR NON-HARDWARE INTERRUPTS THAT HAS :
5783 ; EXECUTED ACCIDENTLY. :
5784 ;-----
FF23 5785 D11 PROC NEAR
5786 ASSUME DS:DATA
FF23 5787 PUSH DS
FF24 5788 PUSH DX
FF25 5789 PUSH AX ; SAVE REG AX CONTENTS
FF26 E830FB 5790 CALL DDS
FF29 B00B 5791 MOV AL,0BH ; READ IN-SERVICE REG
FF2B E620 5792 OUT INTA00,AL ; (FIND OUT WHAT LEVEL BEING
FF2D 90 5793 NOP ; SERVICED)
FF2E E420 5794 IN AL,INTA00 ; GET LEVEL
FF30 8AE0 5795 MOV AH,AL ; SAVE IT
FF32 0AC4 5796 OR AL,AH ; 00? (NO HARDWARE ISR ACTIVE)
FF34 7504 5797 JNZ HM_INT
FF36 B4FF 5798 MOV AH,0FFH
FF38 EB0A 5799 JMP SHORT SET_INTR_FLAG ; SET FLAG TO FF IF NON-HDWARE
FF3A 5800 HM_INT:
FF3A E421 5801 IN AL,INTA01 ; GET MASK VALUE
FF3C 0AC4 5802 OR AL,AH ; MASK OFF LVL BEING SERVICED
FF3E E621 5803 OUT INTA01,AL
FF40 B020 5804 MOV AL,EOI
FF42 E620 5805 OUT INTA00,AL
FF44 5806 SET_INTR_FLAG:
FF44 8826B00 5807 MOV INTR_FLAG,AH ; SET FLAG
FF48 58 5808 POP AX ; RESTORE REG AX CONTENTS
FF49 5A 5809 POP DX
FF4A 1F 5810 POP DS
FF4B 5811 DUMMY_RETURN: ; NEED IRET FOR VECTOR TABLE
FF4B CF 5812 IRET
5813 D11 ENDP
5814
5815 ;-----
5816 ; DUMMY RETURN FOR ADDRESS COMPATIBILITY :
5817 ;-----
FF53 5818 ORG OFF53H
FF53 CF 5819 IRET
5820
5821 ;-- INT 5 -----
5822 ; THIS LOGIC WILL BE INVOKED BY INTERRUPT 05H TO PRINT THE :
5823 ; SCREEN. THE CURSOR POSITION AT THE TIME THIS ROUTINE IS INVOKED :
5824 ; WILL BE SAVED AND RESTORED UPON COMPLETION. THE ROUTINE IS :
5825 ; INTENDED TO RUN WITH INTERRUPTS ENABLED. IF A SUBSEQUENT :
5826 ; 'PRINT SCREEN' KEY IS DEPRESSED DURING THE TIME THIS ROUTINE :
5827 ; IS PRINTING IT WILL BE IGNORED. :
5828 ; ADDRESS 50:0 CONTAINS THE STATUS OF THE PRINT SCREEN: :
5829 ; :
5830 ; 50:0 =0 EITHER PRINT SCREEN HAS NOT BEEN CALLED :
5831 ; OR UPON RETURN FROM A CALL THIS INDICATES :
5832 ; A SUCCESSFUL OPERATION. :
5833 ; =1 PRINT SCREEN IS IN PROGRESS :
5834 ; =255 ERROR ENCOUNTERED DURING PRINTING :
5835 ;-----
5836 ASSUME CS:CODE,DS:X0DATA
FF54 5837 ORG OFF54H
FF54 5838 PRINT_SCREEN PROC FAR
FF54 FB 5839 STI ; MUST RUN WITH INTERRUPTS ENABLED
FF55 1E 5840 PUSH DS ; MUST USE 50:0 FOR DATA AREA STORAGE
FF56 50 5841 PUSH AX
FF57 53 5842 PUSH BX
FF58 51 5843 PUSH CX ; WILL USE THIS LATER FOR CURSOR LIMITS
FF59 52 5844 PUSH DX ; WILL HOLD CURRENT CURSOR POSITION
FF5A B85000 5845 MOV AX,X0DATA ; HEX 50
FF5D 8ED8 5846 MOV DS,AX
FF5F 803E000001 5847 CMP STATUS_BYTE,1 ; SEE IF PRINT ALREADY IN PROGRESS
FF64 745F 5848 JZ EXIT ; JUMP IF PRINT ALREADY IN PROGRESS
FF66 C606000001 5849 MOV STATUS_BYTE,1 ; INDICATE PRINT NOW IN PROGRESS
FF6B B40F 5850 MOV AH,15 ; WILL REQUEST THE CURRENT SCREEN MODE

```

LOC OBJ	LINE	SOURCE	
FF6D CD10	5851	INT 10H	; [AL]=MODE
	5852		; [AH]=NUMBER COLUMNS/LINE
	5853		; [BH]=VISUAL PAGE
	5854		;
	5855	; AT THIS POINT WE KNOW THE COLUMNS/LINE ARE IN	;
	5856	; [AX] AND THE PAGE IF APPLICABLE IS IN [BH]. THE STACK :	;
	5857	; HAS DS,AX,BX,CX,DX PUSHED. [A] HAS VIDEO MODE :	;
	5858		;
FF6F 8ACC	5859	MOV CL,AH	; WILL MAKE USE OF [CX] REGISTER TO
FF71 B519	5860	MOV CH,25	; CONTROL ROW & COLUMNS
FF73 E85500	5861	CALL CRLF	; CARRIAGE RETURN LINE FEED ROUTINE
FF76 51	5862	PUSH CX	; SAVE SCREEN BOUNDS
FF77 B403	5863	MOV AH,3	; WILL NOW READ THE CURSOR.
FF79 CD10	5864	INT 10H	; AND PRESERVE THE POSITION
FF7B 59	5865	POP CX	; RECALL SCREEN BOUNDS
FF7C 52	5866	PUSH DX	; RECALL [BH]=VISUAL PAGE
FF7D 3302	5867	XOR DX,DX	; WILL SET CURSOR POSITION TO [0,0]
	5868		;
	5869	; THE LOOP FROM PRI10 TO THE INSTRUCTION PRIOR TO PRI20 :	;
	5870	; IS THE LOOP TO READ EACH CURSOR POSITION FROM THE :	;
	5871	; SCREEN AND PRINT. :	;
	5872		;
FF7F	5873	PRI10:	
FF7F B402	5874	MOV AH,2	; TO INDICATE CURSOR SET REQUEST
FF81 CD10	5875	INT 10H	; NEW CURSOR POSITION ESTABLISHED
FF83 B408	5876	MOV AH,8	; TO INDICATE READ CHARACTER
FF85 CD10	5877	INT 10H	; CHARACTER NOW IN [AL]
FF87 0A00	5878	OR AL,AL	; SEE IF VALID CHAR
FF89 7502	5879	JNZ PRI15	; JUMP IF VALID CHAR
FF8B B020	5880	MOV AL,' '	; MAKE A BLANK
FF8D	5881	PRI15:	
FF8D 52	5882	PUSH DX	; SAVE CURSOR POSITION
FF8E 3302	5883	XOR DX,DX	; INDICATE PRINTER 1
FF90 32E4	5884	XOR AH,AH	; TO INDICATE PRINT CHAR IN [AL]
FF92 CD17	5885	INT 17H	; PRINT THE CHARACTER
FF94 5A	5886	POP DX	; RECALL CURSOR POSITION
FF95 F6C425	5887	TEST AH, 25H	; TEST FOR PRINTER ERROR
FF98 7521	5888	JNZ ERR10	; JUMP IF ERROR DETECTED
FF9A FEC2	5889	INC DL	; ADVANCE TO NEXT COLUMN
FF9C 3ACA	5890	CHP CL,DL	; SEE IF AT END OF LINE
FF9E 75DF	5891	JNZ PRI10	; IF NOT PROCEED
FFA0 32D2	5892	XOR DL,DL	; BACK TO COLUMN 0
FFA2 8AE2	5893	MOV AH,DL	; [AH]=0
FFA4 52	5894	PUSH DX	; SAVE NEW CURSOR POSITION
FFA5 E82300	5895	CALL CRLF	; LINE FEED CARRIAGE RETURN
FFA8 5A	5896	POP DX	; RECALL CURSOR POSITION
FFA9 FEC6	5897	INC DH	; ADVANCE TO NEXT LINE
FFAB 3AEE	5898	CHP CH,DH	; FINISHED?
FFAD 75D0	5899	JNZ PRI10	; IF NOT CONTINUE
FFAF	5900	PRI20:	
FFAF 5A	5901	POP DX	; RECALL CURSOR POSITION
FFB0 B402	5902	MOV AH,2	; TO INDICATE CURSOR SET REQUEST
FFB2 CD10	5903	INT 10H	; CURSOR POSITION RESTORED
FFB4 C606000000	5904	MOV STATUS_BYTE,0	; INDICATE FINISHED
FFB9 E80A	5905	JMP SHORT EXIT	; EXIT THE ROUTINE
FFBB	5906	ERR10:	
FFBB 5A	5907	POP DX	; GET CURSOR POSITION
FFBC B402	5908	MOV AH,2	; TO REQUEST CURSOR SET
FFBE CD10	5909	INT 10H	; CURSOR POSITION RESTORED
FFC0	5910	ERR20:	
FFC0 C6060000FF	5911	MOV STATUS_BYTE,0FFH	; INDICATE ERROR
FFC5	5912	EXIT:	
FFC5 5A	5913	POP DX	; RESTORE ALL THE REGISTERS USED
FFC6 59	5914	POP CX	
FFC7 5B	5915	POP BX	
FFC8 58	5916	POP AX	
FFC9 1F	5917	POP DS	
FFCA CF	5918	IRET	
	5919	PRINT_SCREEN	ENDP
	5920		
	5921	;	----- CARRIAGE RETURN, LINE FEED SUBROUTINE
	5922		
FFCB	5923	CRLF PROC NEAR	
FFCB 3302	5924	XOR DX,DX	; PRINTER 0
FFCD 32E4	5925	XOR AH,AH	; WILL NOW SEND INITIAL LF,CR
	5926		; TO PRINTER
FFCF B00A	5927	MOV AL,120	; LF

```

LOC OBJ          LINE  SOURCE
FFD1 CD17        5928      INT    17H          ; SEND THE LINE FEED
FFD3 32E4        5929      XOR    AH,AH        ; NOW FOR THE CR
FFD5 B0D0        5930      MOV    AL,15Q       ; CR
FFD7 CD17        5931      INT    17H          ; SEND THE CARRIAGE RETURN
FFD9 C3          5932      RET
5933            CRLF   ENDP
5934
5935            ;-----
5936            ; PRINT A SEGMENT VALUE TO LOOK LIKE A 20 BIT ADDRESS :
5937            ; DX MUST CONTAIN SEGMENT VALUE TO BE PRINTED :
5938            ;-----
FFDA            5939      PRT_SEG PROC  NEAR
FFDA 8AC6        5940      MOV    AL,0H        ;GET MSB
FFDC E8ACF9      5941      CALL  XPC_BYTE
FFDF 8AC2        5942      MOV    AL,DL        ;LSB
FFE1 E8A7F9      5943      CALL  XPC_BYTE
FFE4 B030        5944      MOV    AL,'0'       ; PRINT A '0 '
FFE6 E8B3F9      5945      CALL  PRT_HEX
FFE9 B020        5946      MOV    AL,' '       ;SPACE
FFEB E8AEF9      5947      CALL  PRT_HEX
FFEE C3          5948      RET
5949            PRT_SEG ENDP
5950
----           5951      CODE   ENDS
5952
5953            ;-----
5954            ; POWER ON RESET VECTOR :
5955            ;-----
----           5956      VECTOR SEGMENT AT 0FFFFH
5957
5958            ;---- POWER ON RESET
5959
0000 EASBE00F0   5960      JMP    RESET
5961
0005 31312F30382F30 5962      DB    '11/08/82'   ; RELEASE MARKER
32
----           5963      VECTOR ENDS
5964            END

```

Appendix A

```

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 1-16D)
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 ; REMITTEN. (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 REPLY THE OPERATION.
94 ;
95 ;-----
96
00FF 97 SENSE_FAIL EQU 0FFH ; SENSE OPERATION FAILED
00BB 98 UNDEF_ERR EQU 0BBH ; UNDEFINED ERROR OCCURRED
0080 99 TIME_OUT EQU 80H ; ATTACHMENT FAILED TO RESPOND
0040 100 BAD_SEEK EQU 40H ; SEEK OPERATION FAILED
0020 101 BAD_CNTRLR 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 0DH*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
0078 123 ORG 1EH*4 ; DISKETTE PARAMETERS
0078 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 ??? 138 TIMER_LOW DW ? ; TIMER LOW WORD
0072 139 ORG 72H
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: :

```

LOC OBJ

LINE SOURCE

```

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 HF_PORT EQU 0320H ; DISK PORT
0008 R1_BUSY EQU 00001000B ; DISK PORT 1 BUSY BIT
0004 R1_BUS EQU 00000100B ; COMMAND/DATA BIT
0002 R1_IOMODE EQU 00000010B ; MODE BIT
0001 R1_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
0082 178 DMA_HIGH EQU 082H ; PORT FOR HIGH 4 BITS OF DMA
179
0000 180 TST_RDY_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 EDI EQU 20H ; END OF INTERRUPT COMMAND
202
0008 203 MAX_FILE EQU 8
0002 204 S_MAX_FILE EQU 2
205
0000 206 ASSUME CS:CODE
0000 55 207 ORG 0H
0001 AA 208 DB 055H ; GENERIC BIOS HEADER
0002 10 209 DB 0AAH
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 3530303030303539 223 DB '5000059 (C)COPYRIGHT IBM 1982' ; COPYRIGHT NOTICE
20284329434F50
59524947485420
2049424D203139
3832
0023 224 L3:
225 ASSUME DS:DUMMY
0023 28C0 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 B66007	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 C70644008601	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 C066740000	247	MOV DISK_STATUS,0	; RESET THE STATUS INDICATOR
0067 C066750000	248	MOV HF_NUM,0	; ZERO COUNT OF DRIVES
006C C066430000	249	MOV CHD_BLOCK+1,0	; DRIVE ZERO, SET VALUE IN BLOCK
0071 C066770000	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 3D3412	274	CHP AX,1234H	; KEYBOARD RESET
00A9 7506	275	JNE P8	
00AB C7066C009A01	276	MOV TIMER_LOW,410D	; SKIP WAIT ON RESET
00B1	277		
00B1 E421	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 3D0E01	289	CHP AX,446D	; 25 SECONDS
00C9 72EC	290	JB P4	
00CB E87590	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

LOC OBJ	LINE	SOURCE		
00E5 0E0C	305	MOV	ES,AX	; SET SEGMENT
00E7 2BDB	306	SUB	BX,BX	
00E9 B800F	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	CMF	AL,0FH	
0102 7406	320	JE	BOX_ON	
0104 C7066C00A401	321	MOV	TIMER_LOW,420D	; CONTROLLER IS IN SYSTEM UNIT
010A	322	BOX_ON:		
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	P3:		
0116 2BC0	330	SUB	AX,AX	; RESET
0118 CD13	331	INT	13H	
011A 7240	332	JC	POD_DONE	
011C B80011	333	MOV	AX,01100H	; RECAL
011F CD13	334	INT	13H	
0121 730B	335	JNC	P5	
0123 A16C00	336	MOV	AX,TIMER_LOW	
0126 3DBE01	337	CMF	AX,446D	; 25 SECONDS
0129 72EB	338	JB	P3	
012B EB2F90	339	JMP	POD_DONE	
012E	340	P5:		
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,(80H + S_MAX_FILE - 1)	
013D 731D	346	JAE	POD_DONE	
013F 42	347	INC	DX	
0140 EBD4	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 2E8A846801	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	
016C 0D				
016D 0A				
0006	375	F17L EQU	6-F17	
	376			
016E	377	HD_RESET_1	PROC NEAR	
016E 51	378	PUSH	CX	; SAVE REGISTER
016F 52	379	PUSH	DX	

LOC OBJ	LINE	SOURCE	
0170 F8	380	CLC	; CLEAR CARRY
0171 B9001	381	MOV CX,0100H	; RETRY COUNT
0174	382		
0174 E80706	383	L6: 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		
0183 5A	392	R3: 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	
0186 8ED8	422	MOV DS,AX	; ESTABLISH SEGMENT
	423		
	424	;----- RESET PARAMETER VECTORS	
	425		
018A FA	426	CLI	
018B C760401E703	427	MOV WORD PTR HF_TBL_VEC, OFFSET FD_TBL	
0191 8C0E0601	428	MOV WORD PTR HF_TBL_VEC+2, CS	
0195 C76E78000102	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 2B02	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 2B02	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
01BB 59	450	H2: POP CX	; RECOVER RETRY COUNT
01BC 730A	451	JNC H4	; CF SET BY UNSUCCESSFUL READ
01BE 80FC80	452	CMF 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 EB0690	455	JMP H5	; UNABLE TO IPL FROM THE DISKETTE
01C8	456	H4: ; IPL WAS SUCCESSFUL	

LOC CBJ	LINE	SOURCE		
01C8 EA007C0000	457	JMP	BOOT_LOCN	
	458			
	459	;----- ATTEMPT BOOTSTRAP FROM FIXED DISK		
	460			
01CD	461	H5:		
01C0 2BC0	462	SUB	AX,AX	; RESET DISKETTE
01CF 2BD2	463	SUB	DX,DX	
01D1 CD13	464	INT	13H	
01D3 B90300	465	MOV	CX,3	; SET RETRY COUNT
01D6	466			; 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
01E9 B80102	472	MOV	AX,0201H	; READ IN THE SINGLE SECTOR
01E3 2B0B	473	SUB	BX,BX	
01E5 8EC3	474	MOV	ES,BX	
01E7 B8007C	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
01FD CD13	478	INT	13H	; FILE IO CALL
01F2 59	479	H7:	PDP CX	; RECOVER RETRY COUNT
01F3 7208	480	JC	H8	
01F5 A1FE7D	481	MOV	AX,WORD PTR BOOT_LOCN+510D	
01F8 3D55AA	482	CMF	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=OF - 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 8E08	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 E86905	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,4H	
0227 E85E05	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 E85305	525	CALL	PORT_3	
0235 2AC0	526	SUB	AL,AL	
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

A-90 Fixed Disk BIOS

LOC OBJ	LINE	SOURCE	
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 86267700	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 80FAB1	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 E888FF	581	CALL DSBL	; BE SURE DISABLES OCCURRED
0284 B84000	582	MOV AX,DATA	
0287 0E08	583	MOV DS,AX	; ESTABLISH SEGMENT
0289 58	584	POP AX	
028A 8A267400	585	MOV AH,DISK_STATUS	; GET STATUS FROM OPERATION
028E 80FC01	586	CHP AH,1	; SET THE CARRY FLAG TO INDICATE
0291 F5	587	CHC	; 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	H1 LABEL WORD	; FUNCTION TRANSFER TABLE
029C 3803	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_VERIFY	; 004H
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 DD04	610	DW WR_LONG	; 00BH

LOC OBJ	LINE	SOURCE		
02B4 F204	611	DM	DISK_SEEK	; 00CH
02B6 3803	612	DM	DISK_RESET	; 00DH
02B8 F904	613	DM	RD_BUFF	; 00EH
02BA 0705	614	DM	WR_BUFF	; 00FH
02BC 1505	615	DM	TST_RDY	; 010H
02BE 1C05	616	DM	HDISK_RECAL	; 011H
02C0 2305	617	DM	RAM_DIAG	; 012H
02C2 2A05	618	DM	CHK_DRV	; 013H
02C4 3105	619	DM	CNTRL_DIAG	; 014H
002A	620	MIL	EQU	\$-M1
	621			
02C6	622	SETUP_A PROC	NEAR	
	623			
02C6 C06740000	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	CMD_BLOCK+1,DL	
02E8 59	641	POP	CX	
02E9 C3	642	RET		
	643	SETUP_A ENDP		
	644			
02EA	645	DISK_IO_CONT PROC	NEAR	
02EA 50	646	PUSH	AX	
02EB B84000	647	MOV	AX,DATA	
02EE 8E08	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 C06420000	663	MOV	CMD_BLOCK+0,0	
0308 880E4400	664	MOV	CMD_BLOCK+2,CL	; SECTOR AND HIGH 2 BITS CYLINDER
030F 882E4500	665	MOV	CMD_BLOCK+3,CH	; CYLINDER
0313 A24600	666	MOV	CMD_BLOCK+4,AL	; INTERLEAVE / BLOCK COUNT
0316 A07600	667	MOV	AL,CONTROL_BYTE	; CONTROL BYTE (STEP OPTION)
0319 A24700	668	MOV	CMD_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 2EFA49C02	677	JMP	WORD PTR CS:[SI + OFFSET M1]	
0330	678	BAD_COMMAND:		
0330 C06740001	679	MOV	DISK_STATUS,BAD_CMD	; COMMAND ERROR
0335 B000	680	MOV	AL,0	
0337 C3	681	RET		
	682	DISK_IO_CONT	ENDP	
	683			
	684			;-----
	685			; RESET THE DISK SYSTEM (AH = 000H) ;
	686			;-----
	687			

LOC	OBJ	LINE	SOURCE
0330		608	DISK_RESET PROC NEAR
0330 E04304		609	CALL PORT_1 ; RESET PORT
0330 EE		690	OUT DX,AL ; ISSUE RESET
033C E03F04		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 C066740005		695	MOV DISK_STATUS,BAD_RESET
0349 C3		696	RET
034A		697	DR1:
034A E90A0D		698	JMP INIT_DRV ; SET THE DRIVE PARAMETERS
		699	DISK_RESET ENDP
		700	
		701	;
		702	; DISK STATUS ROUTINE (AH = 001H) ;
		703	;
		704	
0340		705	RETURN_STATUS PROC NEAR
0340 A07400		706	MOV AL,DISK_STATUS ; OBTAIN PREVIOUS STATUS
0350 C066740000		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 C06642000B		717	MOV CMD_BLOCK+0,READ_CMD
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 C06642000A		727	MOV CMD_BLOCK+0,WRITE_CMD
0367 E9DB01		728	JMP DMA_OPN
		729	DISK_WRITE ENDP
		730	
		731	;
		732	; DISK VERIFY (AH = 004H) ;
		733	;
		734	
036A		735	DISK_VERF PROC NEAR
036A C066420005		736	MOV CMD_BLOCK+0,CHK_TRK_CMD
036F E9C401		737	JMP NDMA_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 C066420006		745	MOV CMD_BLOCK,FMTTRK_CMD
0377 EB0C		746	JMP SHORT FMT_CONT
		747	FMT_TRK ENDP
		748	
0379		749	FMT_BAD PROC NEAR ; FORMAT BAD TRACK (AH = 006H)
0379 C066420007		750	MOV CMD_BLOCK,FMTBAD_CMD
037E EB05		751	JMP SHORT FMT_CONT
		752	FMT_BAD ENDP
		753	
0380		754	FMT_DRV PROC NEAR ; FORMAT DRIVE (AH = 007H)
0380 C066420004		755	MOV CMD_BLOCK,FMTDRV_CMD
		756	FMT_DRV ENDP
		757	
0385		758	FMT_CONT:
0385 A04400		759	MOV AL,CMD_BLOCK+2 ; ZERO OUT SECTOR FIELD
0388 24C0		760	AND AL,11000000B
038A A24400		761	MOV CMD_BLOCK+2,AL
038D E9A601		762	JMP NDMA_OPN
		763	

LOC OBJ

LINE SOURCE

```

764 |-----|
765 ; GET PARAMETERS (AH = 0) ;
766 |-----|
767
0390 GET_PARM_N LABEL NEAR
0390 GET_PARM PROC FAR ; GET DRIVE PARAMETERS
0390 1E PUSH DS ; SAVE REGISTERS
0391 06 PUSH ES
0392 53 PUSH BX
773
774 ASSUME DS:DUMMY
0393 2BC0 SUB AX,AX ; ESTABLISH ADDRESSING
0395 8ED8 MOV DS,AX
0397 C41E0401 LES BX,HF_TBL_VEC
778 ASSUME DS:DATA
0398 B84000 MOV AX,DATA
039E 8ED8 MOV DS,AX ; ESTABLISH SEGMENT
781
03A0 80EA80 SUB DL,80H
03A3 80FA08 CMP DL,MAX_FILE ; TEST WITHIN RANGE
03A6 732F JAE G4
785
03A8 E81BFF CALL SETUP_A
787
03AB E8DF03 CALL SW2_OFFS
03AE 7227 JC G4
03B0 03D8 ADD BX,AX
791
03B2 268B07 MOV AX,ES:[BX] ; MAX NUMBER OF CYLINDERS
03B5 2D0200 SUB AX,2 ; ADJUST FOR 0-N
; AND RESERVE LAST TRACK
794
03B8 0AE8 MOV CH,AL
03BA 250003 AND AX,0300H ; HIGH TWO BITS OF CYL
03BD D1E8 SHR AX,1
03BF D1E8 SHR AX,1
03C1 0C11 OR AL,011H ; SECTORS
03C3 8AC8 MOV CL,AL
801
03C5 268A7702 MOV DH,ES:[BX][2] ; HEADS
03C9 FECE DEC DH ; 0-N RANGE
03CB 8A167500 MOV DL,HF_NUM ; DRIVE COUNT
03CF 2BC0 SUB AX,AX
G5:
03D1 806
03D1 58 POP BX ; RESTORE REGISTERS
03D2 07 POP ES
03D3 1F POP DS
03D4 CA0200 RET 2
G4:
03D7 811
03D7 C606740007 MOV DISK_STATUS,INIT_FAIL ; OPERATION FAILED
03DC B407 MOV AH,INIT_FAIL
03DE 2AC0 SUB AL,AL
03E0 2BD2 SUB DX,DX
03E2 2BC9 SUB CX,CX
03E4 F9 STC ; SET ERROR FLAG
03E5 EBEA JMP G5
819 GET_PARM ENDP
820
821 |-----|
822 ; INITIALIZE DRIVE CHARACTERISTICS ;
823 ; ;
824 ; FIXED DISK PARAMETER TABLE ;
825 ; ;
826 ; - THE TABLE IS COMPOSED OF A BLOCK DEFINED AS: ;
827 ; ;
828 ; (1 WORD) - MAXIMUM NUMBER OF CYLINDERS ;
829 ; (1 BYTE) - MAXIMUM NUMBER OF HEADS ;
830 ; (1 WORD) - STARTING REDUCED WRITE CURRENT CYL ;
831 ; (1 WORD) - STARTING WRITE PRECOMPENSATION CYL ;
832 ; (1 BYTE) - MAXIMUM ECC DATA BURST LENGTH ;
833 ; (1 BYTE) - CONTROL BYTE (DRIVE STEP OPTION) ;
834 ; BIT 7 DISABLE DISK-ACCESS RETRIES ;
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 CARD SWITCH SETTINGS ;
854 ; ;
855 ;           DRIVE 0   DRIVE 1 ;
856 ;           ----- ;
857 ;           ON   :   -1-  -2-  /  -3-  -4-  : ;
858 ;           OFF  :   /      /      /      /      : ;
859 ;           ----- ;
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 FD_TBL:
874 ;----- DRIVE TYPE 00
875 ;
876 ;
877 ;
03E7 3201 878 DW 0306D
03E9 02 879 DB 02D
03EA 3201 880 DW 0306D
03EC 0000 881 DW 0000D
03EE 0B 882 DB 0BH
03EF 00 883 DB 00H
03F0 0C 884 DB 0CH ; STANDARD
03F1 B4 885 DB 0B4H ; FORMAT DRIVE
03F2 28 886 DB 028H ; CHECK DRIVE
03F3 00000000 887 DB 0,0,0,0
888 ;
889 ;----- DRIVE TYPE 01
890 ;
03F7 7701 891 DW 0375D
03F9 08 892 DB 08D
03FA 7701 893 DW 0375D
03FC 0000 894 DW 0000D
03FE 0B 895 DB 0BH
03FF 05 896 DB 05H
0400 0C 897 DB 0CH ; STANDARD
0401 B4 898 DB 0B4H ; FORMAT DRIVE
0402 28 899 DB 028H ; CHECK DRIVE
0403 00000000 900 DB 0,0,0,0
901 ;
902 ;----- DRIVE TYPE 02
903 ;
0407 3201 904 DW 0306D
0409 06 905 DB 06D
040A 8000 906 DW 0128D
040C 0001 907 DW 0256D
040E 0B 908 DB 0BH
040F 05 909 DB 05H
0410 0C 910 DB 0CH ; STANDARD
0411 B4 911 DB 0B4H ; FORMAT DRIVE
0412 28 912 DB 028H ; 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

```

LOC OBJ	LINE	SOURCE	
041A 3201	919	DH	0306D
041C 0000	920	DH	0000D
041E 0B	921	DB	0B8
041F 05	922	DB	058
0420 0C	923	DB	0C8
0421 B4	924	DB	0B4H
0422 28	925	DB	028H
0423 00000000	926	DB	0,0,0,0
	927		
0427	928	INIT_DRV	PROC NEAR
	929		
	930	I----- DO DRIVE ZERO	
	931		
0427 C60642000C	932	MOV	CHD_BLOCK+0,INIT_DRV_CMD
042C C60643000D	933	MOV	CHD_BLOCK+1,0
0431 E81000	934	CALL	INIT_DRV_R
0434 720D	935	JC	INIT_DRV_OUT
	936		
	937	I----- DO DRIVE ONE	
	938		
0436 C60642000C	939	MOV	CHD_BLOCK+0,INIT_DRV_CMD
043B C60643002D	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 2ACD	948	SUB	AL,AL
0446 E81901	949	CALL	COMMAND
0449 7301	950	JNC	B1
044B C3	951	RET	
044C	952	B1:	
044C 1E	953	PUSH	DS
	954	ASSUME	DS:DUMMY
0440 2BC0	955	SUB	AX,AX
044F 8E08	956	MOV	DX,AX
0451 C41E0401	957	LES	BX,HF_TBL_VEC
0455 1F	958	POP	DS
	959	ASSUME	DS:DATA
0456 E83403	960	CALL	SM2_OFFS
0459 7257	961	JC	B3
045B 0308	962	ADD	BX,AX
	963		
	964	I----- 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 723D	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
	989		
048D 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

```

LOC OBJ          LINE  SOURCE

049B 7215        996          JC      B3
                997
049D BF0800     998          MOV     DI,8           ; DRIVE STEP OPTION
04A0 268A01     999          MOV     AL,ES:[BX + DI]
04A3 A27600    1000         MOV     CONTROL_BYTE,AL
                1001
04A6 2BC9       1002         SUB     CX,CX
04A8           1003         B5:
04A8 E8D302     1004         CALL   PORT_1
04AB EC         1005         IN     AL,DX
04AC A802     1006         TEST   AL,R1_IOMODE   ; STATUS INPUT MODE
04AE 7509     1007         JNZ   B6
04B0 E2F6     1008         LOOP  B5
04B2           1009         B3:
04B2 C06740007  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 C0664200E5  1042         MOV     CMD_BLOCK+0,RD_LONG_CMD
04D9 B047     1043         MOV     AL,DMA_READ
04DB EB68     1044         JHP   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 C0664200E6  1054         MOV     CMD_BLOCK+0,WR_LONG_CMD
04E7 B04B     1055         MOV     AL,DMA_WRITE
04E9 EB5A     1056         JHP   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         CHP   AL,080H
04F0 F5       1062         CMC
04F1 C3       1063         RET
                1064         CHK_LONG     ENDP
                1065
                1066         ;-----
                1067         ;   SEEK (AH = 0CH)   ;
                1068         ;-----
                1069
04F2           1070         DISK_SEEK   PROC   NEAR
04F2 C06642000B  1071         MOV     CMD_BLOCK,SEEK_CMD
04F7 EB3D     1072         JHP   SHORT NDMA_OPN

```

```

LOC OBJ          LINE  SOURCE

1073  DISK_SEEK      ENDP
1074
1075  ;-----
1076  ;      READ SECTOR BUFFER  (AH = 0EH)      :
1077  ;-----
1078
04F9  1079  RD_BUFF PROC      NEAR
04F9 C60642000E     1080      MOV      CHD_BLOCK+0,RD_BUFF_CHD
04FE C606460001     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  MR_BUFF PROC      NEAR
0507 C60642000F     1091      MOV      CHD_BLOCK+0,MR_BUFF_CHD
050C C606460001     1092      MOV      CHD_BLOCK+4,1      ; ONLY ONE BLOCK
0511 B04B           1093      MOV      AL,DMA_WRITE
0513 EB30          1094      JMP      SHORT  DMA_OPN
1095  MR_BUFF ENDP
1096
1097  ;-----
1098  ;      TEST DISK READY  (AH = 010H)      :
1099  ;-----
1100
0515  1101  TST_RDY PROC      NEAR
0515 C606420000     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 C606420001     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 C60642000E     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 C6064200E3     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  CNTLR_DIAG PROC  NEAR
0531 C6064200E4     1138      MOV      CHD_BLOCK+0,CNTLR_DIAG_CHD
1139  CNTLR_DIAG ENDP
1140
1141  ;-----
1142  ;      SUPPORT ROUTINES      :
1143  ;-----
1144
0536  1145  NDMA_OPN:
0536 B002           1146      MOV      AL,02H
0538 E82700        1147      CALL     COMMAND      ; ISSUE THE COMMAND
053B 7221           1148      JC      G11
053D EB16          1149      JMP      SHORT  G3

```

LOC OBJ	LINE	SOURCE
053F	1150	60:
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 720D	1158	JC G11
0551 B003	1159	MOV AL,03H
0553 E60A	1160	OUT DMA+10,AL ; INITIALIZE THE DISK CHANNEL
0555	1161	63:
0555 E421	1162	IN AL,021H
0557 240F	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 28C9	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 3C0D	1189	CMP 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_CNTLRL
059A F9	1209	STC
059B	1210	CH7:
059B C3	1211	RET
	1212	COMMAND ENDP
	1213	
	1214	-----
	1215	; SENSE STATUS BYTES :
	1216	; :
	1217	; BYTE 0 :
	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 :

```

LOC OBJ          LINE   SOURCE

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             1238     ASSUME   ES:DATA
059C A07400      1239     MOV    AL,DISK_STATUS   ; CHECK IF THERE HAS AN ERROR
059F 0AC0       1240     OR    AL,AL
05A1 7501       1241     JNZ   621
05A3 C3        1242     RET
1243
1244             ;---- PERFORM SENSE STATUS
1245
05A4             1246   G21:
05A4 B84000     1247     MOV    AX,DATA
05A7 8ECC      1248     MOV    ES,AX             ; ESTABLISH SEGMENT
05A9 2BC0      1249     SUB    AX,AX
05AB 8BF8      1250     MOV    DI,AX
05AD C606420003 1251     MOV    CMD_BLOCK+0,SENSE_CMD
05B2 2AC0      1252     SUB    AL,AL
05B4 E8ABFF     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 E8CB00     1257     CALL  HD_WAIT_REQ
05BF 7220      1258     JC    G24
05C1 E8AD01     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 E8B101     1263     CALL  PORT_1
05CD E2ED      1264     LOOP  G22
05CF E8B800     1265     CALL  HD_WAIT_REQ
05D2 720D      1266     JC    G24
05D4 E89A01     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 C6067400FF 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
0602 00204020800020 1295     DB   0,BAD_CNTRLR,BAD_SEEK,BAD_CNTRLR,TIME_OUT,0,BAD_CNTRLR
0609 0040      1296     DB   0,BAD_SEEK
0009         1297     TYPE0_LEN EQU  $-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  $-TYPE1_TABLE
0615         1302   TYPE2_TABLE LABEL BYTE
0615 0102      1303     DB   BAD_CMD,BAD_ADDR_MARK

```


LOC OBJ	LINE	SOURCE
0002	1304	TYPE2_LEN EQU \$-TYPE2_TABLE
0617	1305	TYPE3_TABLE LABEL BYTE
0617 202010	1306	DB BAD_CNTRL,BAD_CNTRL,BAD_ECC
0003	1307	TYPE3_LEN EQU \$-TYPE3_TABLE
	1308	
	1309	;----- TYPE 0 ERROR
	1310	
061A	1311	TYPE_0:
061A BB0206	1312	MOV BX,OFFSET TYPE0_TABLE
061D 3C09	1313	CMP 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	;----- TYPE 1 ERROR
	1320	
0627	1321	TYPE_1:
0627 BB0806	1322	MOV BX,OFFSET TYPE1_TABLE
062A 8BC8	1323	MOV CX,AX
062C 3C0A	1324	CMP 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,08H ; CORRECTED ECC
0638 80F908	1329	CHP CL,08H
063B 752A	1330	JNZ 630
	1331	
	1332	;----- OBTAIN ECC ERROR BURST LENGTH
	1333	
063D C60642000D	1334	MOV CMD_BLOCK+0,RD_ECC_CMD
0642 2AC0	1335	SUB AL,AL
0644 E81BFF	1336	CALL COMMAND
0647 721E	1337	JC 630
0649 E83E00	1338	CALL HD_WAIT_REQ
064C 7219	1339	JC 630
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 630
0659 E81501	1345	CALL PORT_0
065C EC	1346	IN AL,DX
065D A801	1347	TEST AL,01H
065F 7406	1348	JZ 630
0661 C606740020	1349	MOV DISK_STATUS,BAD_CNTRL
0666 F9	1350	STC
0667	1351	630:
0667 8AC1	1352	MOV AL,CL
0669 C3	1353	RET
	1354	
	1355	;----- TYPE 2 ERROR
	1356	
066A	1357	TYPE_2:
066A BB1506	1358	MOV BX,OFFSET TYPE2_TABLE
066D 3C02	1359	CMP AL,TYPE2_LEN ; CHECK IF ERROR IS DEFINED
066F 7313	1360	JAE UNDEF_ERR_L
0671 2ED7	1361	XLAT CS:TYPE2_TABLE ; TABLE LOOKUP
0673 A27400	1362	MOV DISK_STATUS,AL ; SET ERROR CODE
0676 C3	1363	RET
	1364	
	1365	;----- TYPE 3 ERROR
	1366	
0677	1367	TYPE_3:
0677 BB1706	1368	MOV BX,OFFSET TYPE3_TABLE
067A 3C03	1369	CMP AL,TYPE3_LEN
067C 7306	1370	JAE UNDEF_ERR_L
067E 2ED7	1371	XLAT CS:TYPE3_TABLE
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

LOC OBJ	LINE	SOURCE	
060B 2BC9	1381	SUB	CX,CX
060D E8EE00	1382	CALL	PORT_1
0690	1383	L1:	
0690 EC	1384	IN	AL,DX
0691 A801	1385	TEST	AL,R1_REQ
0693 7508	1386	JNZ	L2
0695 E2F9	1387	LOOP	L1
0697 C0674008D	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,CMD_BLOCK+4
06A3 3CB1	1407	CMF	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 24F0	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,CMD_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,0FFH
	1443		
	1444	;	----- HANDLE READ AND WRITE LONG (5160 BYTE BLOCKS)
	1445		
06DA 50	1446	PUSH	AX ; SAVE REGISTER
06DB A04200	1447	MOV	AL,CMD_BLOCK+0 ; GET COMMAND
06DE 3CE5	1448	CMF	AL,RD_LONG_CMD
06E0 7407	1449	JE	ADD4
06E2 3CE6	1450	CMF	AL,WR_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

```

LOC OBJ                LINE  SOURCE
06EE 2AFF              1458          SUB    BH,BH
06F0 8A1E4600         1459          MOV    BL,CHD_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
DMA_SETUP              1477          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 8E08              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 268A5C09         1502          MOV    BL,BYTE PTR ES:[SI]9          ; STANDARD TIME OUT
071D 8A264200         1503          MOV    AH,CHD_BLOCK
0721 80FC04            1504          CMP    AH,FMTDRV_CHD
0724 7506              1505          JNZ    M5
0726 268A5C0A         1506          MOV    BL,BYTE PTR ES:[SI]0AH        ; FORMAT DRIVE
072A EB09              1507          JMP    SHORT M4
072C 80FCE3           1508          M5:  CMP    AH,CHK_DRV_CHD
072F 7504              1509          JNZ    M4
0731 268A5C0B         1510          MOV    BL,BYTE PTR ES:[SI]0BH        ; CHECK DRIVE
0735                1511          M4:
0735 2BC9              1512          SUB    CX,CX
1513
1514          ;---- WAIT FOR INTERRUPT
1515
0737                1516          M1:
0737 E04400           1517          CALL   PORT_1
073A EC                1518          IN     AL,DX
073B 2420             1519          AND    AL,020H
073D 3C20             1520          CMP    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:
074B E82300           1527          CALL   PORT_0
074E EC                1528          IN     AL,DX
074F 2402             1529          AND    AL,2                ; ERROR BIT
0751 0B067400         1530          OR     DISK_STATUS,AL      ; SAVE
0755 E83000           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

```

```

LOC OBJ          LINE    SOURCE

075C 07          1535      POP     ES
075D 59          1536      POP     CX
075E 5B          1537      POP     BX
075F C3          1538      RET
0759          1539      WAIT_INT  ENDP
0760          1540

0760          1541      HO_INT  PROC  NEAR
0760 50          1542      PUSH   AX
0761 B020        1543      MOV     AL,EDI                ; 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 58          1550      POP     AX
0770 CF          1551      IRET
0770          1552      HO_INT  ENDP
0770          1553
0770          1554      ;-----
0770          1555      ; PORTS                :
0770          1556      ; GENERATE PROPER PORT VALUE :
0770          1557      ; BASED ON THE PORT OFFSET :
0770          1558      ;-----
0770          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 58          1566      POP     AX
077D C3          1567      RET
077D          1568      PORT_0  ENDP
077D          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
0782          1574      PORT_1  ENDP
0782          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
0787          1580      PORT_2  ENDP
0787          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
078C          1586      PORT_3  ENDP
078C          1587
078C          1588      ;-----
078C          1589      ; SM2_OFFS                :
078C          1590      ; DETERMINE PARAMETER TABLE OFFSET :
078C          1591      ; USING CONTROLLER PORT TWO AND :
078C          1592      ; DRIVE NUMBER SPECIFIER (0-1) :
078C          1593      ;-----
078C          1594

078D          1595      SM2_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 58          1602      POP     AX
0799 7516        1603      JNZ    SM2_OFFS_ERR
079B 8A264300    1604      MOV     AH,CMD_BLOCK+1
079F 80E420      1605      AND    AH,00100000B        ; DRIVE 0 OR 1
07A2 7504        1606      JNZ    SM2_AND
07A4 D0E8        1607      SHR    AL,1                ; ADJUST
07A6 D0E8        1608      SHR    AL,1
07A8          1609      SM2_AND:
07A8 2403        1610      AND    AL,011B            ; ISOLATE
07AA B104        1611      MOV     CL,4

```

LOC OBJ	LINE	SOURCE			
07AC D2E0	1612	SHL	AL,CL		; ADJUST
07AE 2AE4	1613	SUB	AH,AH		
07B0 C3	1614	RET			
07B1	1615	SM2_OFFS_ERR:			
07B1 F9	1616	STC			
07B2 C3	1617	RET			
	1618	SM2_OFFS	ENDP		
	1619				
07B3 30382F31362F38	1620	DB	'00/16/82'		; RELEASE MARKER
32					
	1621				
07BB	1622	END_ADDRESS	LABEL	BYTE	
----	1623	CODE	ENDS		
	1624	END			

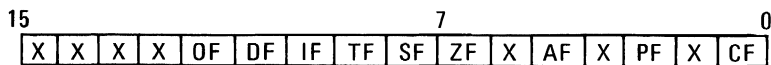
Notes:

APPENDIX B: 8088 ASSEMBLY INSTRUCTION SET REFERENCE

8088 Register Model

AX:	AH	AL	Accumulator	} General Register File
BX:	BH	BL	Base	
CX:	CH	CL	Count	
DX:	DH	DL	Data	
	SP		Stack Pointer	
	BP		Base Pointer	
	SI		Source Index	
	DI		Destination Index	
	IP		Instruction Pointer	
	FLAGSH	FLAGSL	Status Flags	
	CS		Code Segment	} Segment Register File
	DS		Data Segment	
	SS		Stack Segment	
	ES		Extra Segment	

Instructions which reference the flag register file as a 16-bit object use the symbol **FLAGS** to represent the file:



X = Don't Care

AF:	Auxiliary Carry - BCD	} 8080 Flags
CF:	Carry Flag	
PF:	Parity Flag	
SF:	Sign Flag	
ZF:	Zero Flag	
DF:	Direction Flag (Strings)	} 8088 Flags
IF:	Interrupt Enable Flag	
OF:	Overflow Flag (CF ⊕ SF)	
TF:	Trap - Single Step Flag	

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
-----	-----	-----

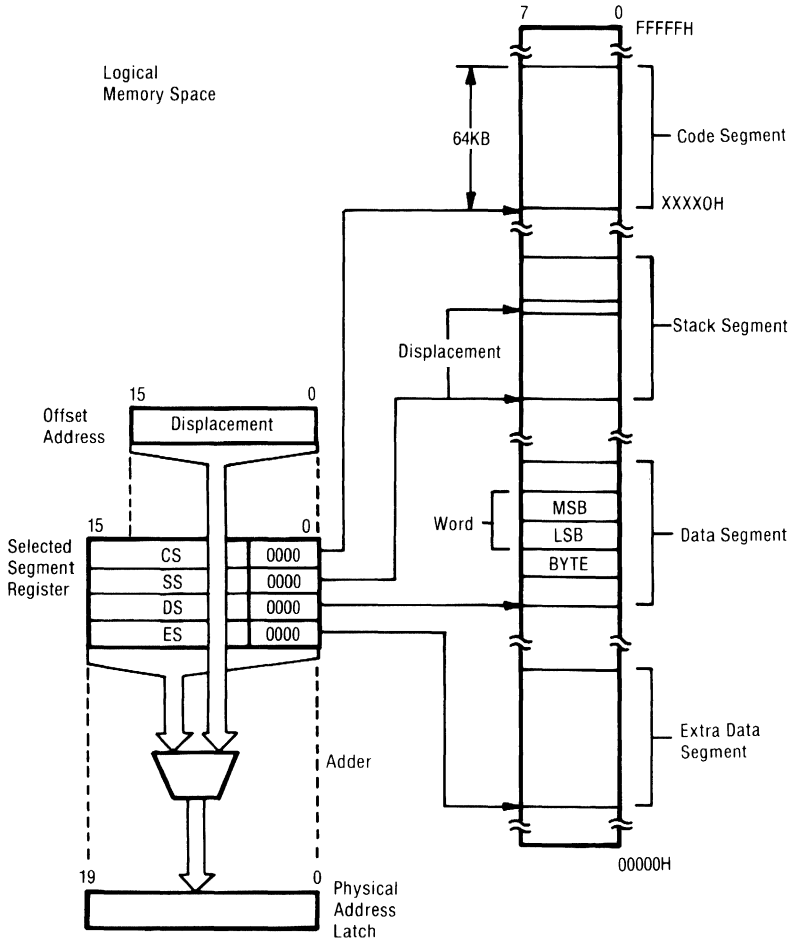
mod	Displacement
00	DISP = 0*, disp-low and disp-high are absent
01	DISP = disp-low sign-extended to 16-bits, disp-high is absent
10	DISP = disp-high: disp-low
11	r/m is treated as a “reg” field

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

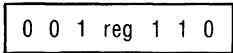
DISP follows 2nd byte of instruction (before data if required).

*except if mod = 00 and r/m = 110 then EA = disp-high: disp-low.

Memory Segmentation Model



Segment Override Prefix



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 and 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 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 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 0

INTO = 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

LO								
HI	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	JO	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 AX	MOV AL m	MOV AX 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	MOV 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	PCP r/m
9	CBW	CWD	CALL l,d	WAIT	PUSHF	POPF	SAHF	LAHF
A	TEST b,i	TEST w,i	STOS b	STOS w	LODS b	LODS 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

Appendix B

where:

mod □ 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 COLOR

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, Ctrl ←		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	←	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
28	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

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	ü	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	å	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	ì	Alt 141	Note 6	Black	Light Magenta	High Intensity
8E	142	À	Alt 142	Note 6	Black	Yellow	High Intensity
8F	143	Å	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	ò	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	ó	Alt 153	Note 6	Blue	Light Blue	High Intensity Underline
9A	154	ū	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
FO	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.)

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	l	k	{
12	C	♀	└	,	<	L	\	l	!
13	D	🎵	↔	—	=	M	l	m	}
14	E	🎵	▲	.	>	N	^	n	~
15	F	☀	▼	/	?	O	_	o	△

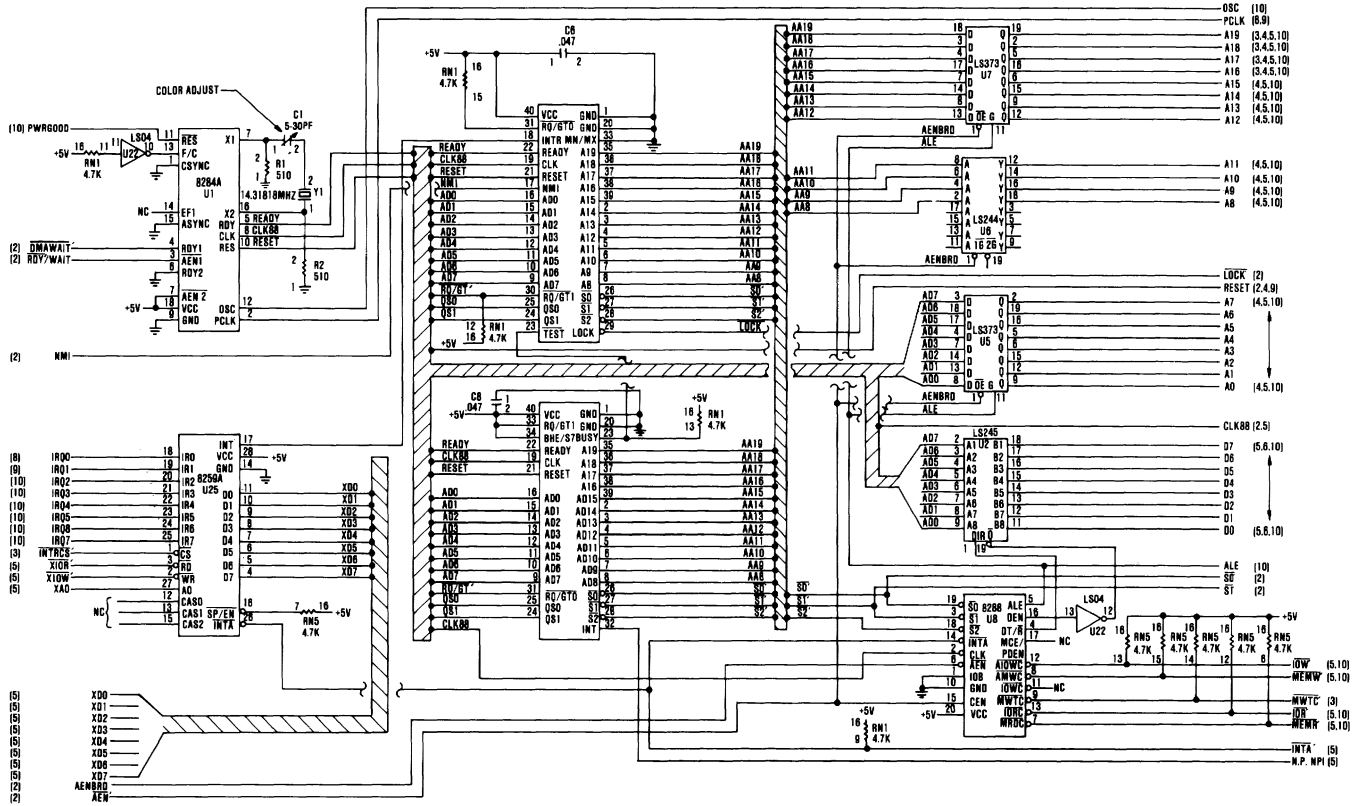
DECIMAL VALUE	➡	128	144	160	176	192	208	224	240
➡	HEXA DECIMAL VALUE	8	9	A	B	C	D	E	F
0	0	Ç	É	á	⋮			∞	≡
1	1	ü	æ	í	⋮			β	±
2	2	é	Æ	ó	⋮			Γ	≥
3	3	â	ô	ú				π	≤
4	4	ä	ö	ñ				Σ	∫
5	5	à	ò	Ñ				σ	∫
6	6	å	û	ä				μ	÷
7	7	ç	ù	ó				τ	≈
8	8	ê	ÿ	ı				ϕ	◦
9	9	ë	Ö	└				θ	•
10	A	è	Ü	└				Ω	•
11	B	ï	ç	½				δ	√
12	C	î	£	¼				∞	n
13	D	ì	¥	ı				φ	2
14	E	Ä	℞	«				€	■
15	F	Å	ƒ	»				∩	BLANK 'FF'

Notes:

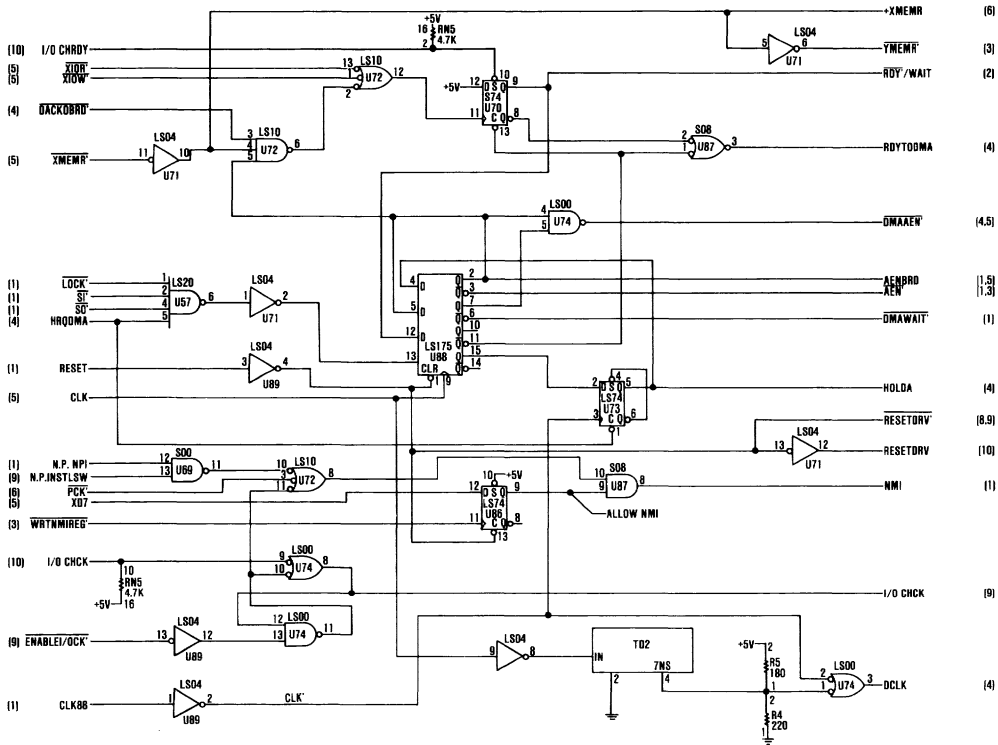
APPENDIX D: LOGIC DIAGRAMS

System Board	D-2
Keyboard – Type 1	D-12
Keyboard – Type 2	D-14
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-45
5–1/4 Inch Diskette Drive – Type 1	D-49
5–1/4 Inch Diskette Drive – Type 2	D-52
Fixed Disk Drive Adapter	D-54
Fixed Disk Drive – Type 1	D-60
Fixed Disk Drive – Type 2	D-63
32K Memory Expansion Option	D-66
64K Memory Expansion Option	D-69
64/256K Memory Expansion Option	D-72
Game Control Adapter	D-76
Prototype Card	D-77
Asynchronous Communications Adapter	D-78
Binary Synchronous Communications Adapter	D-79
SDLC Communications Adapter	D-81

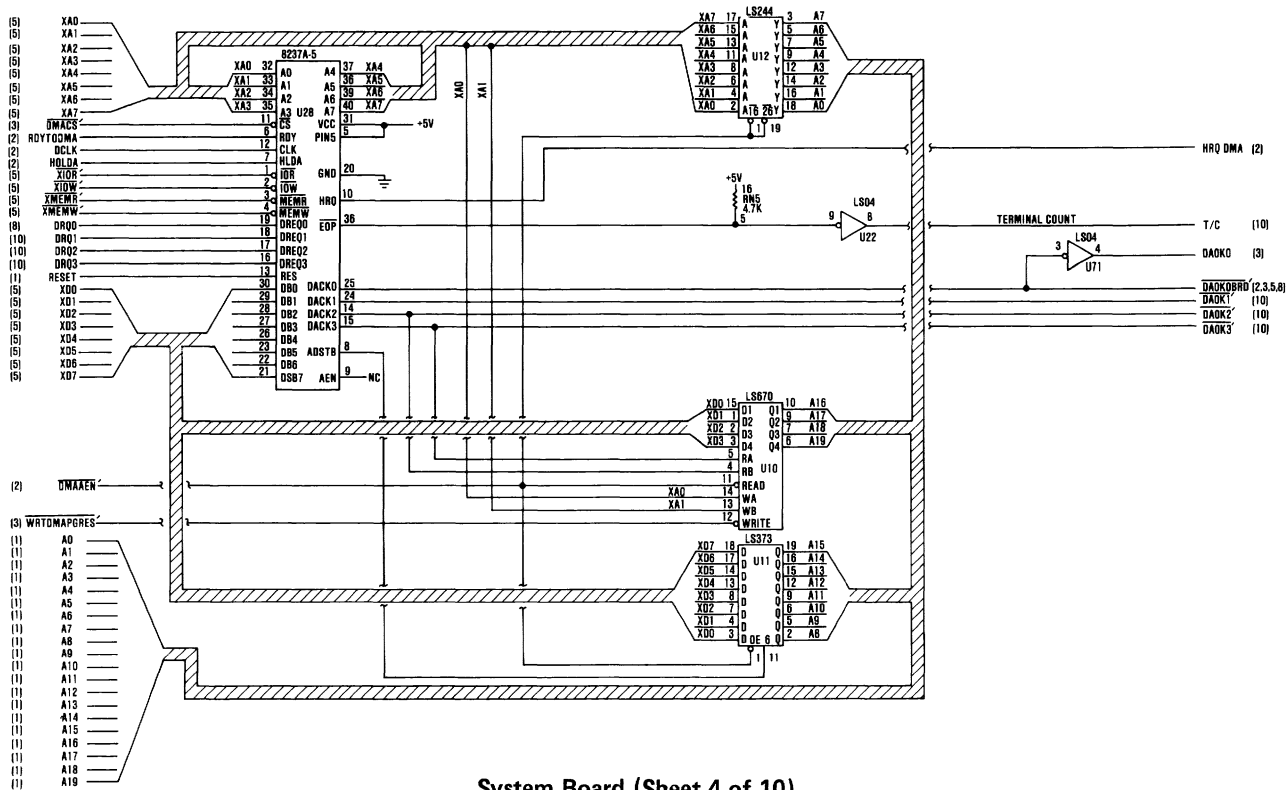
D-2 Logic Diagrams



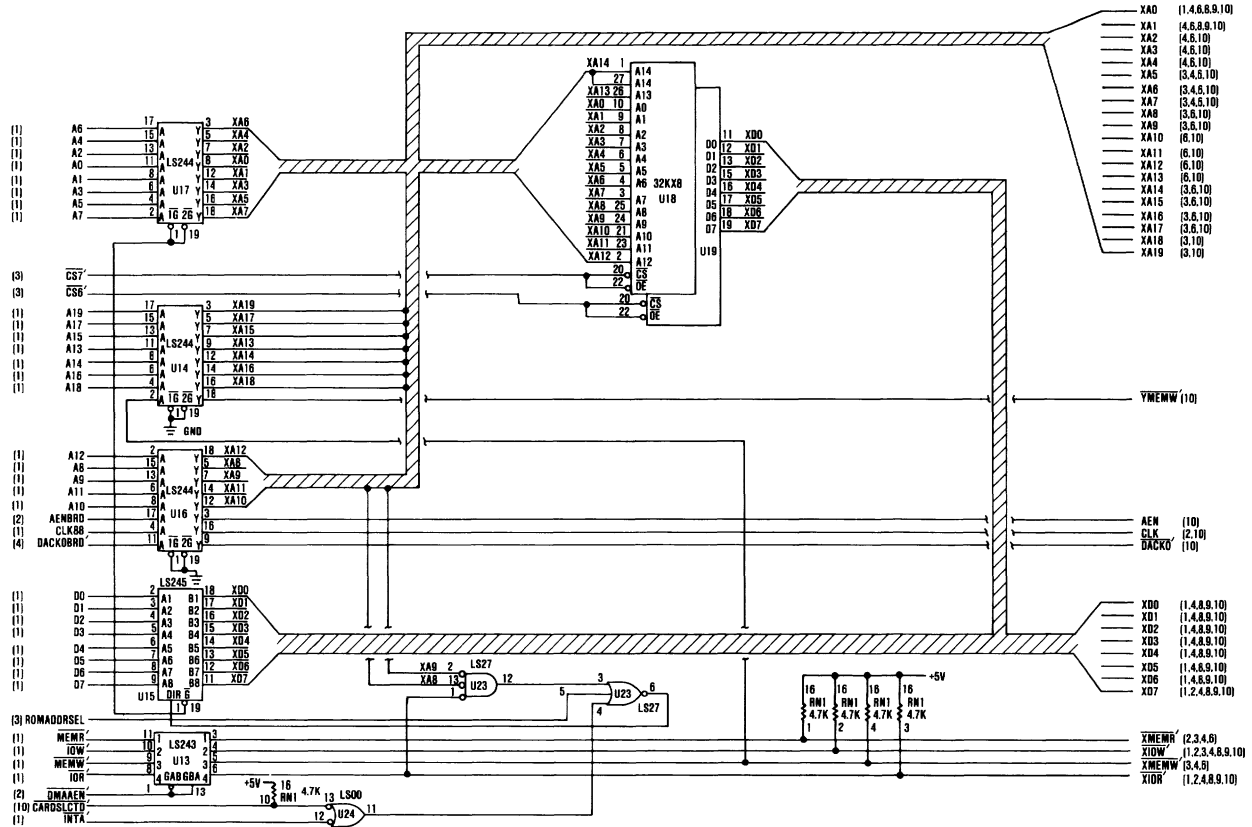
System Board (Sheet 1 of 10)



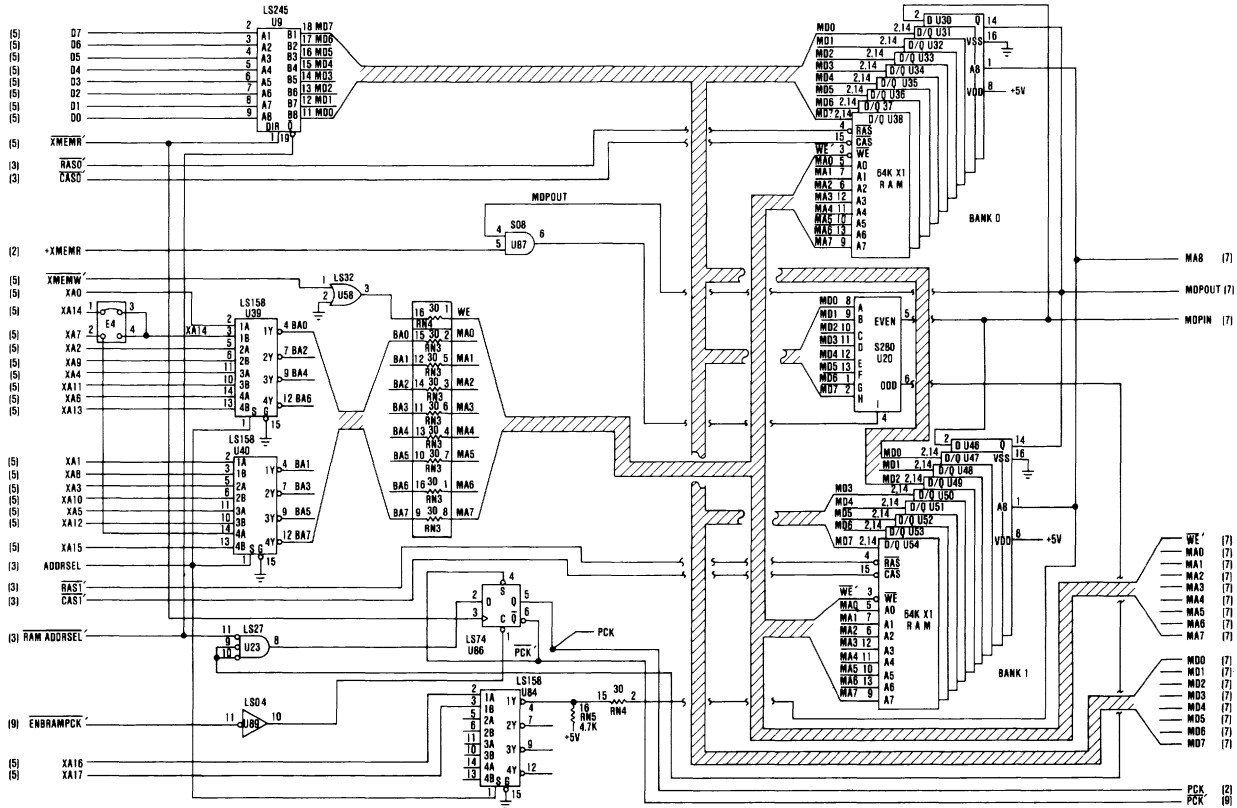
System Board (Sheet 2 of 10)



System Board (Sheet 4 of 10)

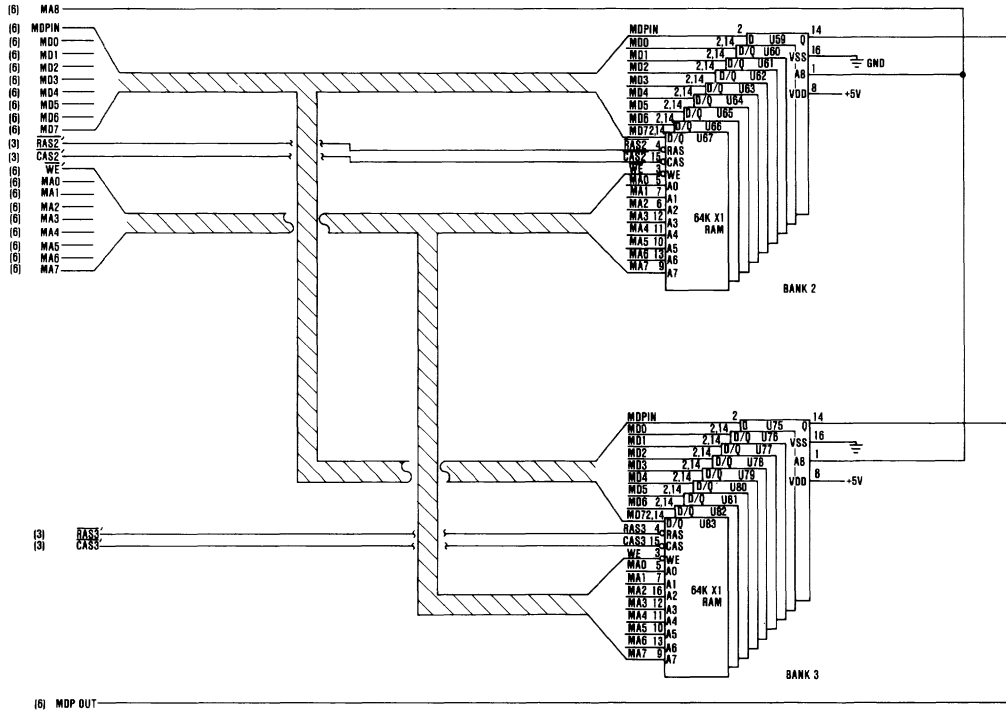


System Board (Sheet 5 of 10)

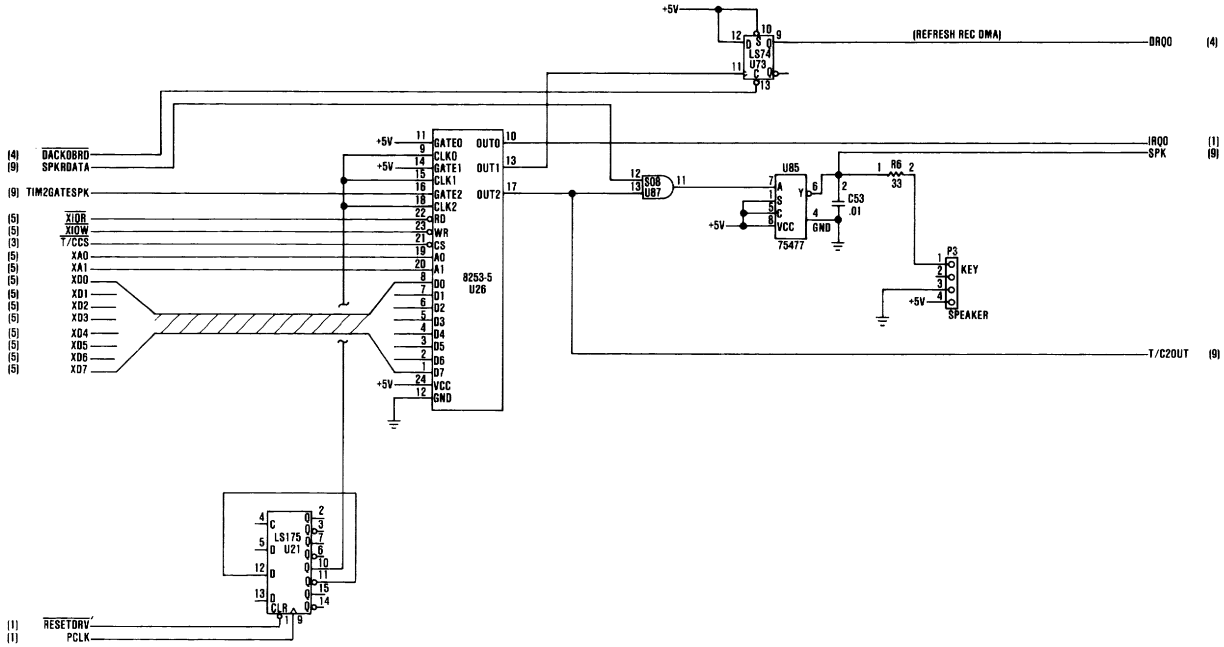


System Board (Sheet 6 of 10)

D-8 Logic Diagrams

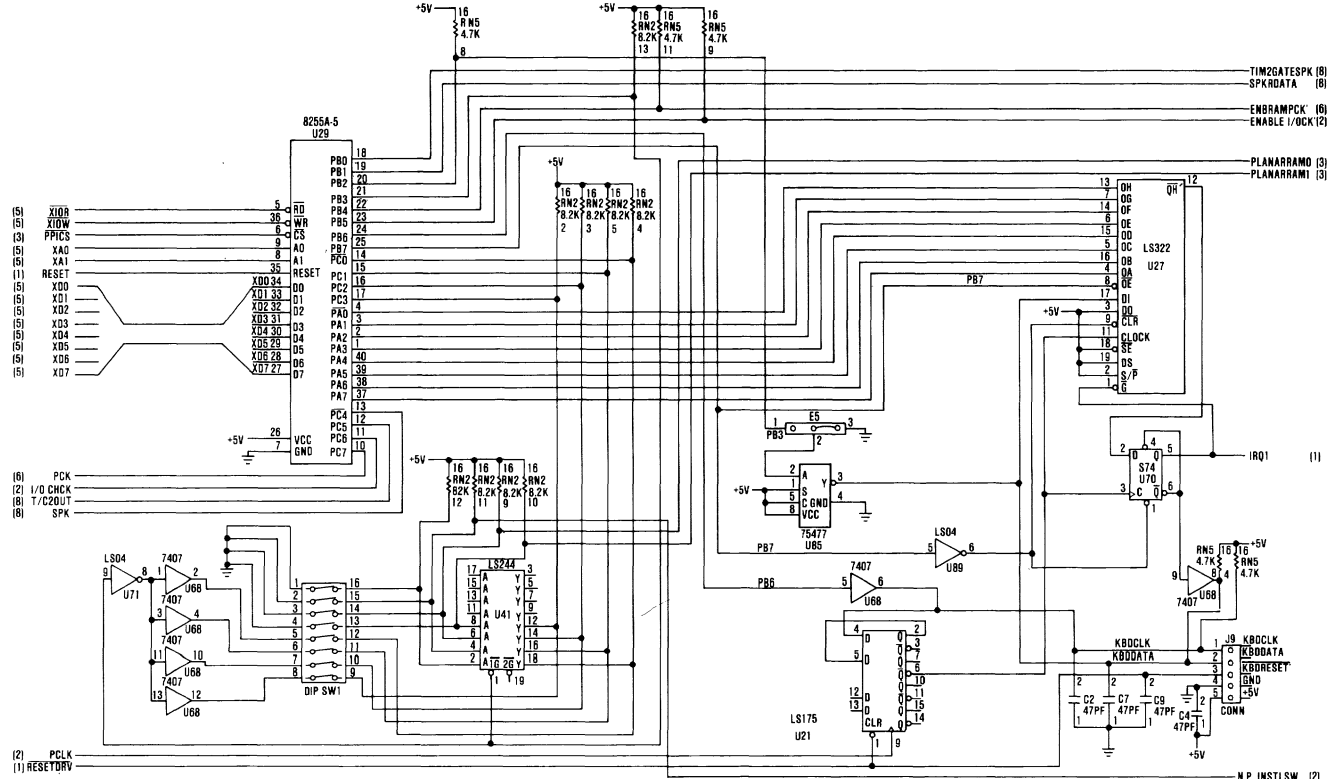


System Board (Sheet 7 of 10)

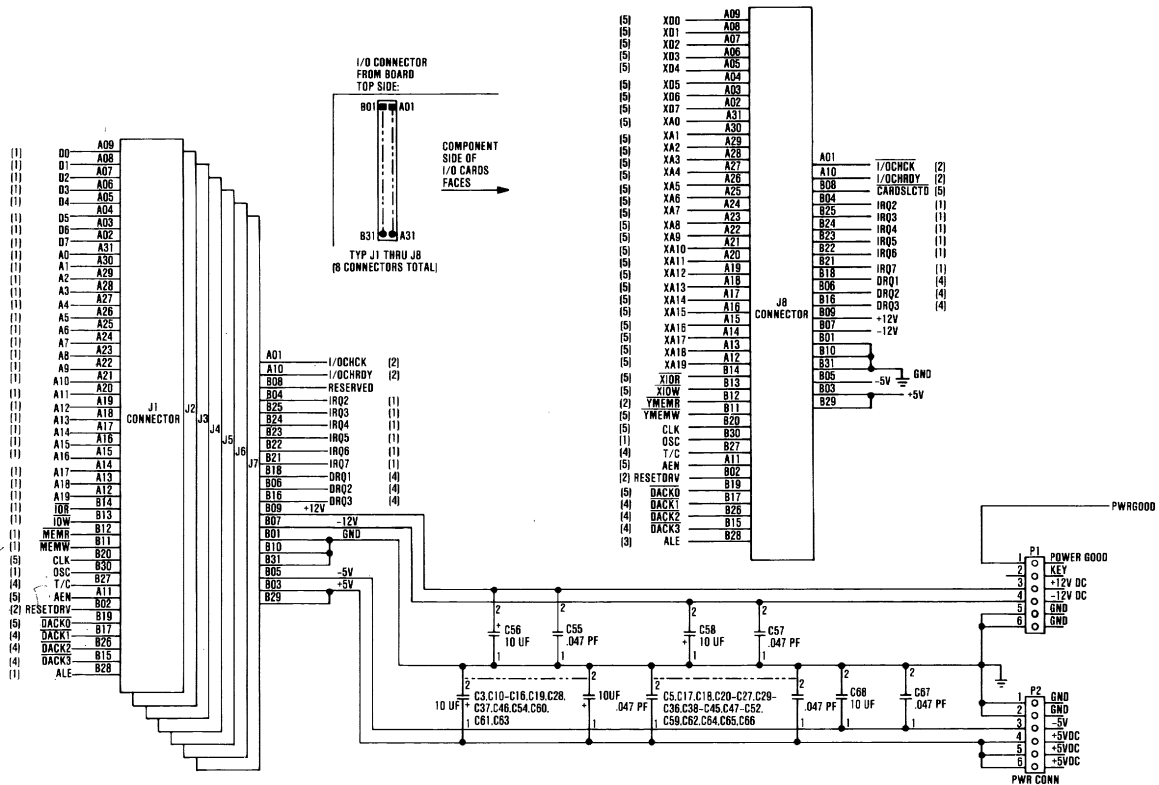


System Board (Sheet 8 of 10)

D-10 Logic Diagrams

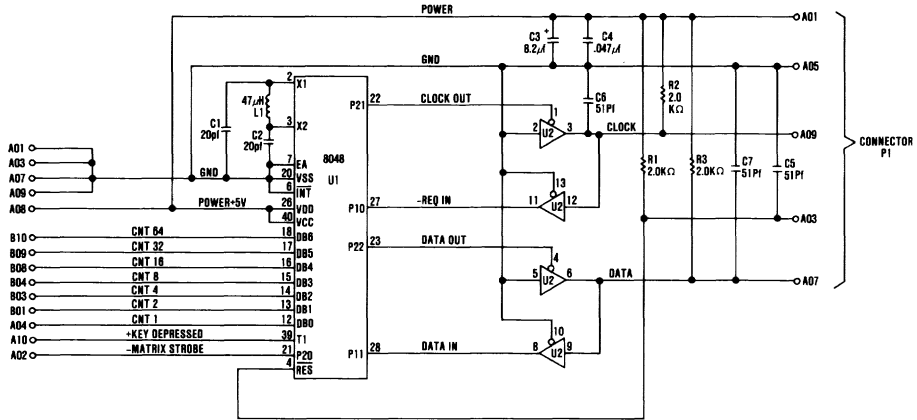


System Board (Sheet 9 of 10)

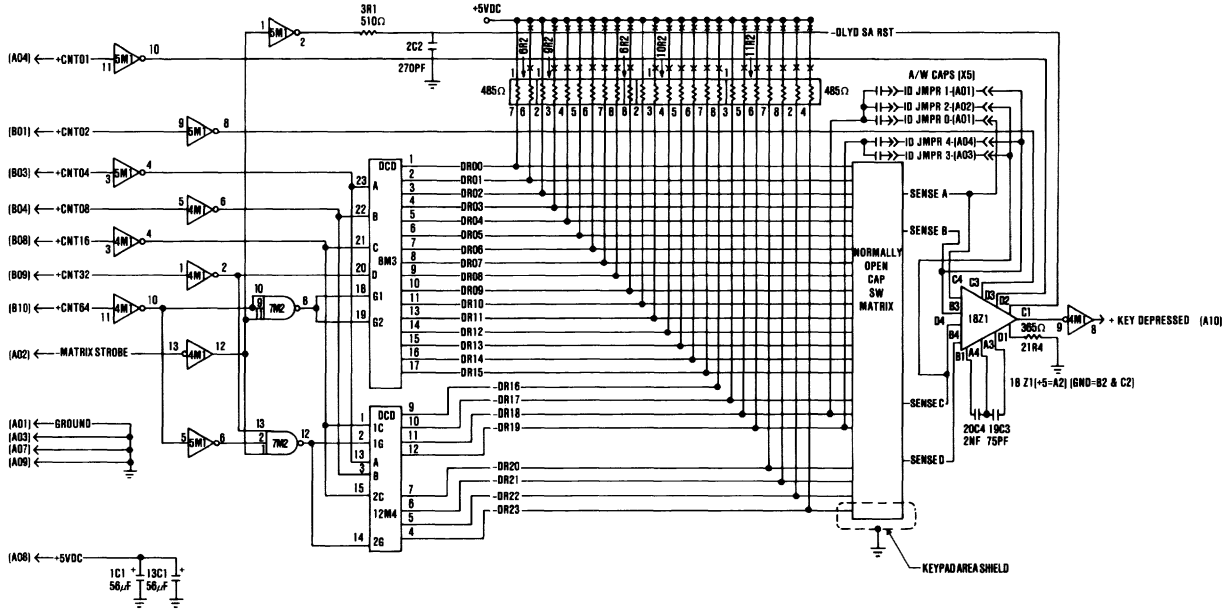


System Board (Sheet 10 of 10)

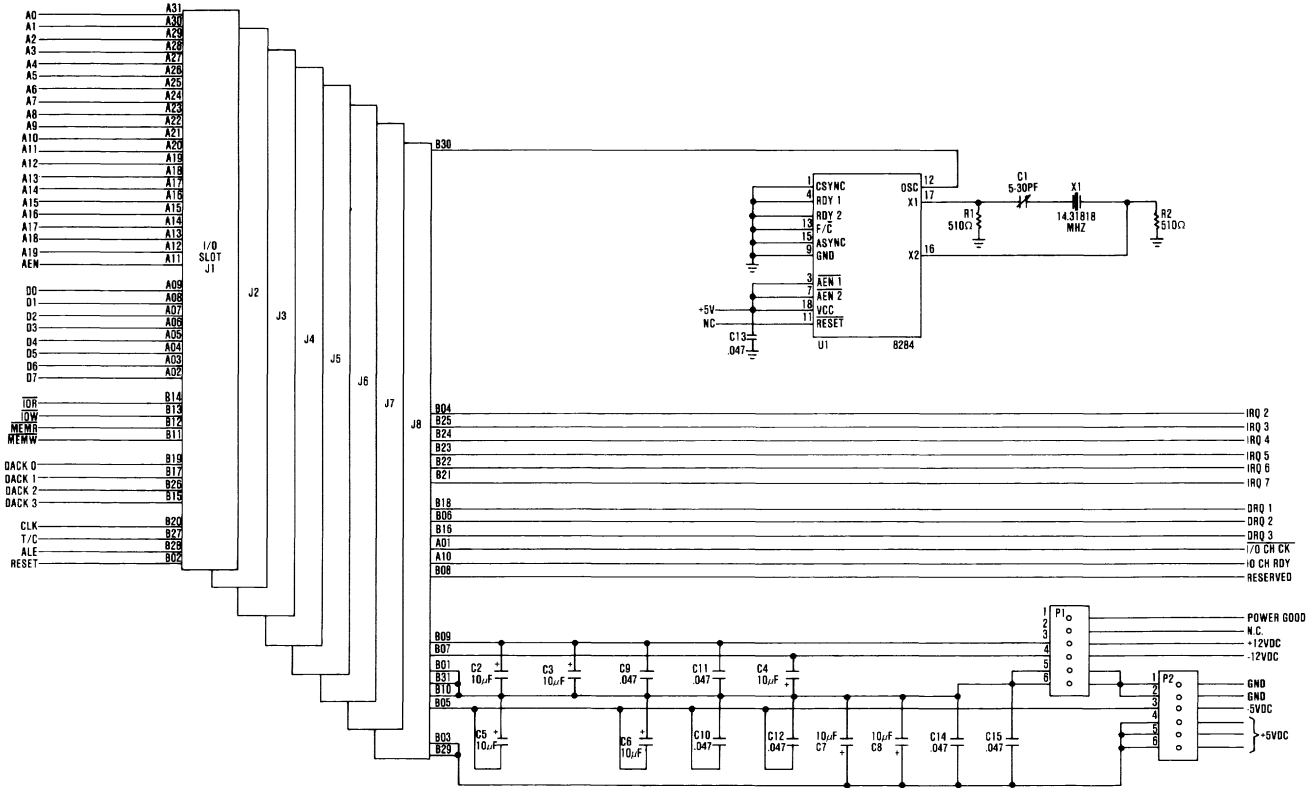
D-12 Logic Diagrams



Keyboard - Type 1 (Sheet 1 of 2)

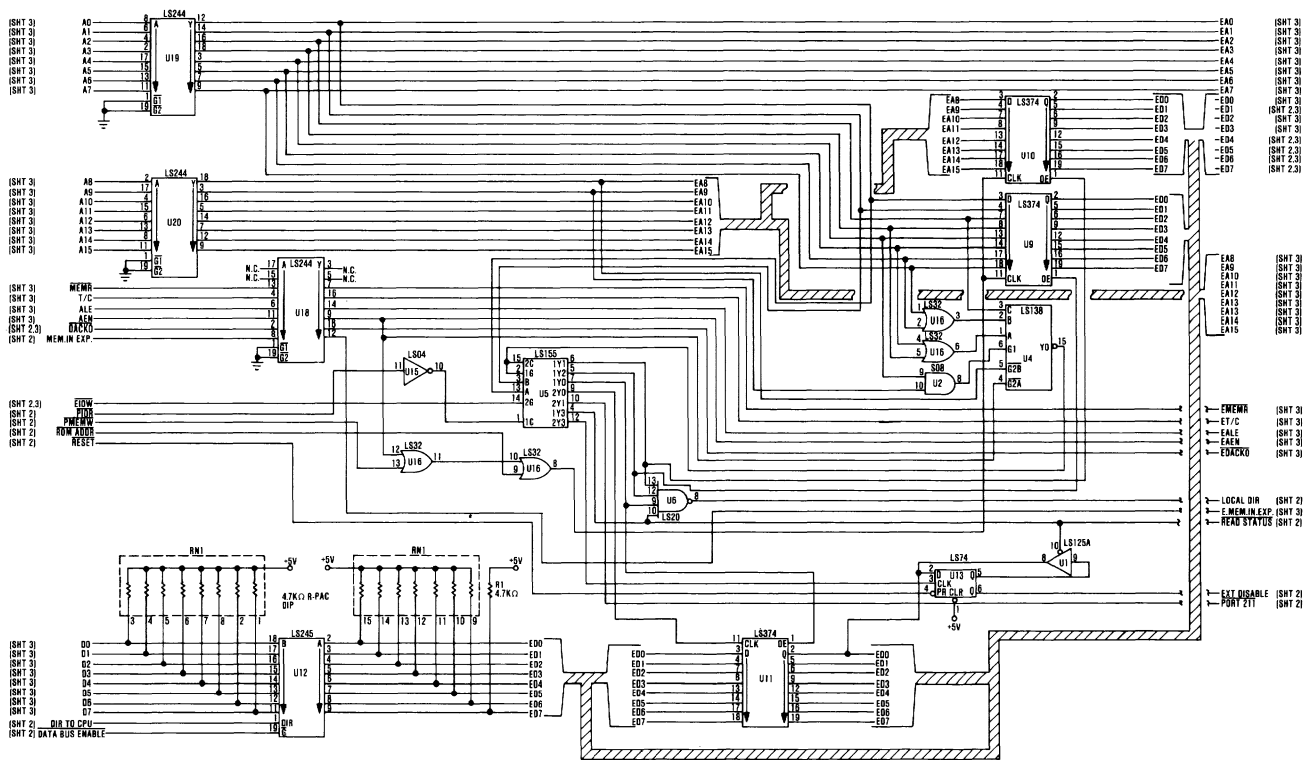


Keyboard - Type 1 (Sheet 2 of 2)

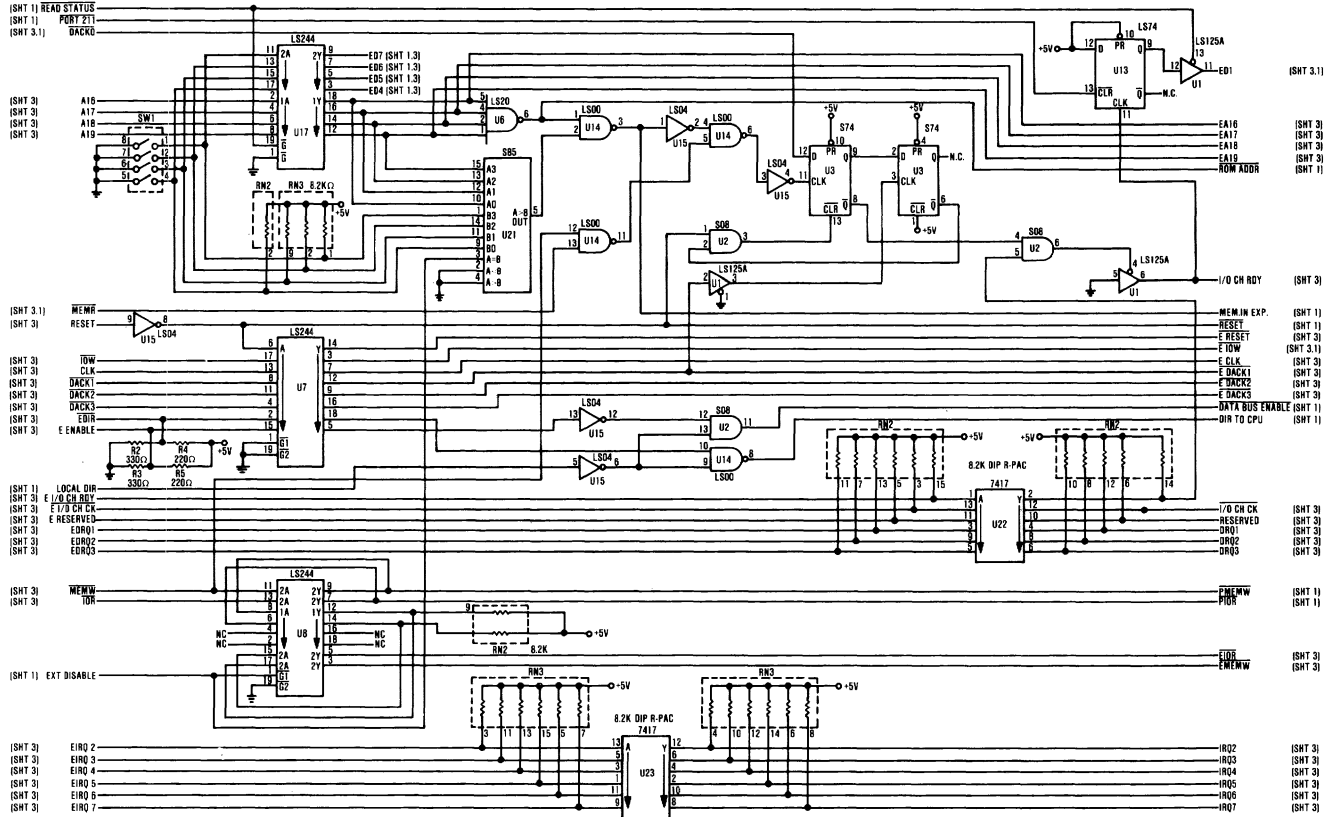


Expansion Board (Sheet 1 of 1)

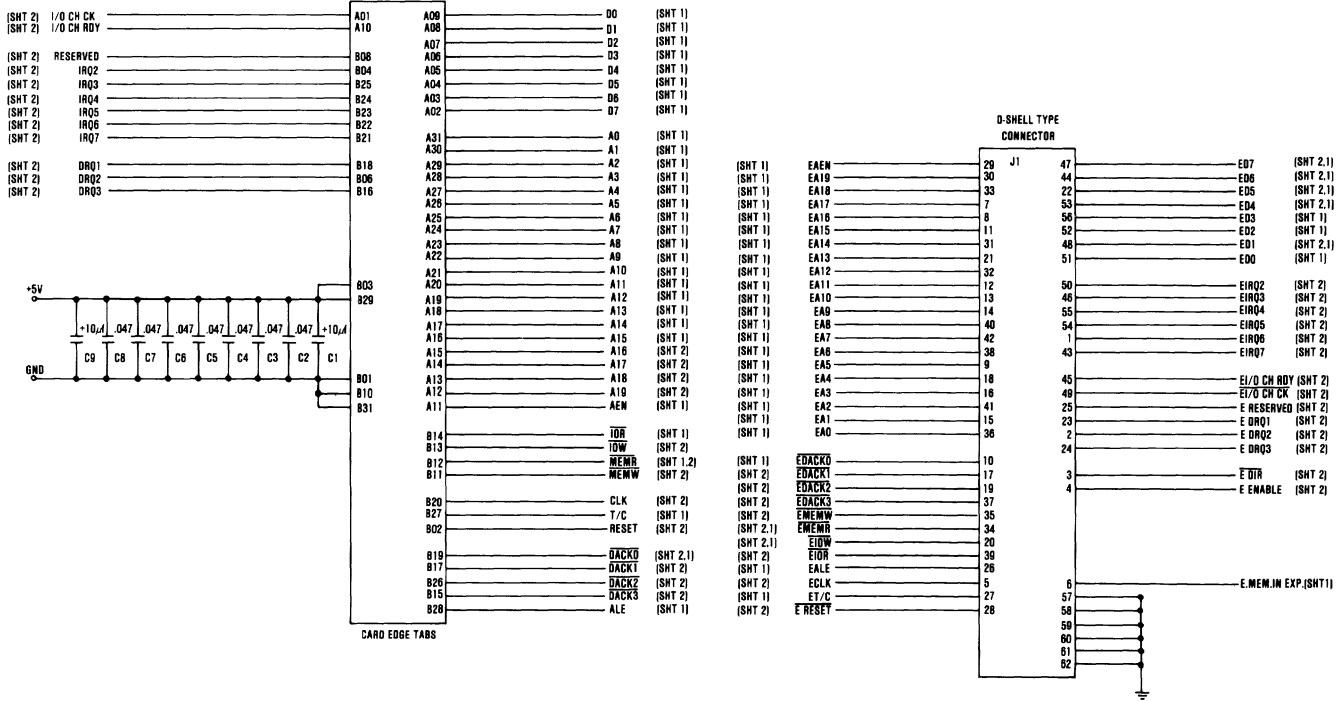
D-16 Logic Diagrams



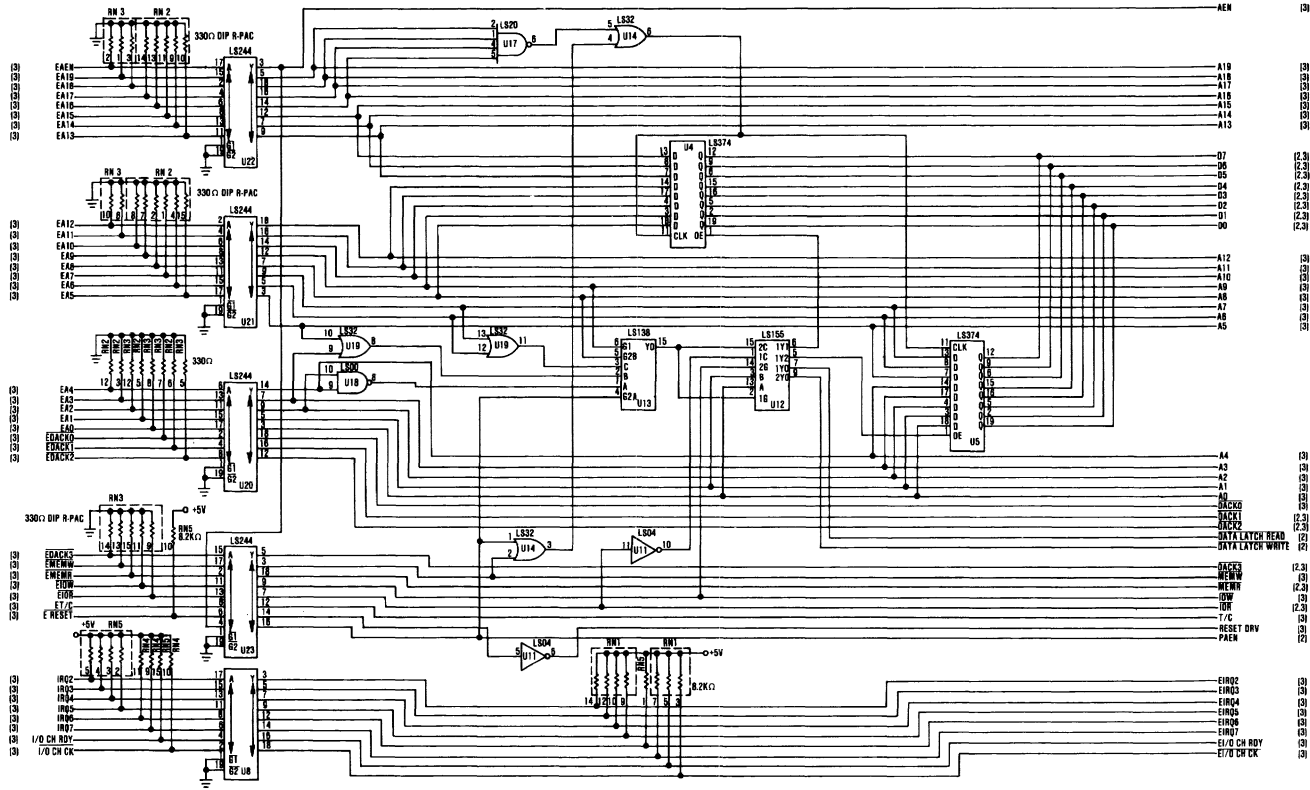
Extender Card (Sheet 1 of 3)



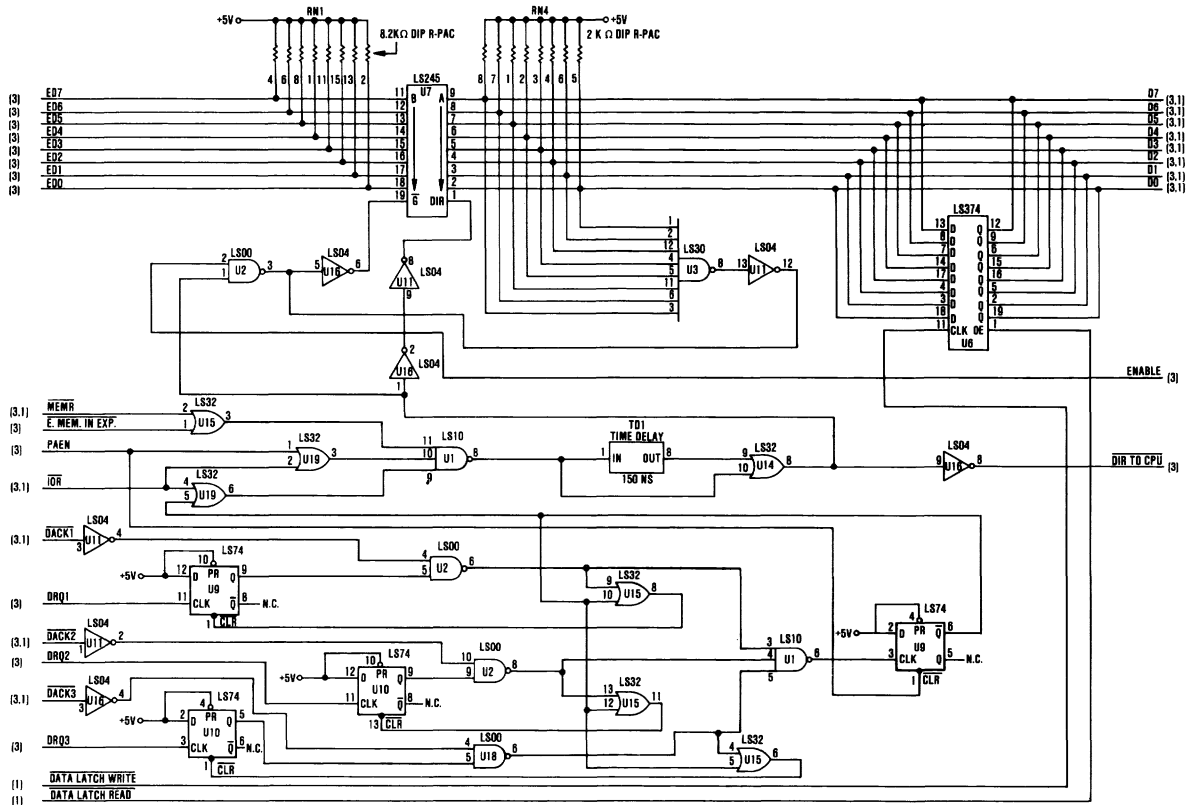
Extender Card (Sheet 2 of 3)



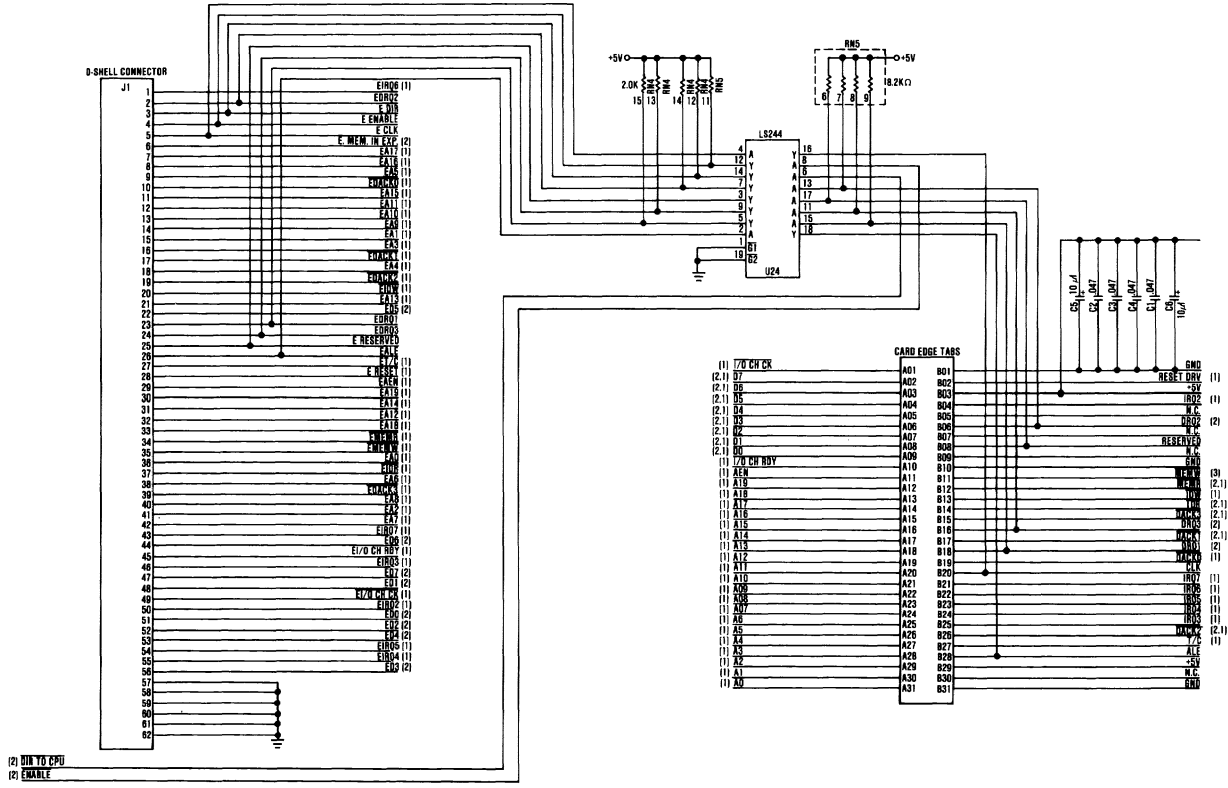
Extender Card (Sheet 3 of 3)



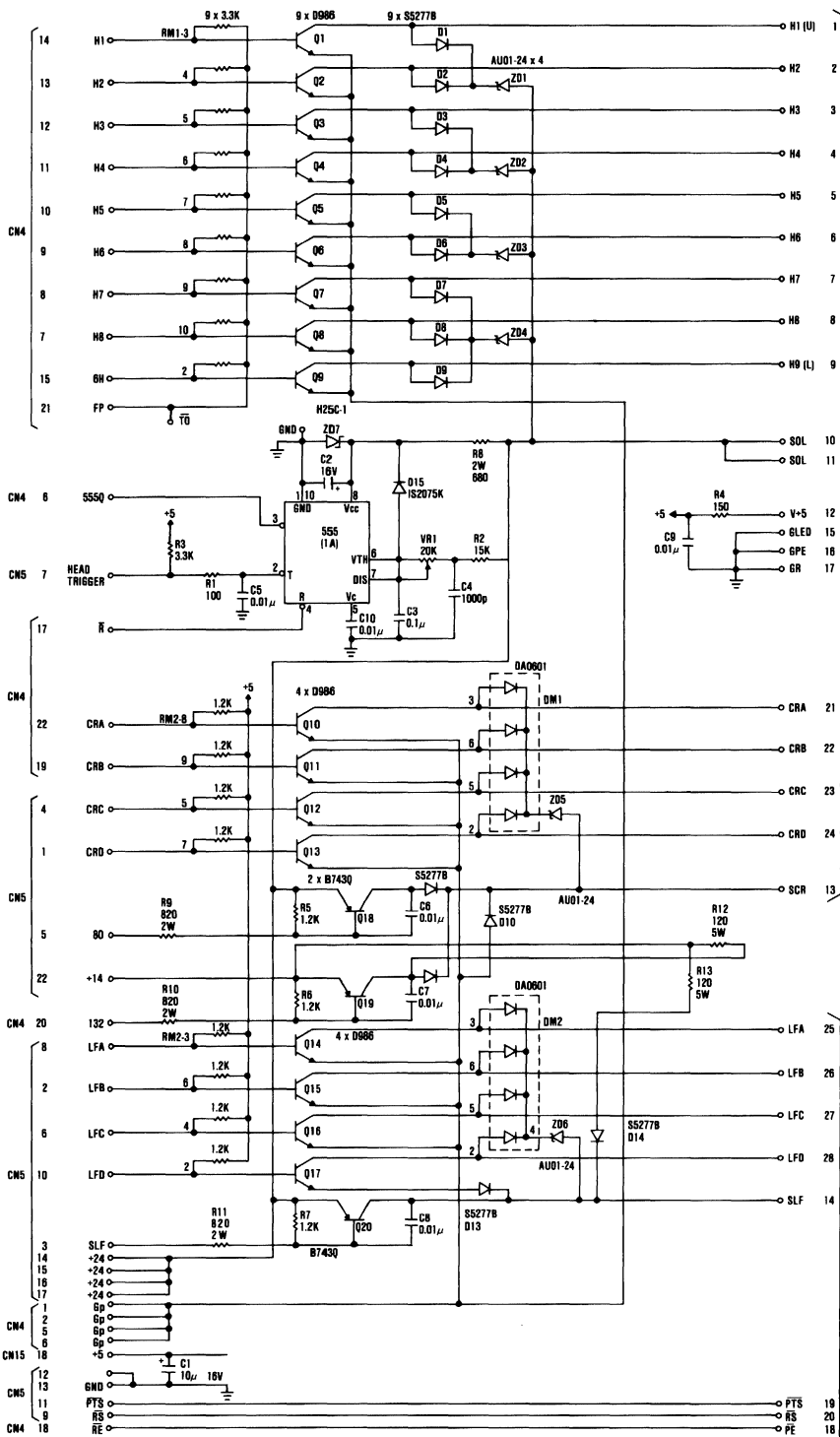
Receiver Card (Sheet 1 of 3)

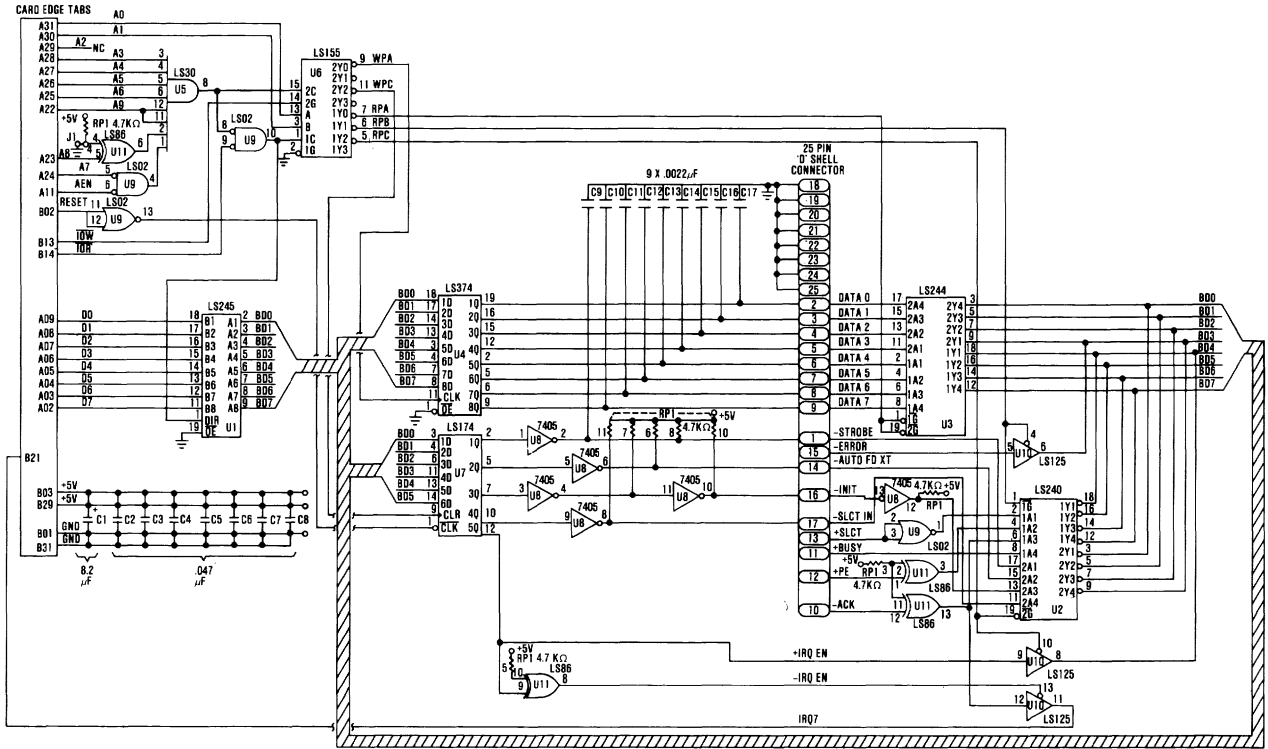


Receiver Card (Sheet 2 of 3)

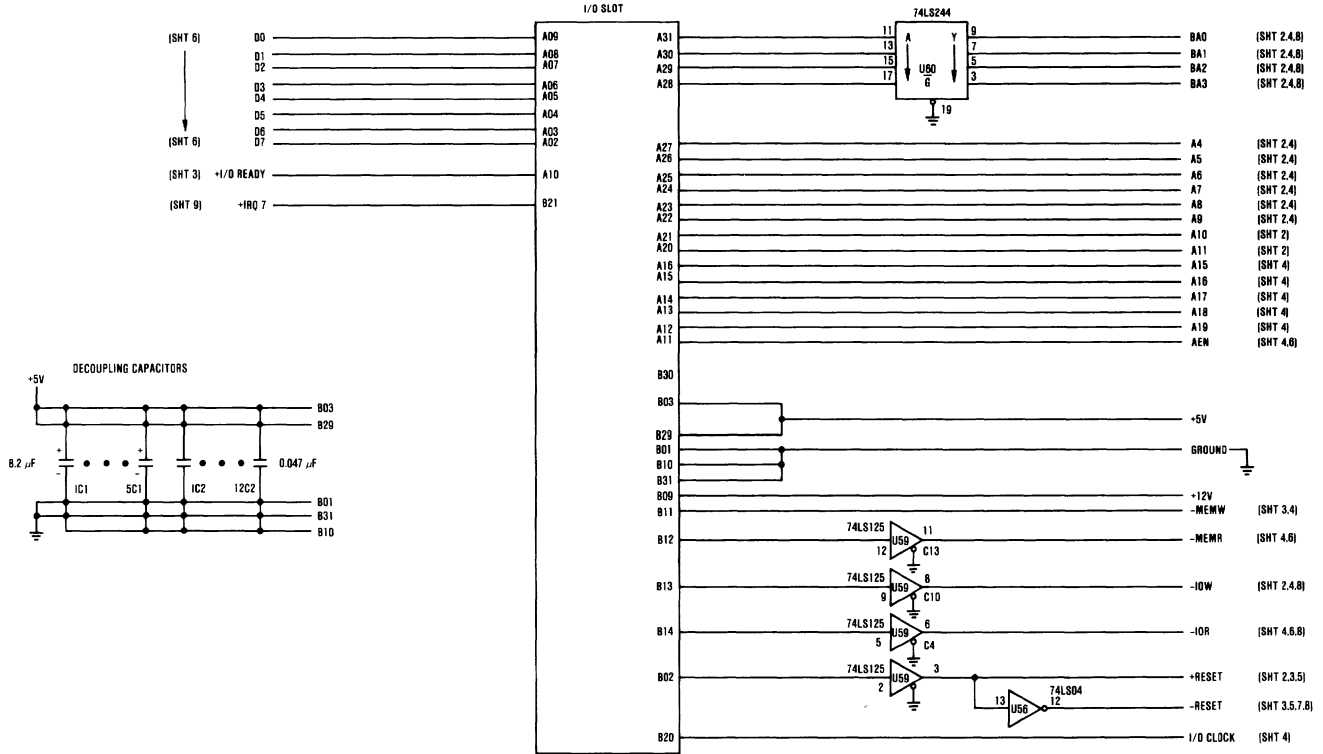


Receiver Card (Sheet 3 of 3)

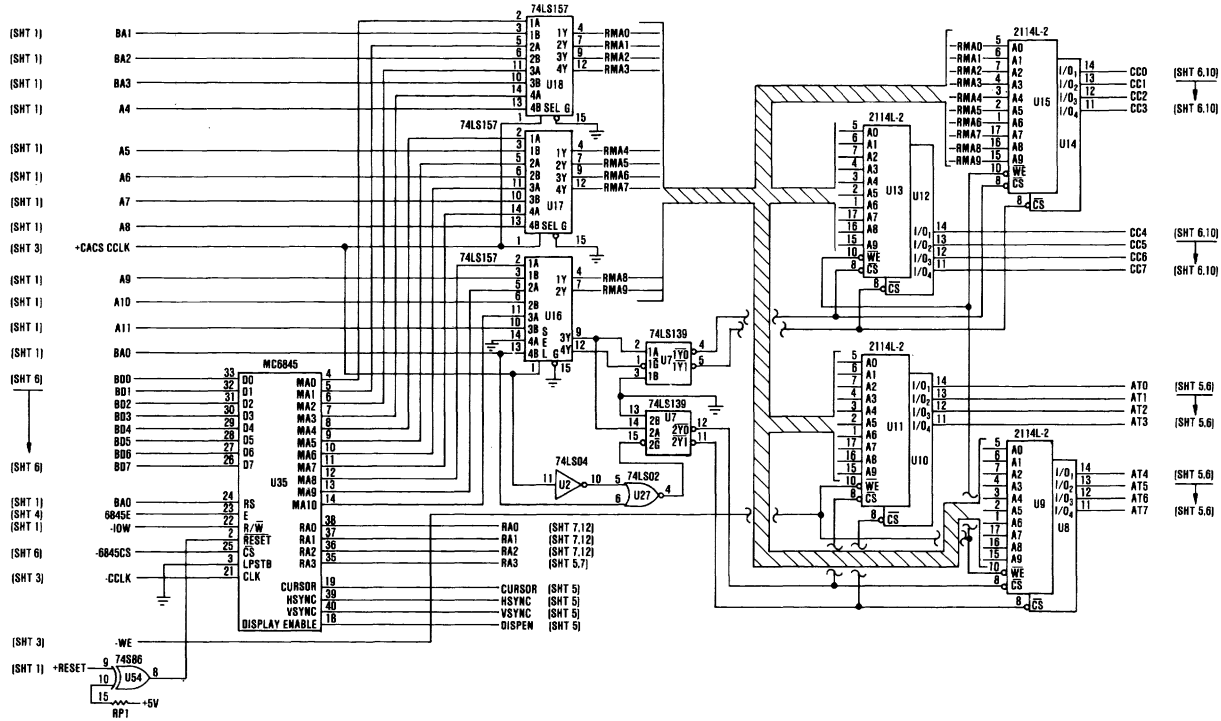




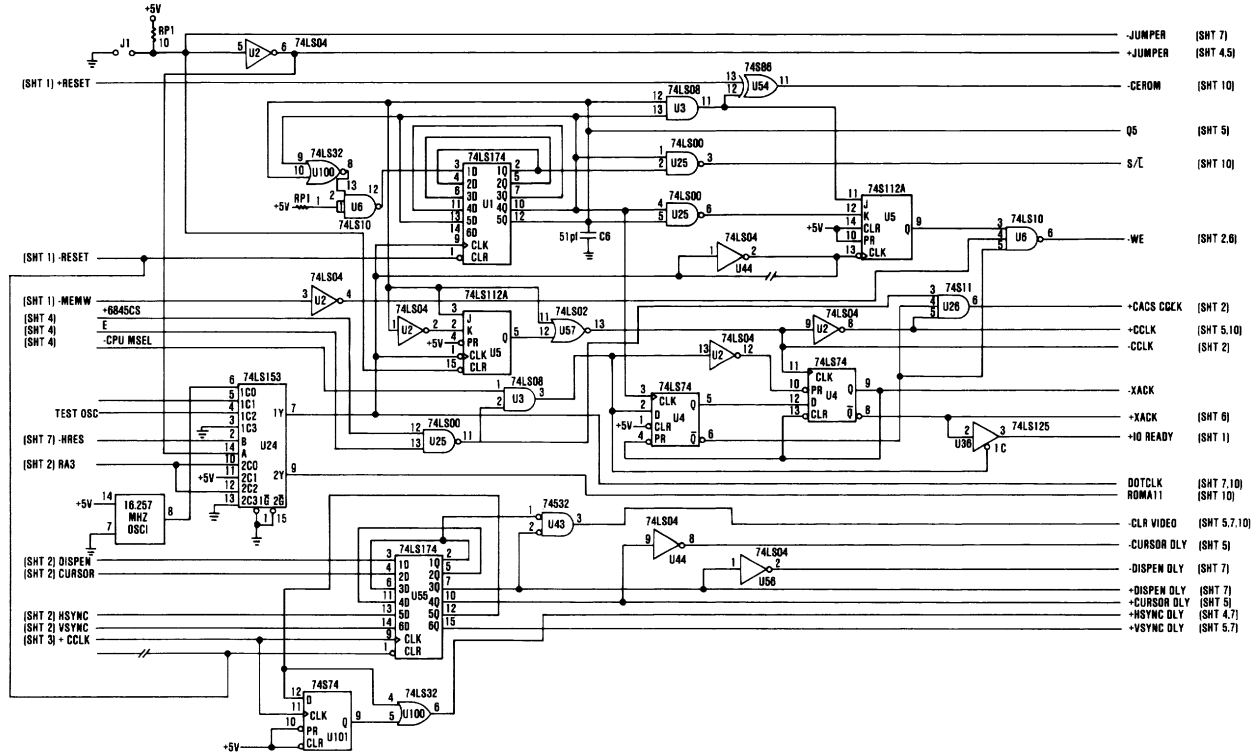
Printer Adapter (Sheet 1 of 1)



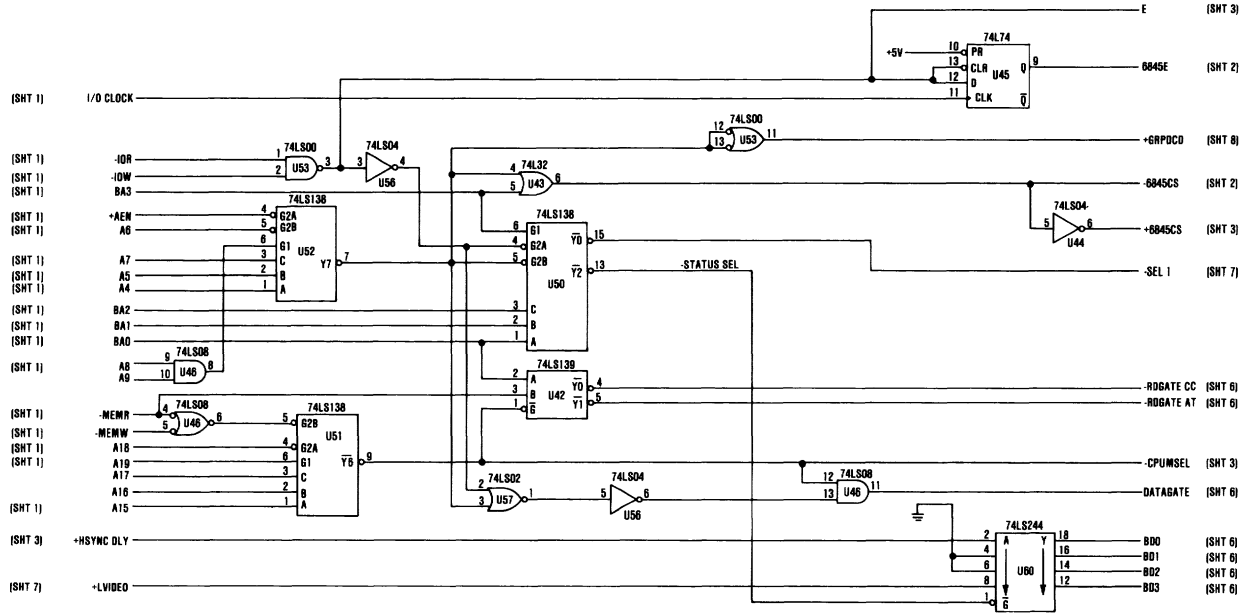
Monochrome Display Adapter (Sheet 1 of 10)



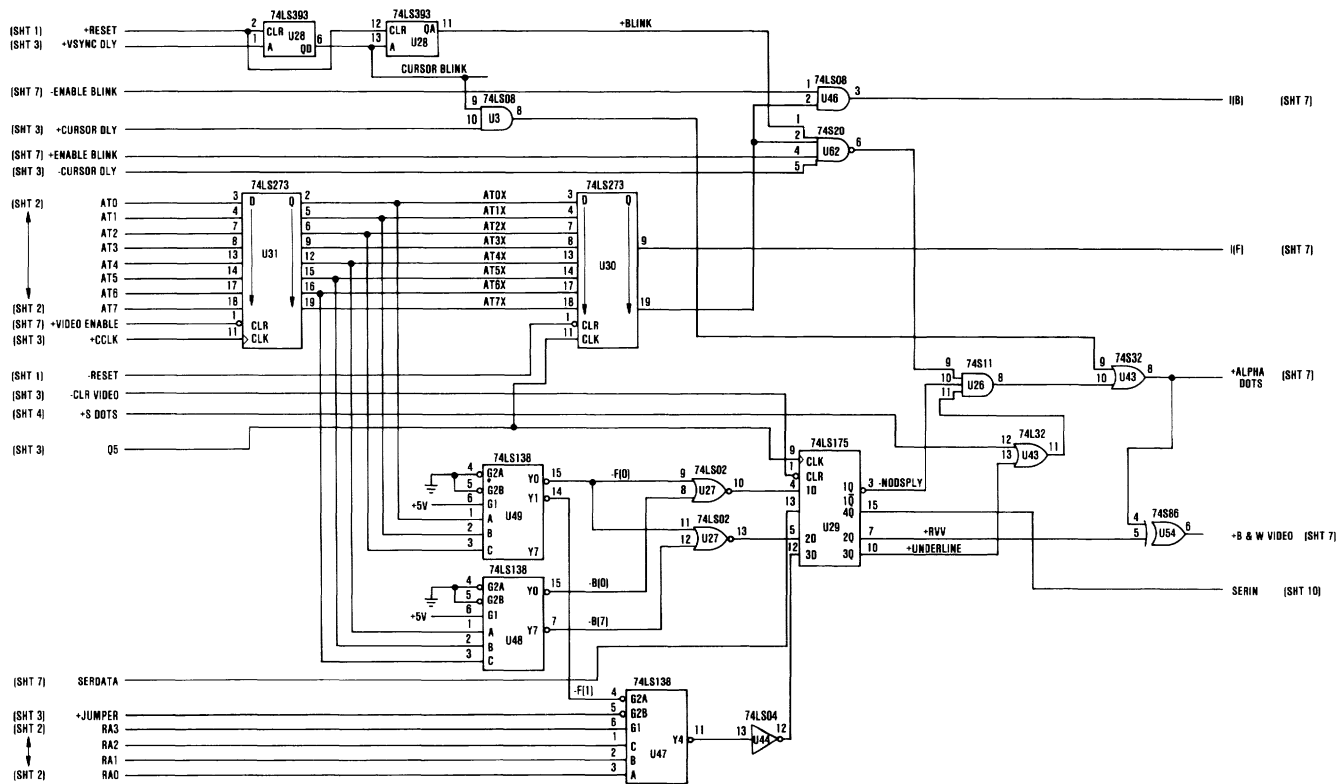
Monochrome Display Adapter (Sheet 2 of 10)



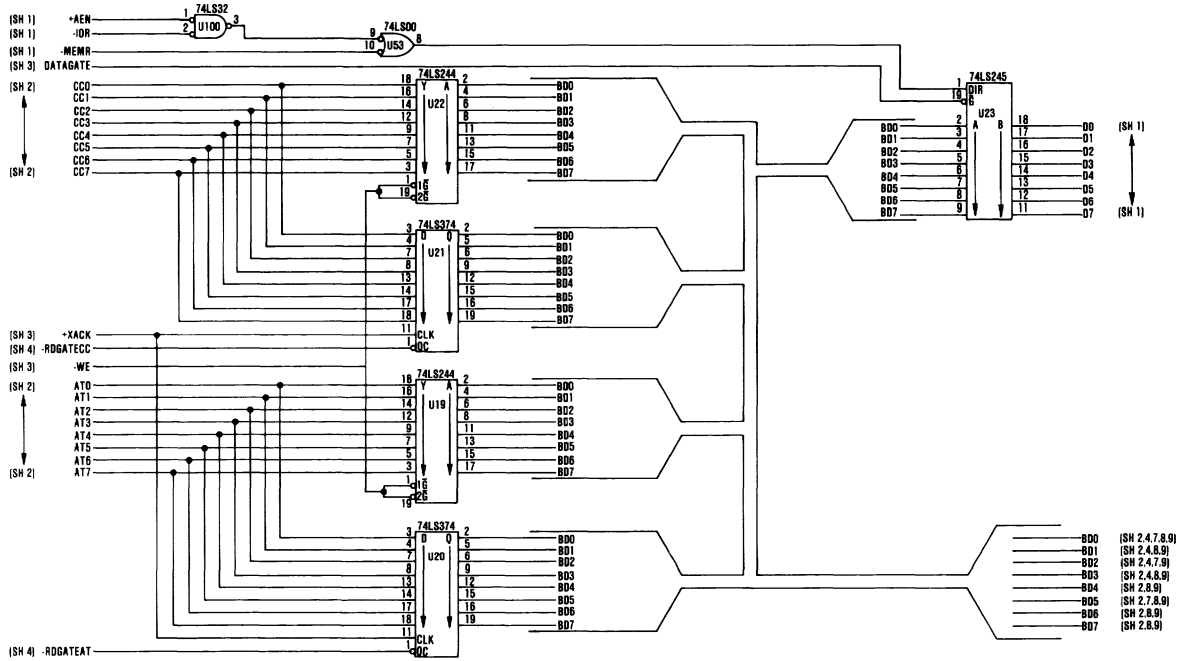
Monochrome Display Adapter (Sheet 3 of 10)



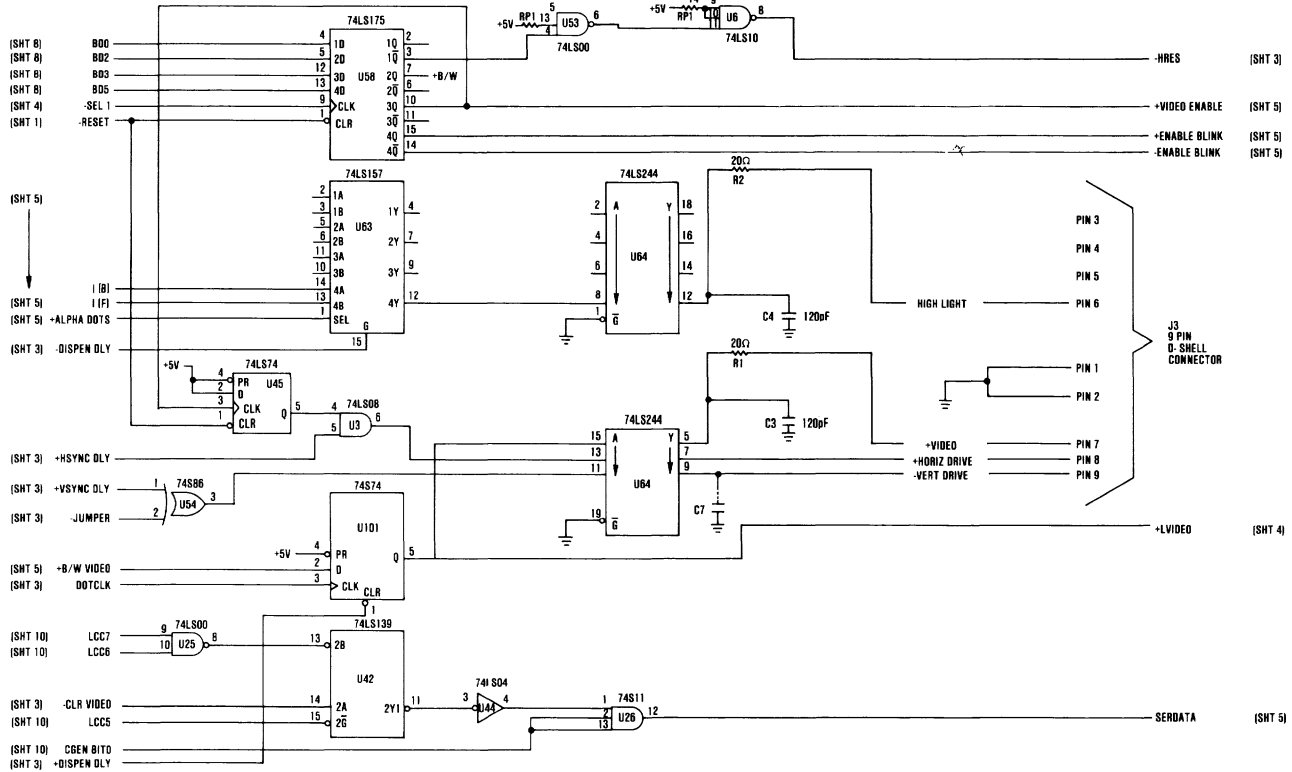
Monochrome Display Adapter (Sheet 4 of 10)



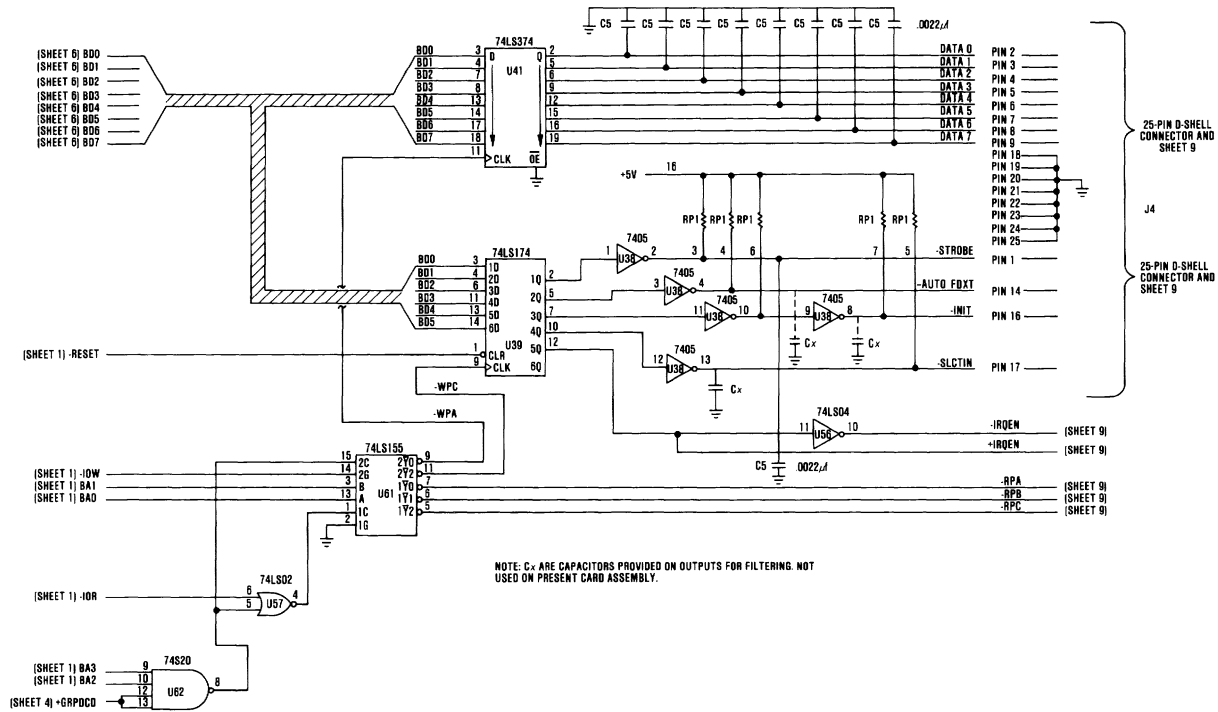
Monochrome Display Adapter (Sheet 5 of 10)



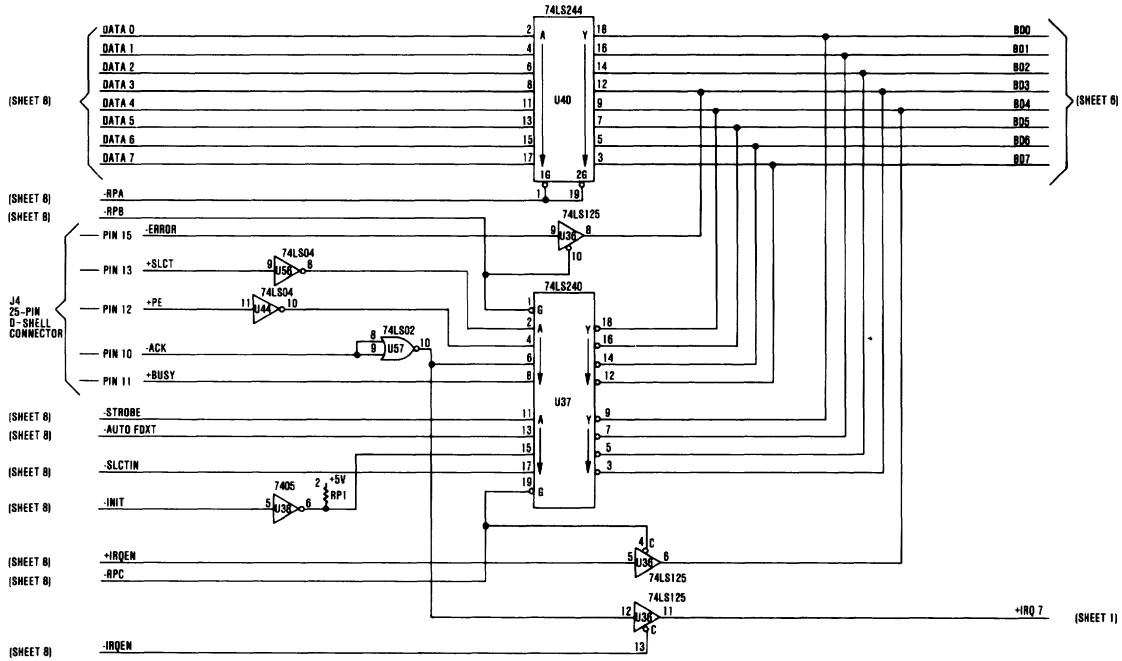
Monochrome Display Adapter (Sheet 6 of 10)



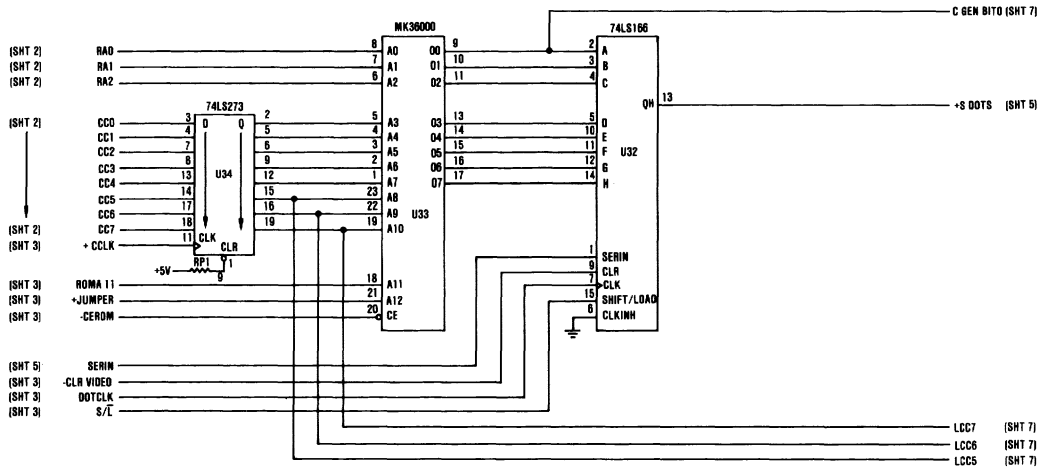
Monochrome Display Adapter (Sheet 7 of 10)



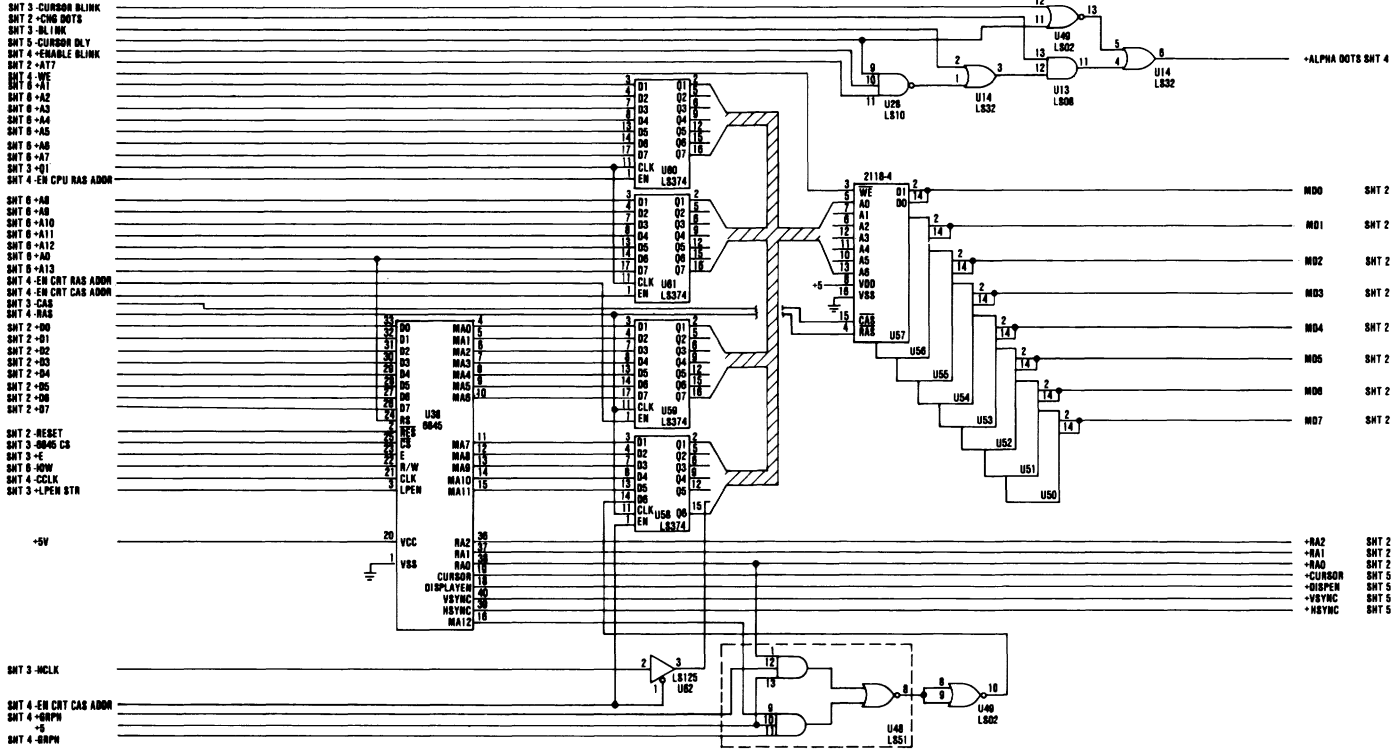
Monochrome Display Adapter (Sheet 8 of 10)



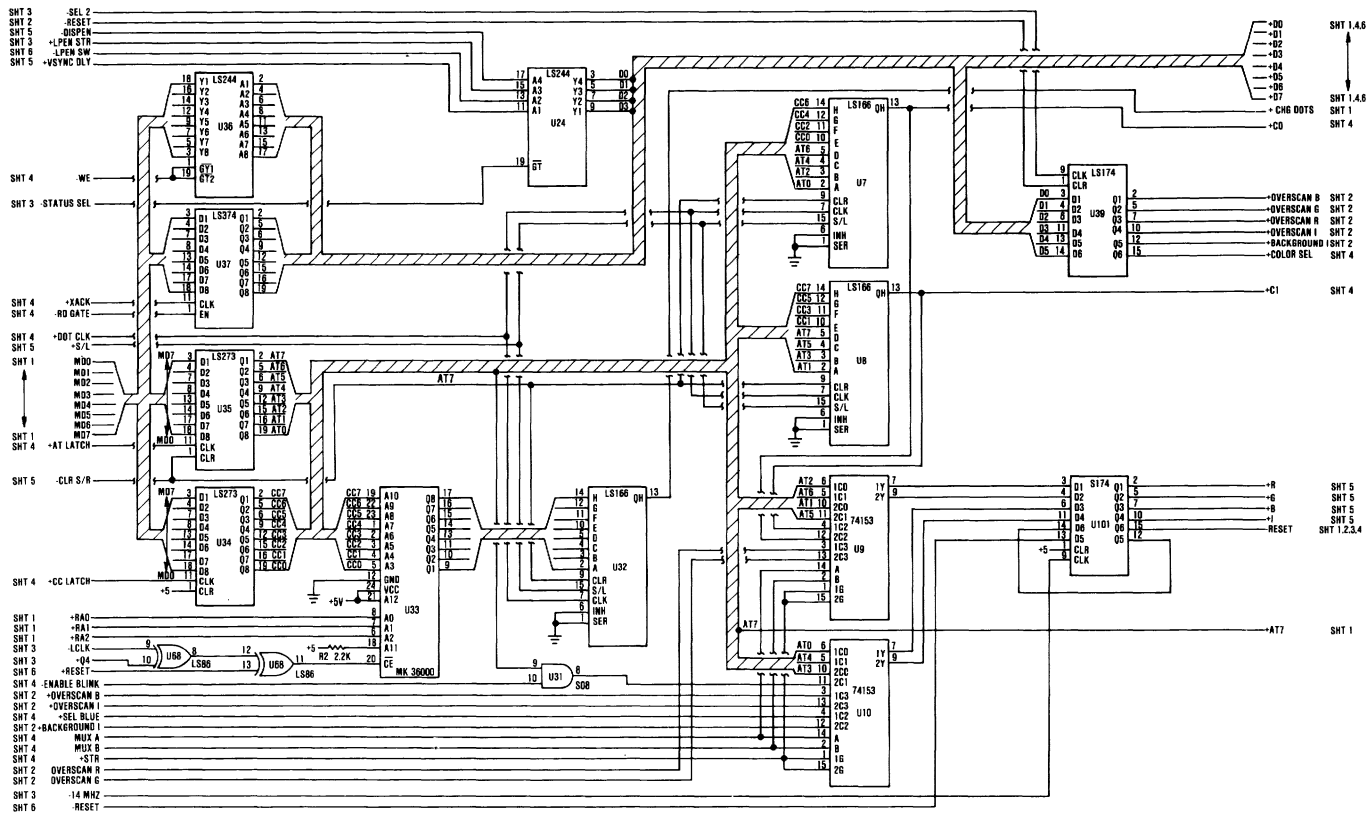
Monochrome Display Adapter (Sheet 9 of 10)



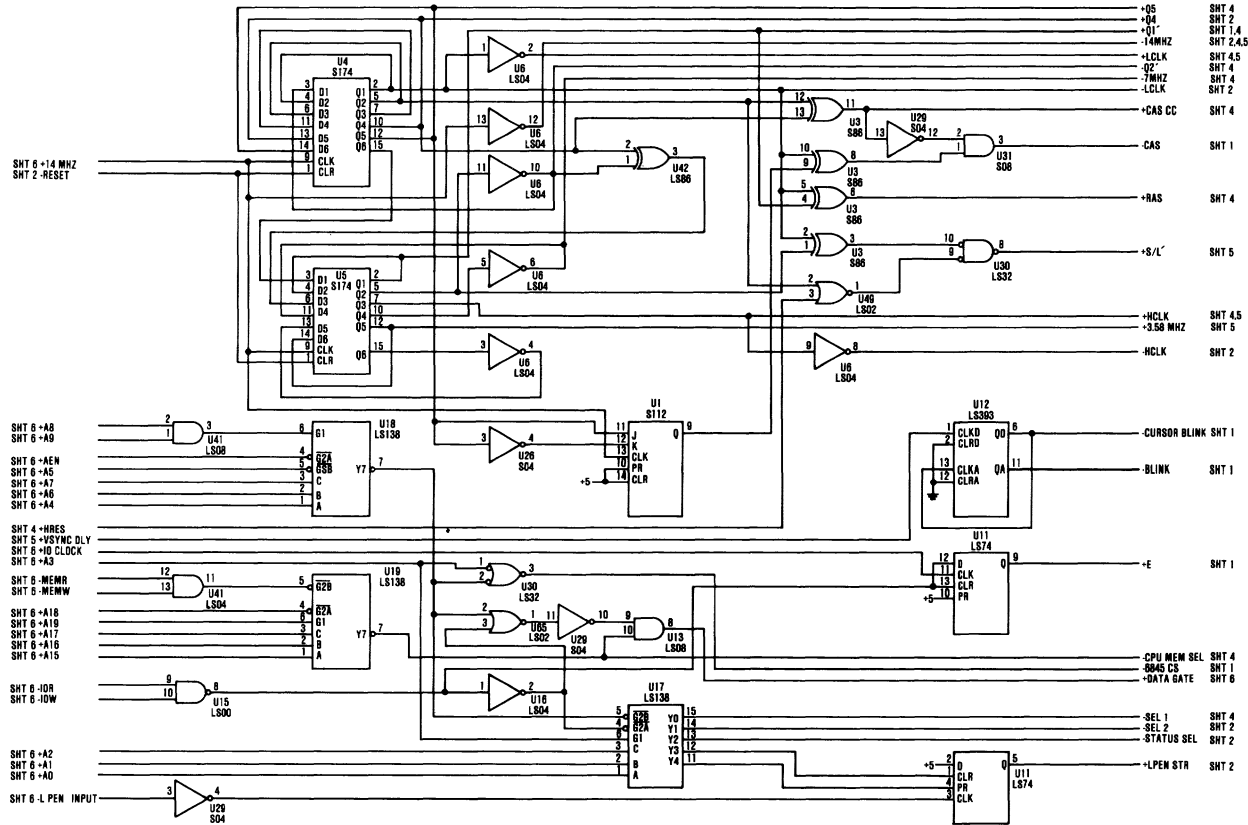
Monochrome Display Adapter (Sheet 10 of 10)



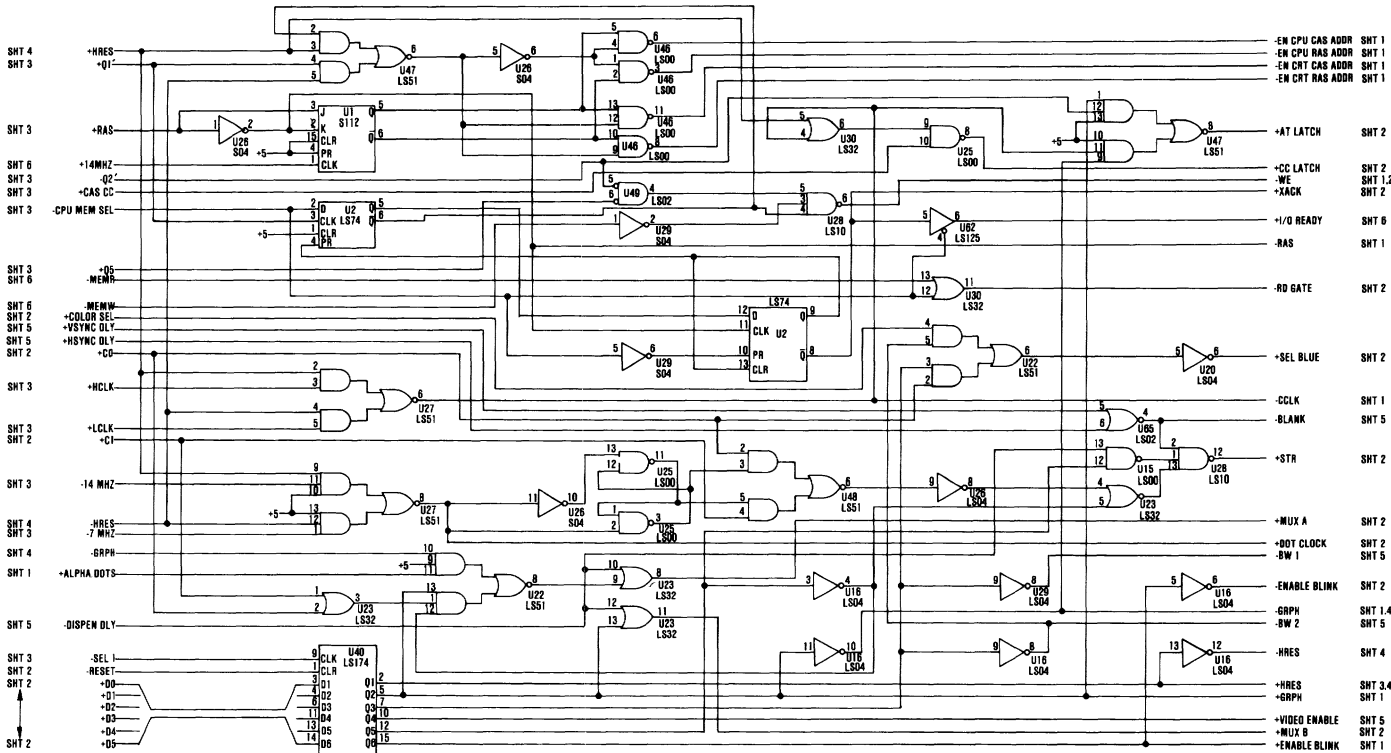
Color/Graphics Monitor Adapter (Sheet 1 of 6)



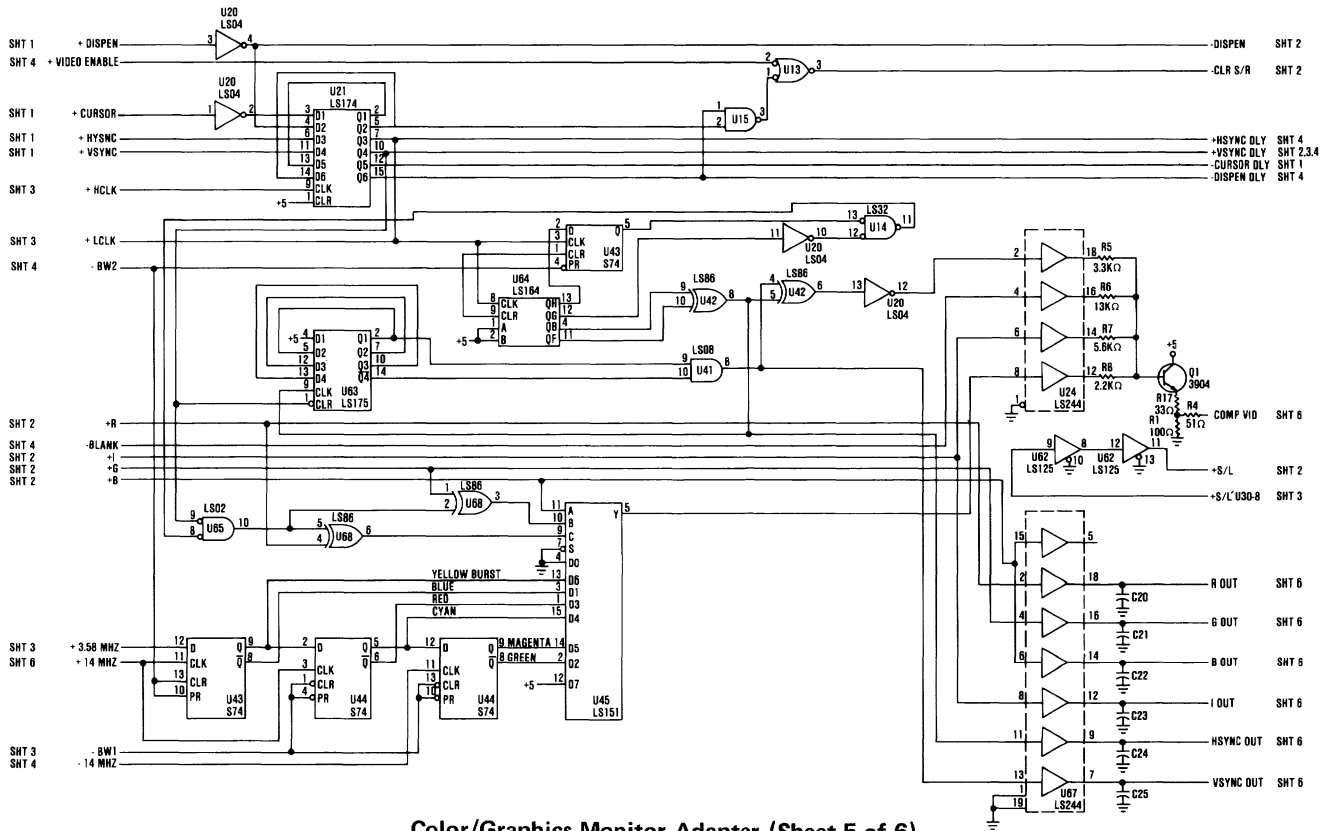
Color/Graphics Monitor Adapter (Sheet 2 of 6)



Color/Graphics Monitor Adapter (Sheet 3 of 6)

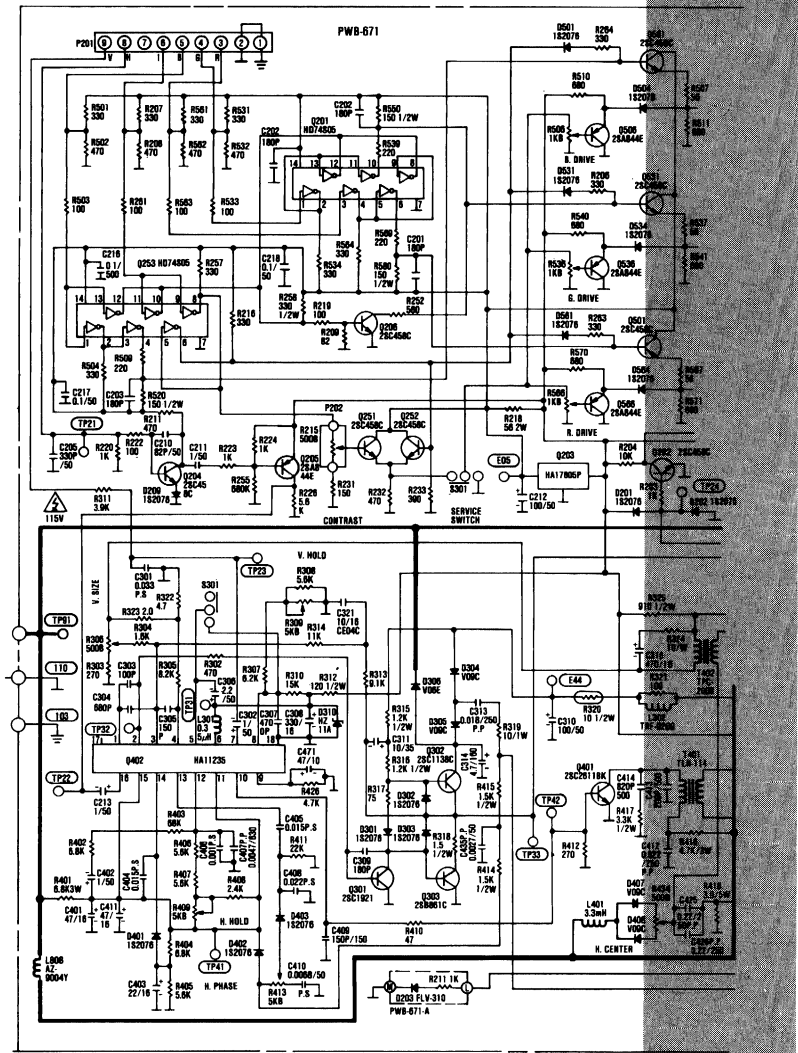


Color/Graphics Monitor Adapter (Sheet 4 of 6)



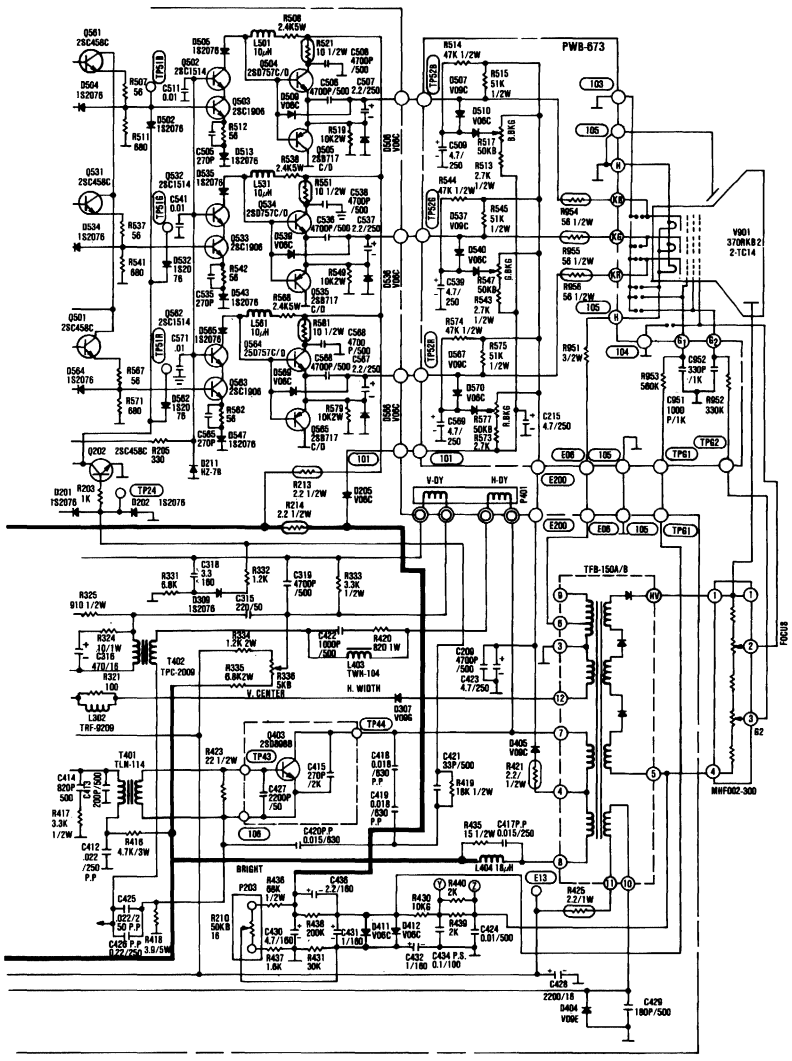
Color/Graphics Monitor Adapter (Sheet 5 of 6)

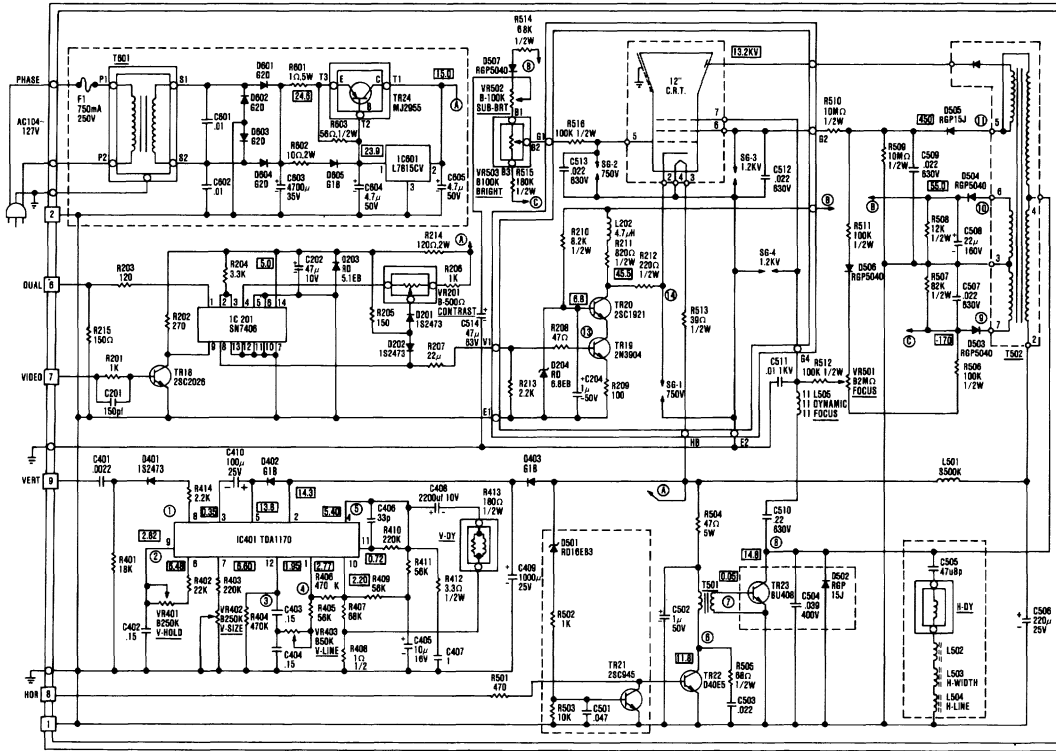
DANGER
HAZARDOUS VOLTAGES
UP TO 450 VOLTS EXIST
ON THE PRINTED
CIRCUIT BOARDS



Color Display (Sheet 1 of 1)

DANGER
HAZARDOUS VOLTAGES
UP TO 450 VOLTS EXIST
ON THE PRINTED
CIRCUIT BOARDS

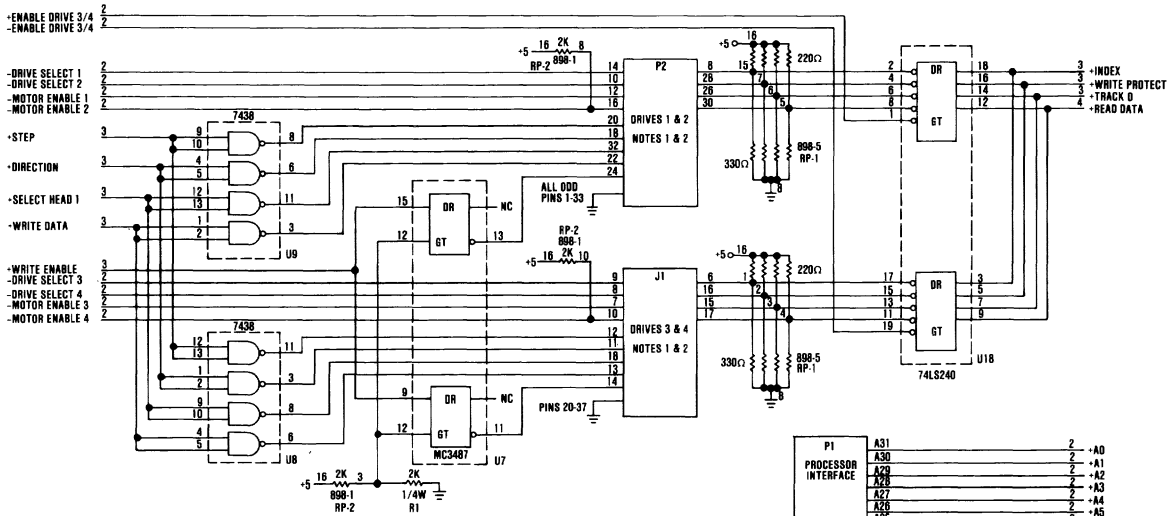




DANGER
HAZARDOUS VOLTAGES
UP TO 450 VOLTS EXIST
ON THE PRINTED
CIRCUIT BOARDS

- NOTES:
1. RESISTOR VALUES ARE IN (OHM) \square K = 1000; M = 1,000,000.
 2. ALL RESISTORS ARE 1/4W EXCEPT WHERE OTHERWISE INDICATED.
 3. ALL CAPACITORS ARE 50V EXCEPT WHERE OTHERWISE INDICATED.
 4. CAPACITORS VALUES ARE μ F UNLESS OTHERWISE INDICATED. μ = μ F = 10⁻⁶.
 5. AC WIRING INFORMATION
 PHASE = BLACK/BROWN WIRE
 NEUTRAL = WHITE/BLUE WIRE
 GROUND = GREEN AND YELLOW WIRE
 IMPORTANT: THE PHASE WIRE MUST GO TO THE FUSED SIDE OF TRANSFORMER.

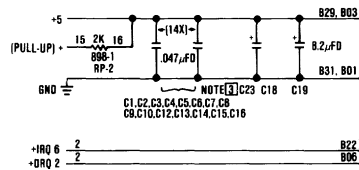
Monochrome Display (Sheet 1 of 1)



NOTES:

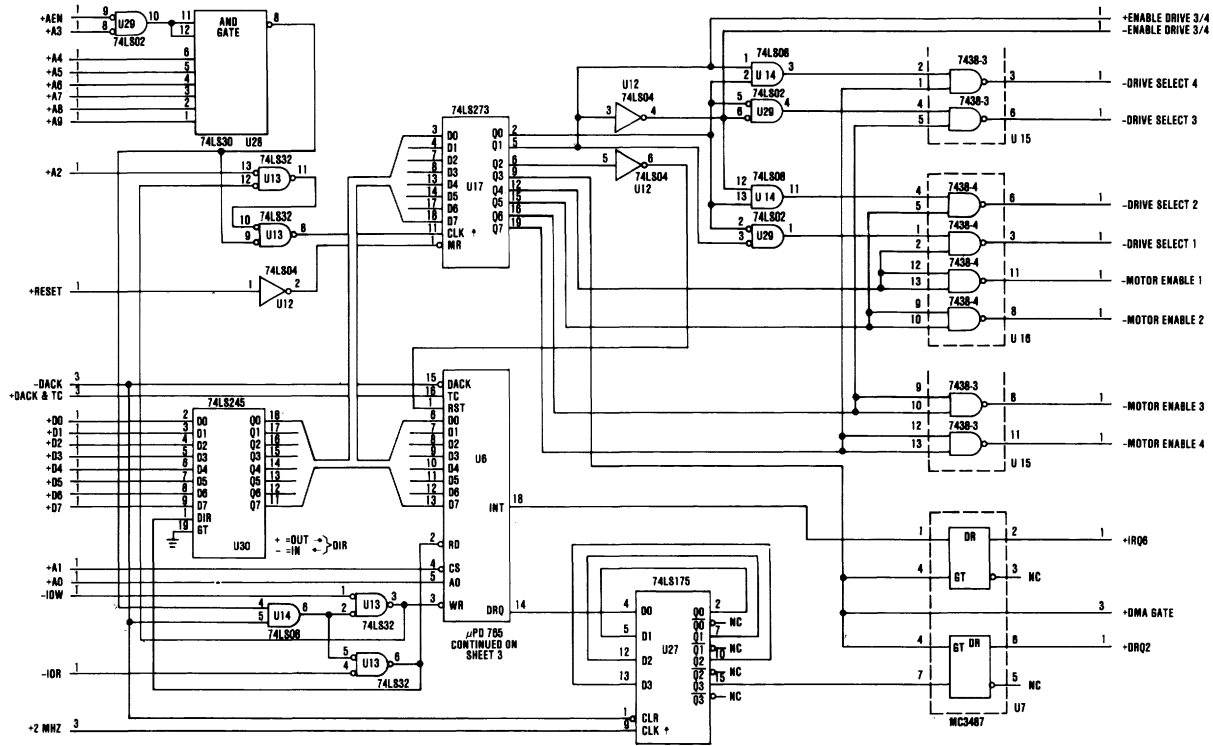
1. SIGNALS ON DRIVE PINS 10 THRU 16 ARE SWAPPED BY THE DRIVE CABLE BETWEEN DRIVES 1 & 2 (AND 3 & 4) AS FOLLOWS:

10	TO	18
11	TO	15
12	TO	14
13	TO	13
14	TO	12
15	TO	11
16	TO	10
2. ALL DRIVES ARE JUMPERED FOR MULTIPLEX OPERATION. HEAD LOAD WITH DRIVE SELECT AND DRIVE SELECT VIA INPUT PIN 12. TERMINATING R-PACS ARE LEFT IN DRIVES 1 & 3 ONLY
3. .047 μ F CAP SHOULD BE ADJACENT TO MODULES MC3487, 7438, 74S74, 16 MHz OSC, RP-1, MC4044, MC4024, 74LS161 & 74LS191. 8.2 μ F CAPS SHOULD BE NEAR ASSOCIATED P1 PINS.
4. ALL SIGNAL LINES HIGHER THAN OR EQUAL TO 1MHZ SHOULD BE KEPT TO THE SHORTEST POSSIBLE LENGTH. THIS IS A PRIMARY DESIGN GOAL.
5. MAKE NO CONNECTION TO UNUSED PINS ON THE VCD, CHARGE PUMP & DATA SEPARATOR MODULES.
6. ALL VOLTAGE AND GROUND CONNECTORS TO THE VCD, CHARGE PUMP AND ASSOCIATED DISCRETE COMPONENTS SHOULD BE SEPARATE FROM OTHER CIRCUITS AND THEN JOINED TO THE OTHER CIRCUITS AT ONE POINT.

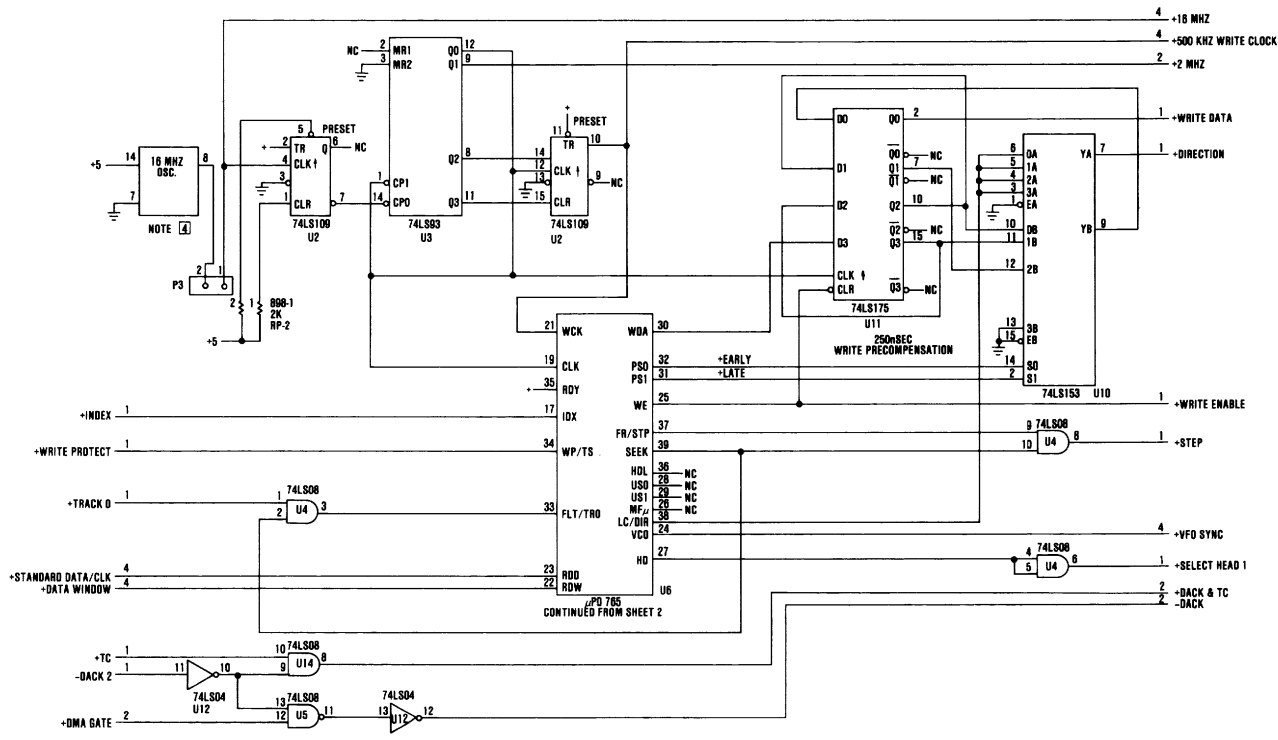


P1	A31	2	-A0
PROCESSOR	A30	2	-A1
INTERFACE	A29	2	-A2
	A28	2	-A3
	A27	2	-A4
	A26	2	-A5
	A25	2	-A6
	A24	2	-A7
	A23	2	-A8
	A22	2	-A9
	A11	2	-AEN
	B14	2	-IOR
	B13	2	-IOW
	B02	2	-RESCT
	B27	2	-DACK
	A09	2	-D0
	A08	2	-D1
	A07	2	-D2
	A06	2	-D3
	A05	2	-D4
	A04	2	-D5
	A03	2	-D6
	A02	2	-D7
	B22	3	-DACK 2
	B06		

5-1/4 Inch Diskette Drive Adapter (Sheet 1 of 4)

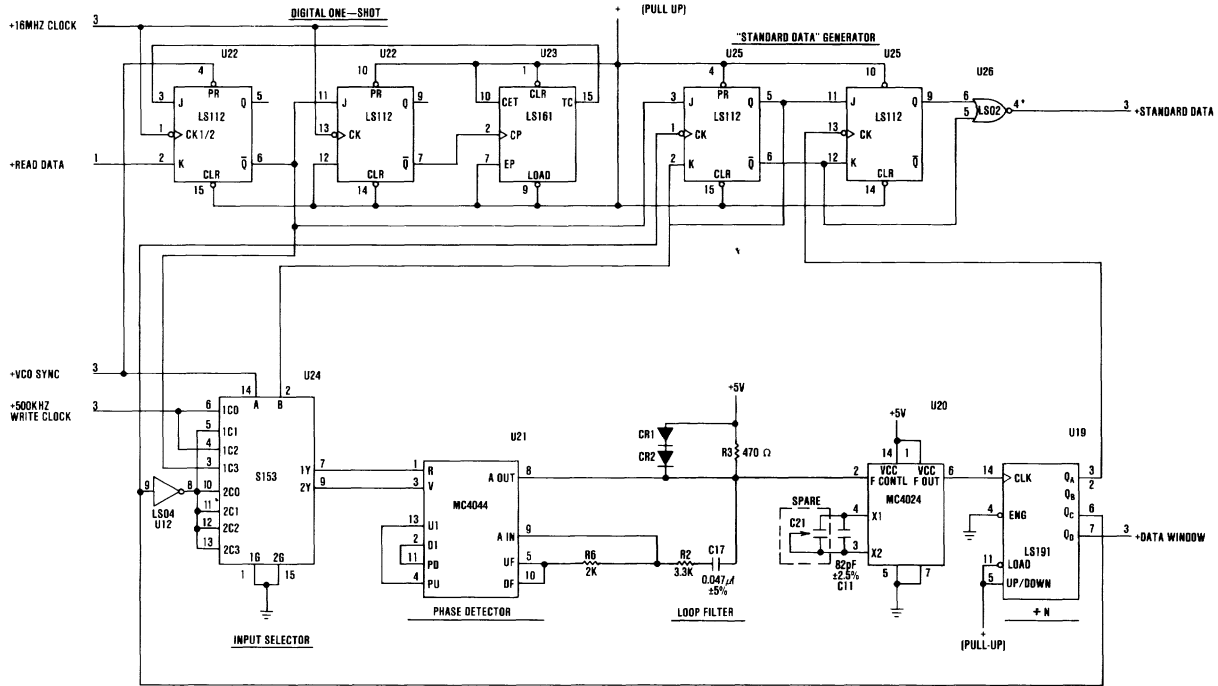


5-1/4 Inch Diskette Drive Adapter (Sheet 2 of 4)

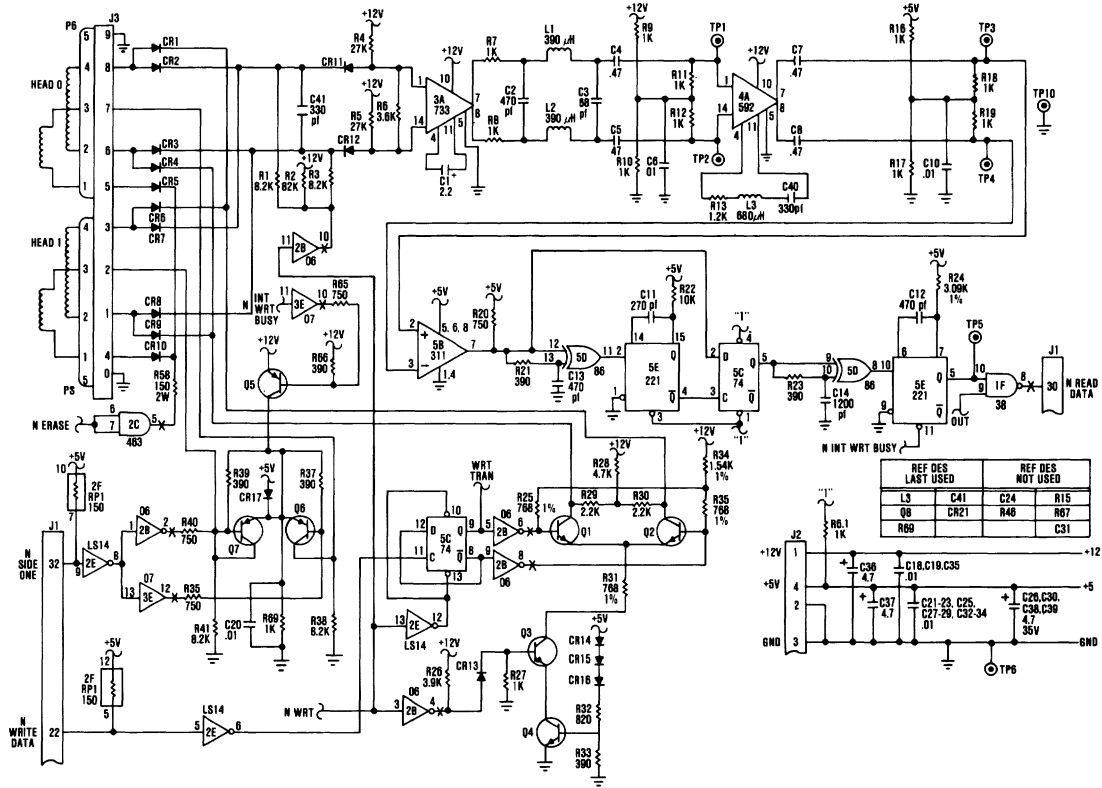


NOTE:
U4 [74LS08] PINS 12 AND 13 ARE CONNECTED ONLY ON CARDS BUILT
USING RAW CARD P/N 5001293

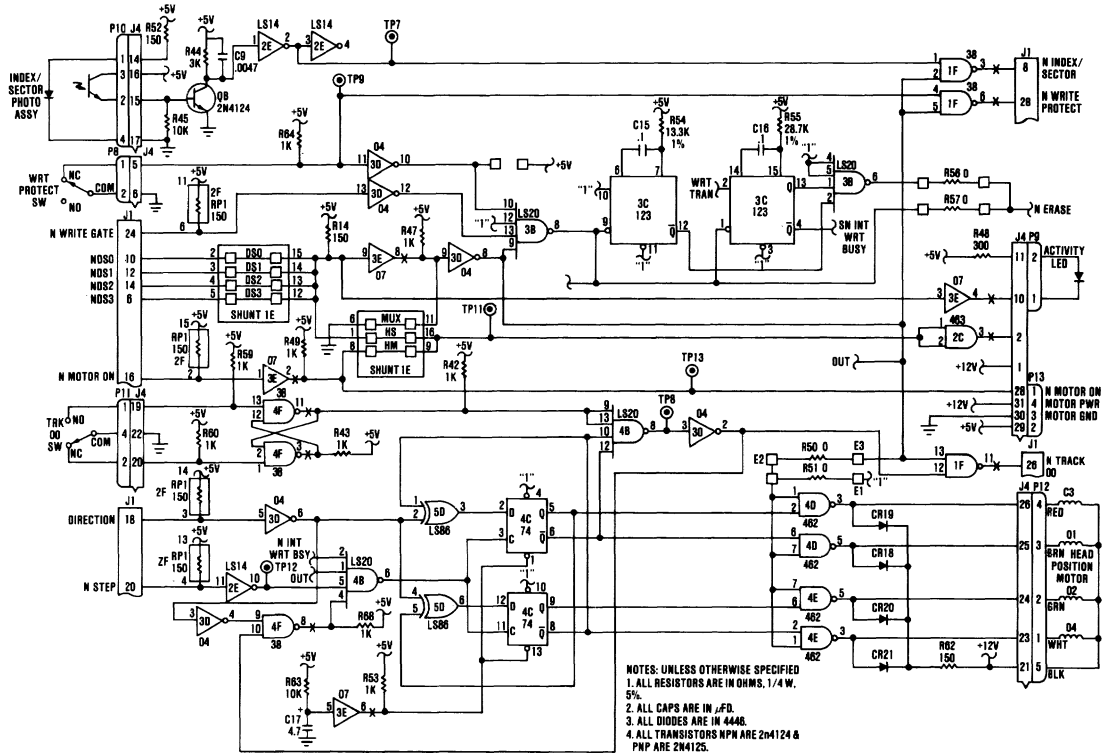
5-1/4 Inch Diskette Drive Adapter (Sheet 3 of 4)



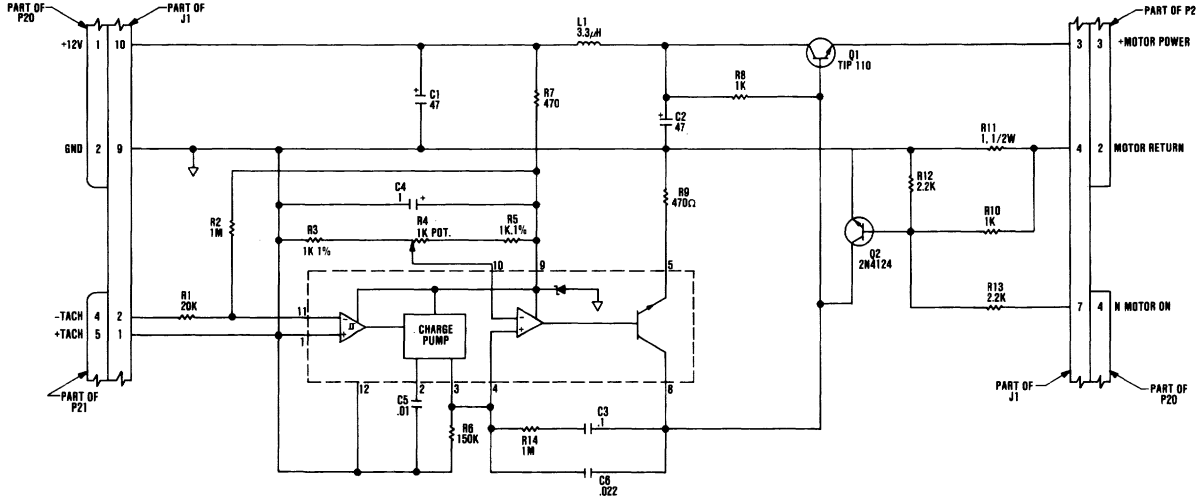
5-1/4 Inch Diskette Drive Adapter (Sheet 4 of 4)



5-1/4 Inch Diskette Drive - Type 1 (Sheet 1 of 3)

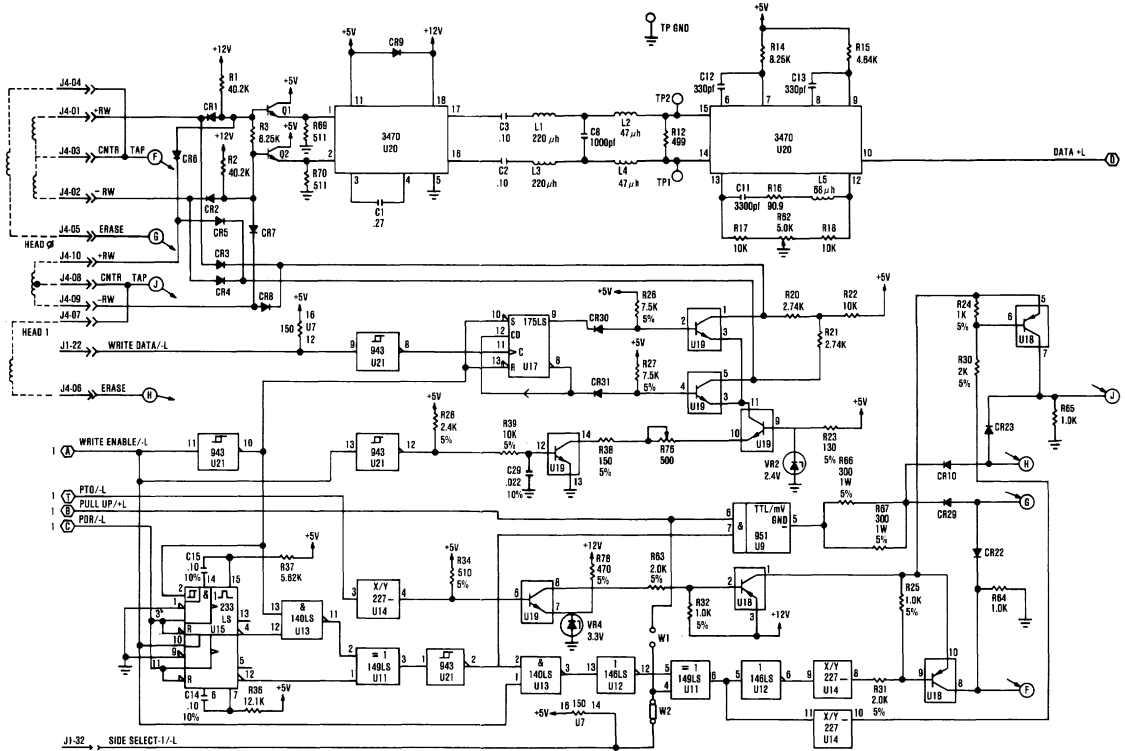


5-1/4 Inch Diskette Drive - Type 1 (Sheet 2 of 3)

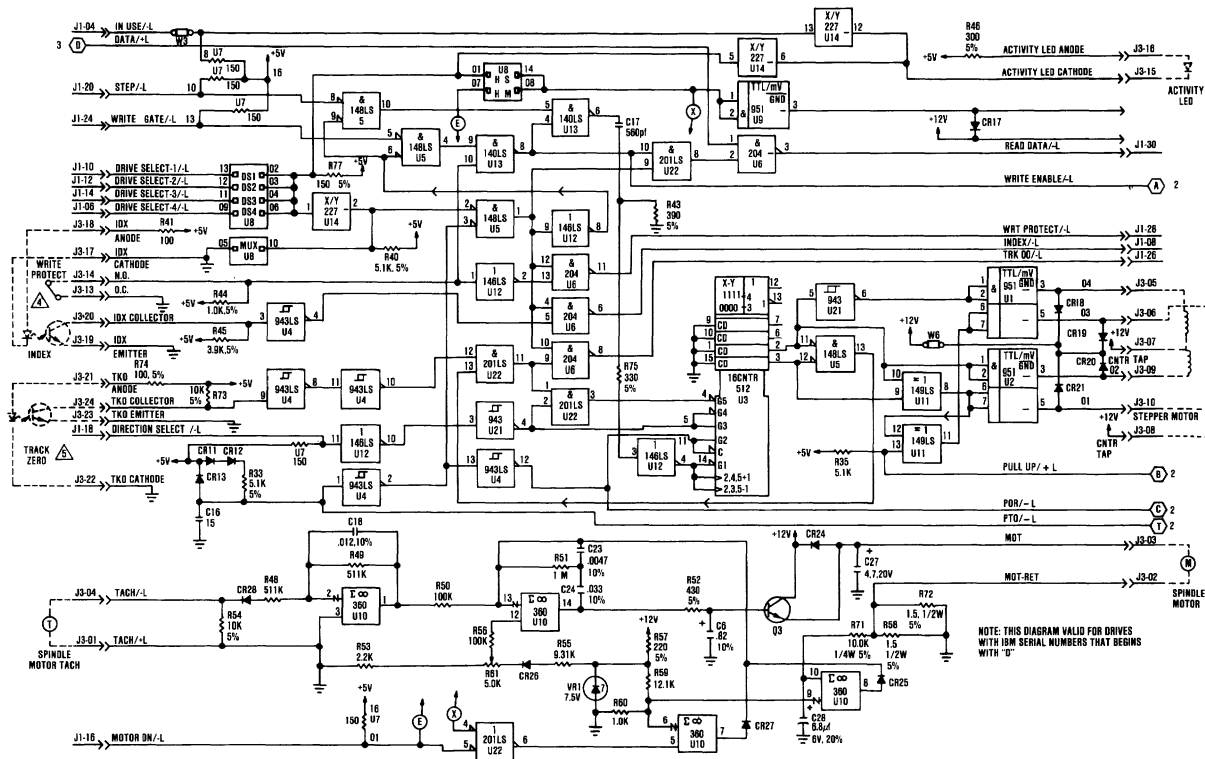


NOTES: UNLESS OTHERWISE SPECIFIED
 1. RESISTORS ARE IN OHMS, ±5%, 1/4W.
 2. 1% RESISTORS ARE 1/8W.
 3. CAPACITORS ARE IN μF, ±20%, 35V.

5-1/4 Inch Diskette Drive - Type 1 (Sheet 3 of 3)

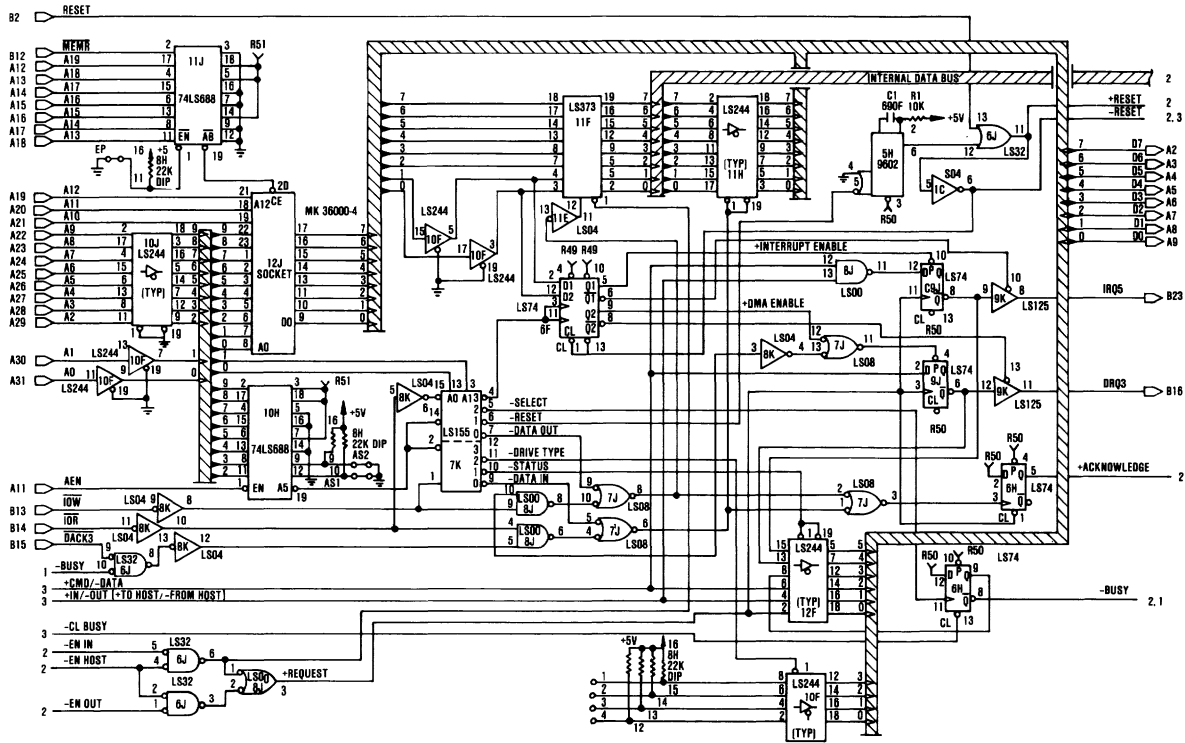


5-1/4 Inch Diskette Drive - Type 2 (Sheet 1 of 2)

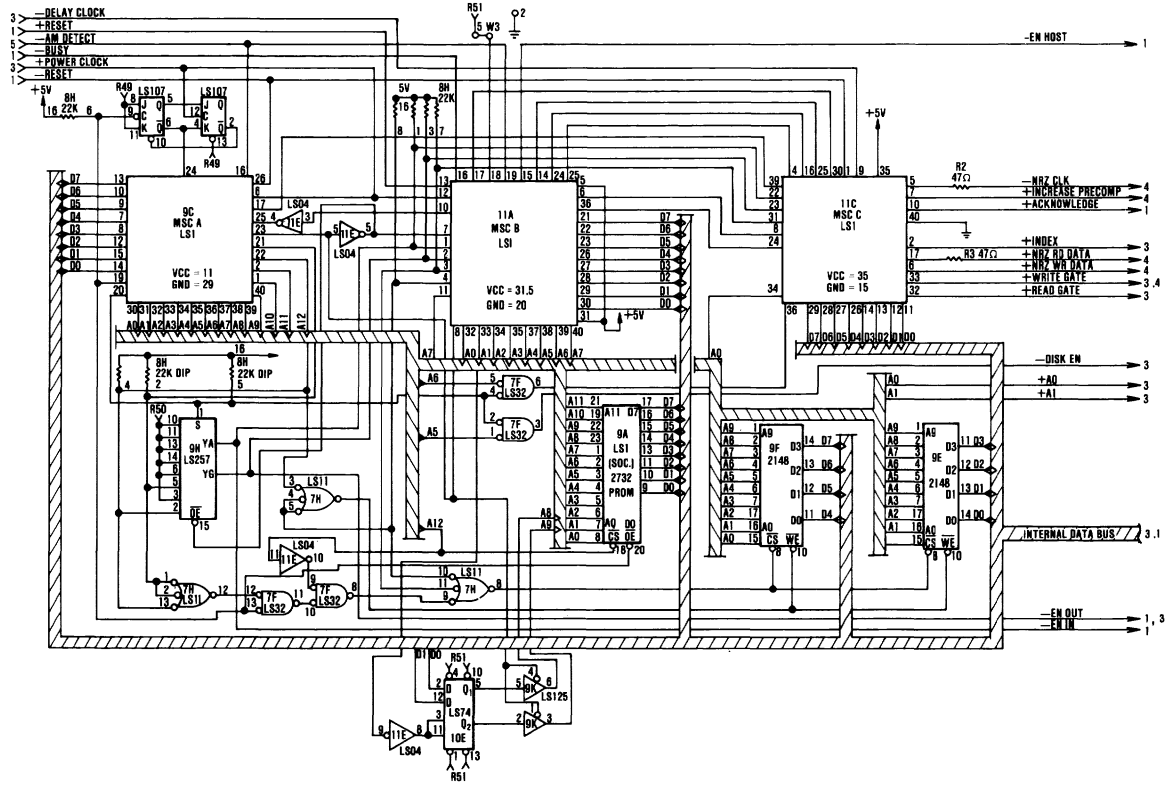


5-1/4 Inch Diskette Drive - Type 2 (Sheet 2 of 2)

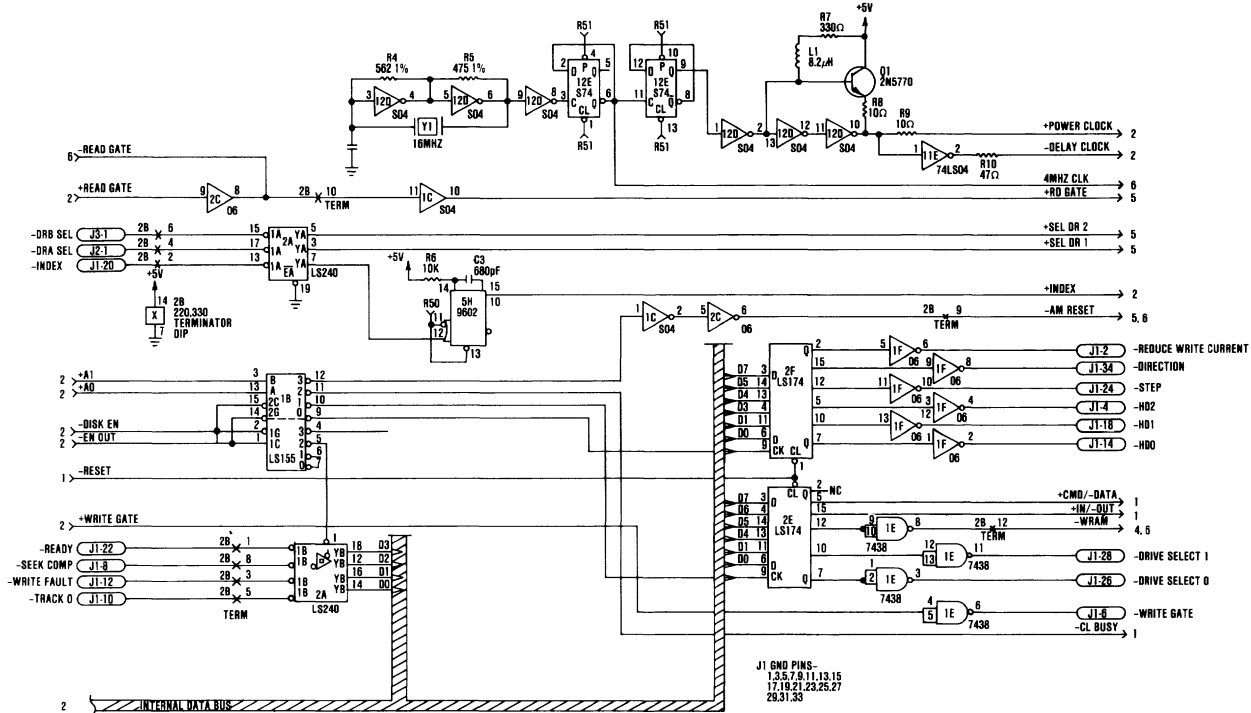
D-54 Logic Diagrams



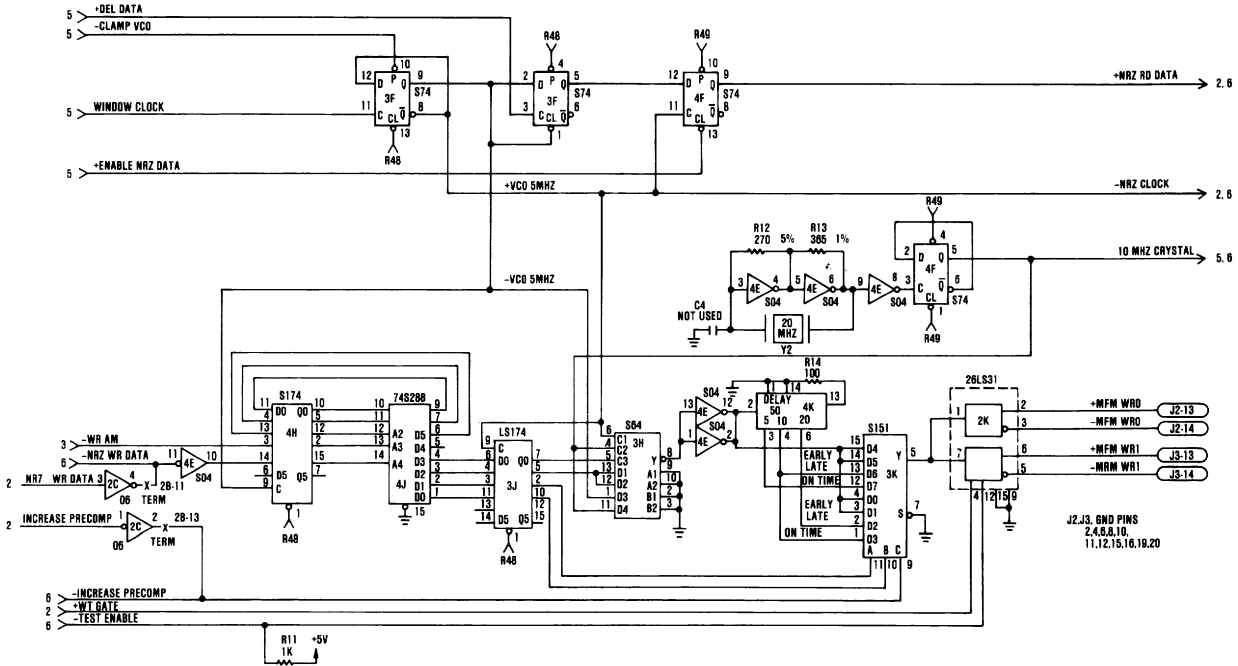
Fixed Disk Drive Adapter (Sheet 1 of 6)



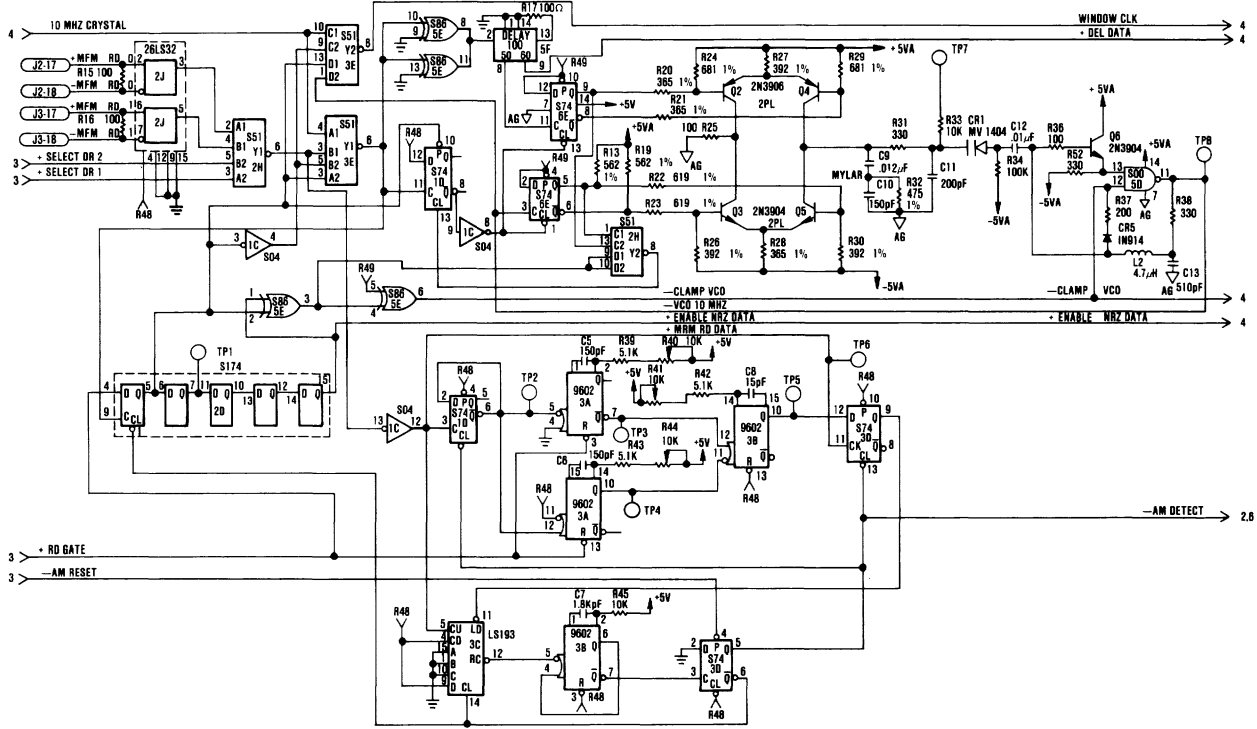
Fixed Disk Drive Adapter (Sheet 2 of 6)



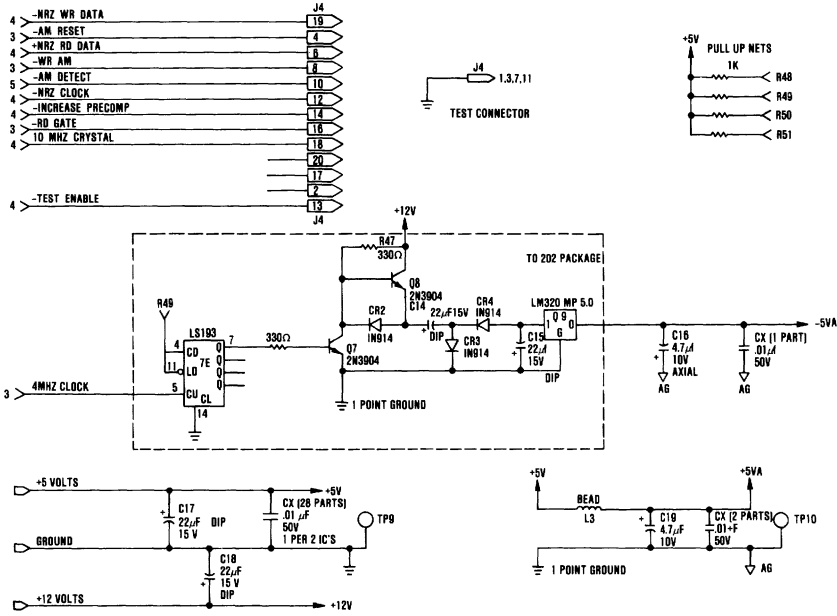
Fixed Disk Drive Adapter (Sheet 3 of 6)



Fixed Disk Drive Adapter (Sheet 4 of 6)

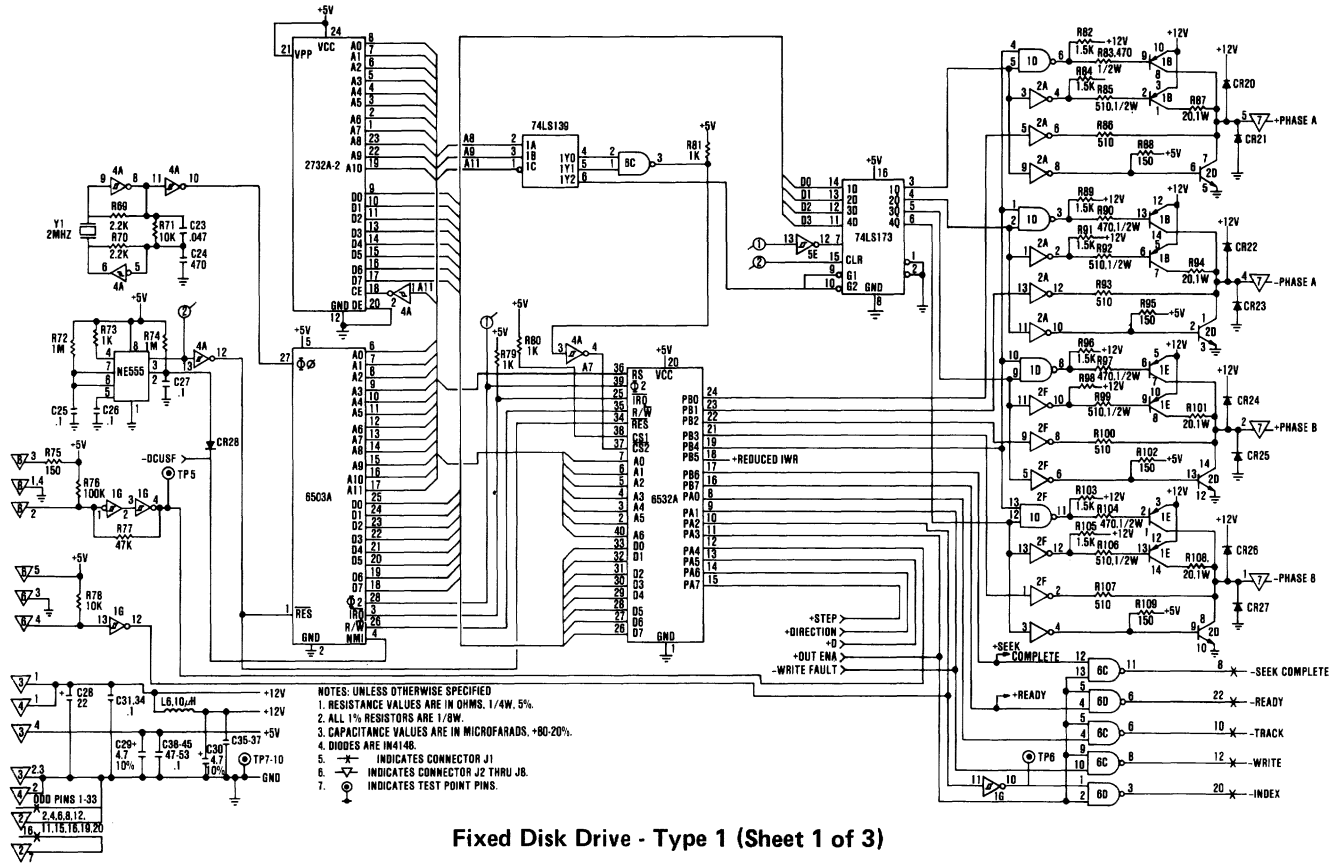


Fixed Disk Drive Adapter (Sheet 5 of 6)

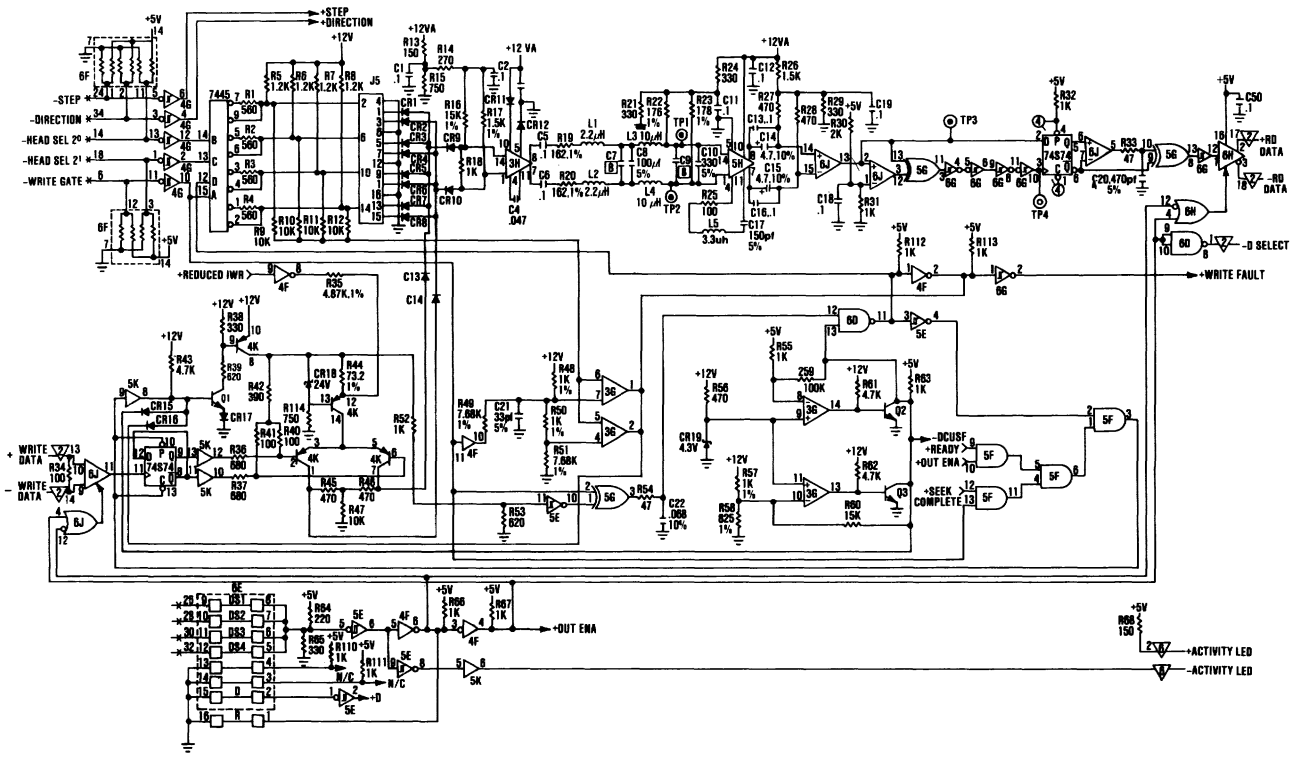


- NOTES:
 UNLESS OTHERWISE SPECIFIED:
 1. ALL RESISTORS 1/4 W. 5% CARBON FILTER.
 2. ALL CAPS +10V OR GREATER +15%.
 3. NO MORE THAN 15 LOADS PER PULLUP NET.

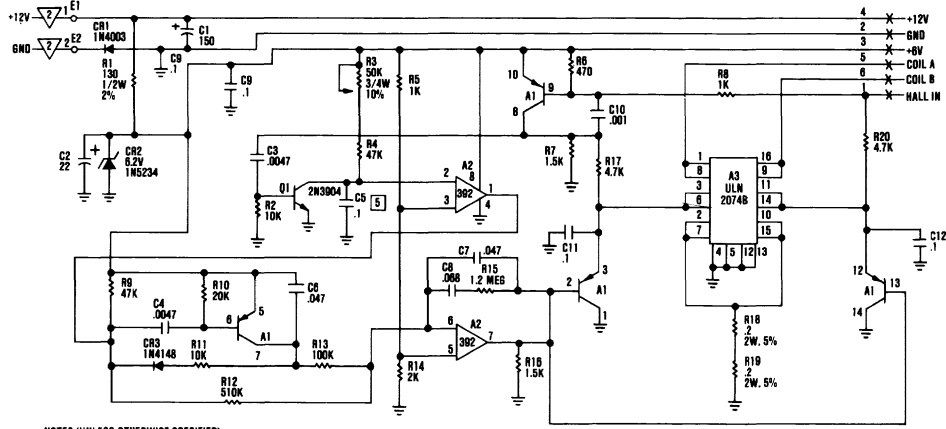
Fixed Disk Drive Adapter (Sheet 6 of 6)



Fixed Disk Drive - Type 1 (Sheet 1 of 3)

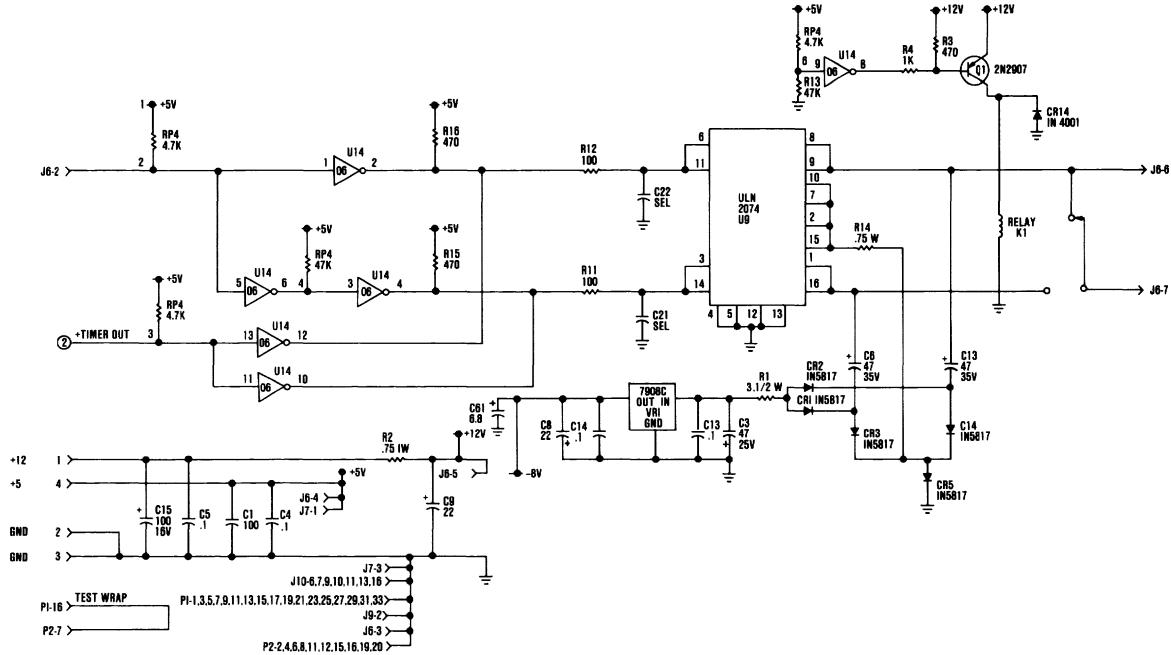


Fixed Disk Drive - Type 1 (Sheet 2 of 3)



NOTES (UNLESS OTHERWISE SPECIFIED):
 1. ALL RESISTORS ARE IN OHMS 1/4W, 5%.
 2. ALL CAPACITORS ARE IN MICROFARADS, 10%.
 3. —|— INDICATES J1.
 4. □ INDICATES J2.
 5. □ POLYCARBONATE 50V, 10%.

Fixed Disk Drive - Type 1 (Sheet 3 of 3)



NOTES:

1. SHEET TO SHEET CONNECTION IS AS FOLLOWS:

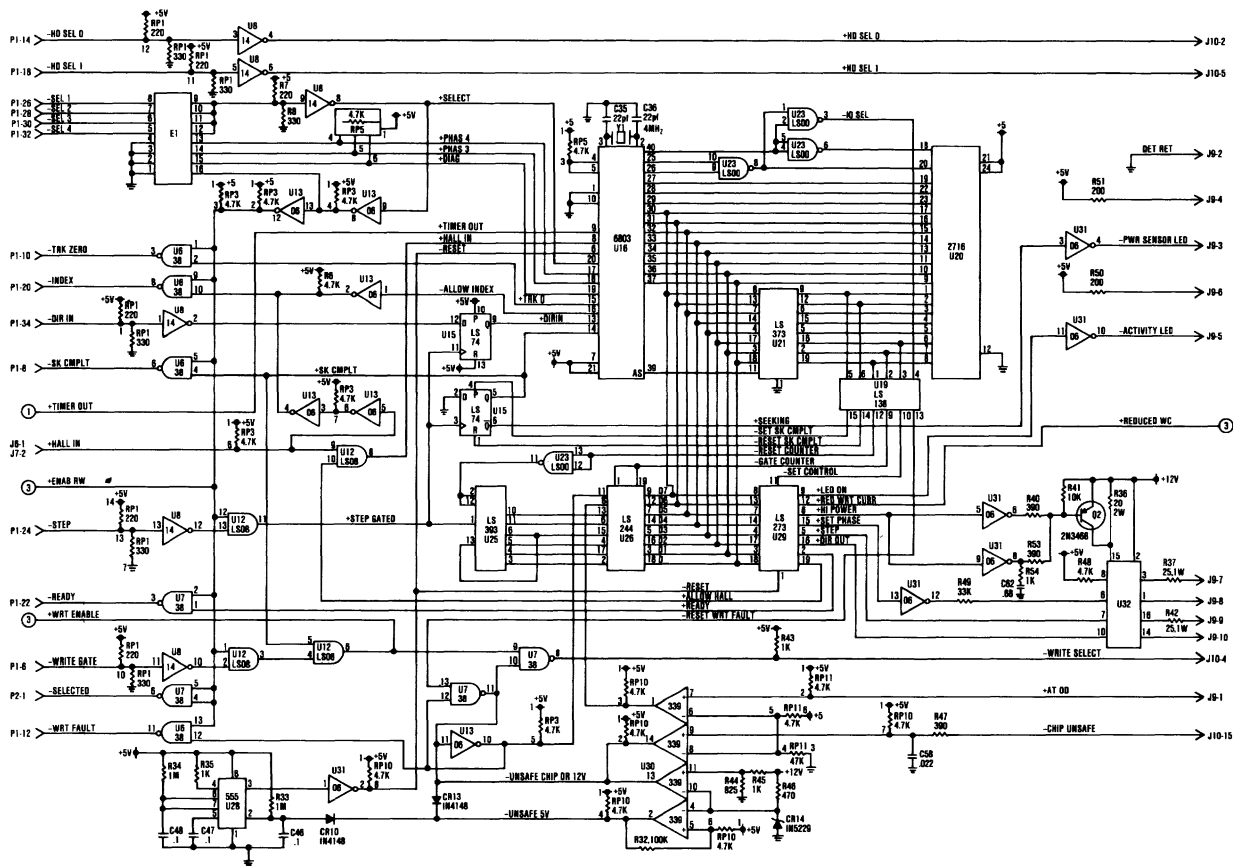


2. UNLESS OTHERWISE SPECIFIED ALL RESISTORS ARE 1/4W 5% VALUE IN OHMS

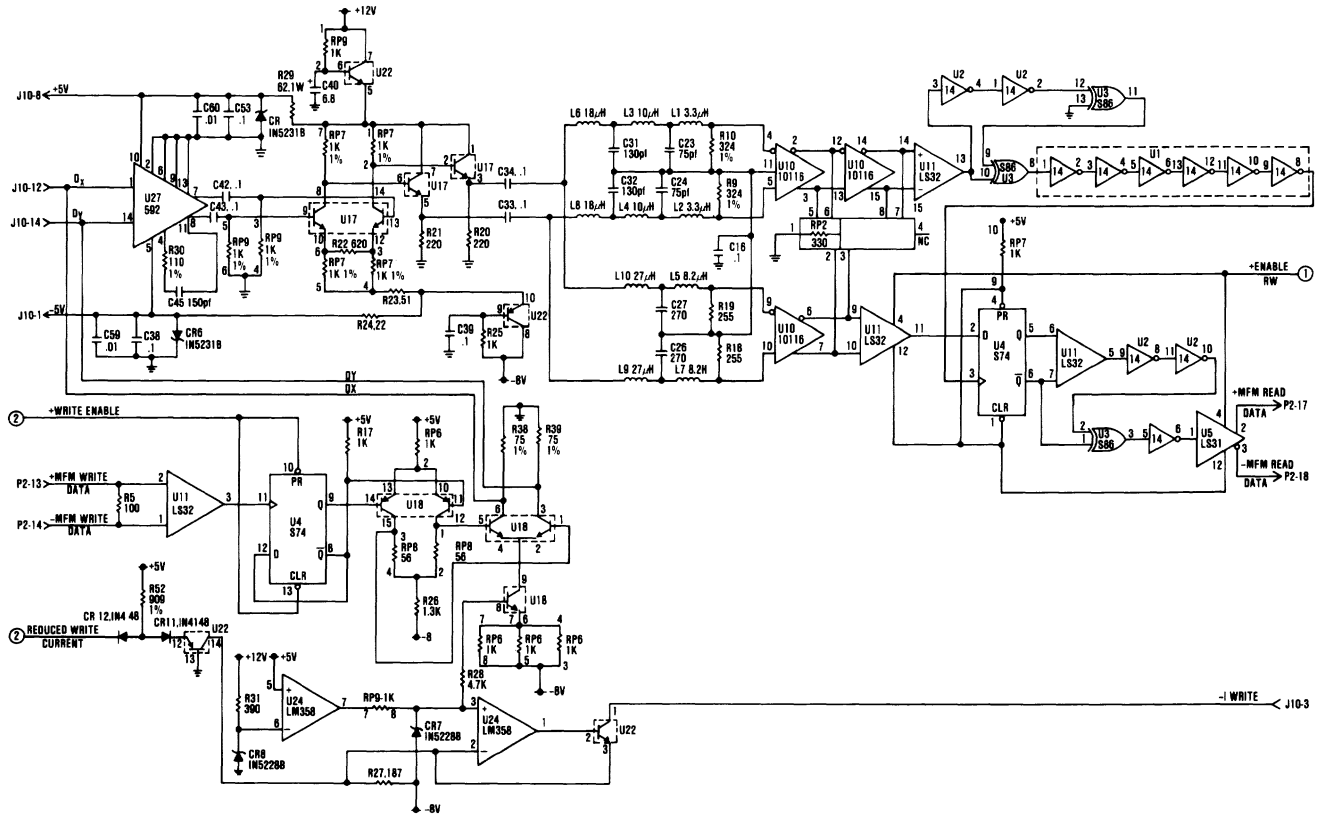
CAPACITORS +80-20% VALUE IN μF

3. E1 IS A PROGRAMMABLE JUMPER SOCKET

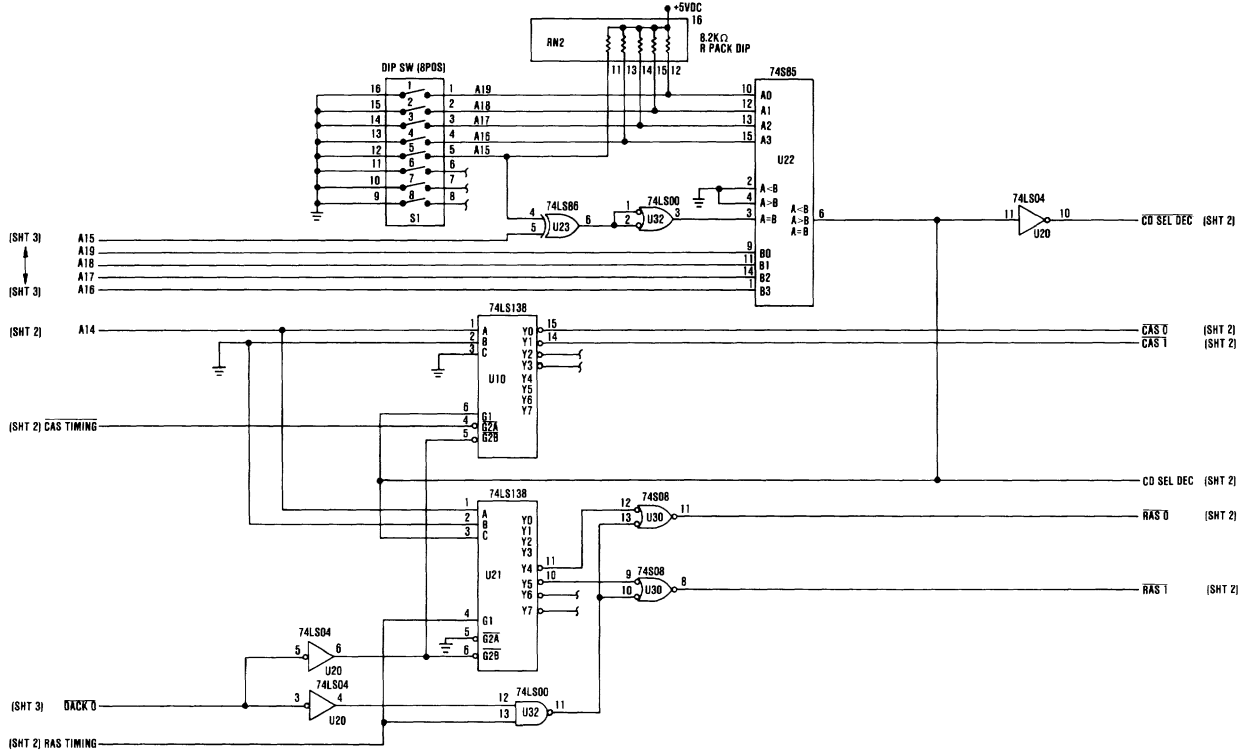
Fixed Disk Drive - Type 2 (Sheet 1 of 3)



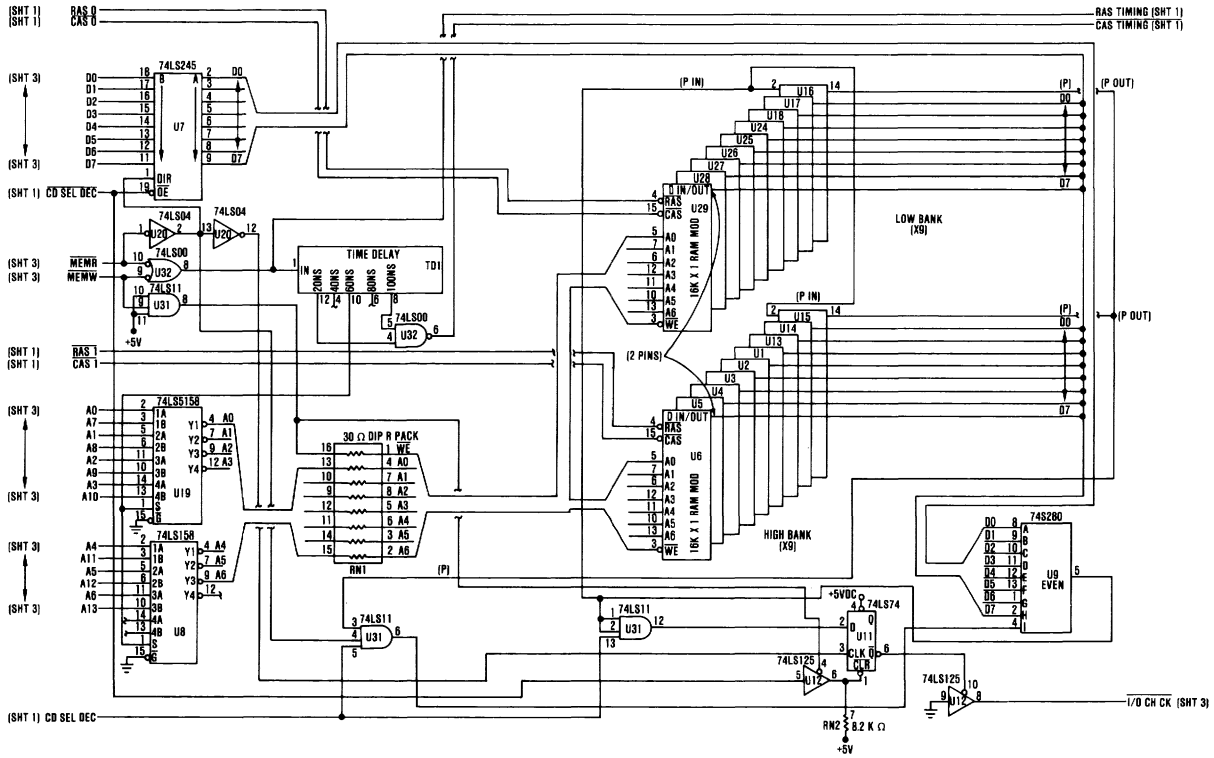
Fixed Disk Drive - Type 2 (Sheet 2 of 3)



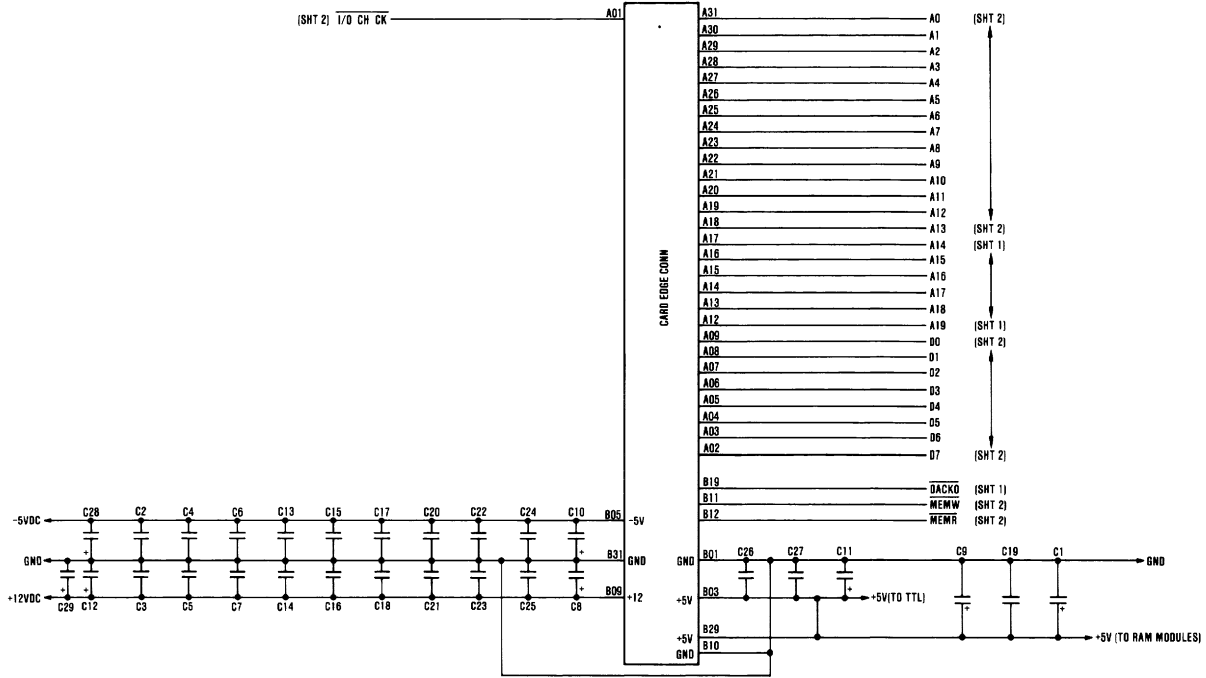
Fixed Disk Drive - Type 2 (Sheet 3 of 3)



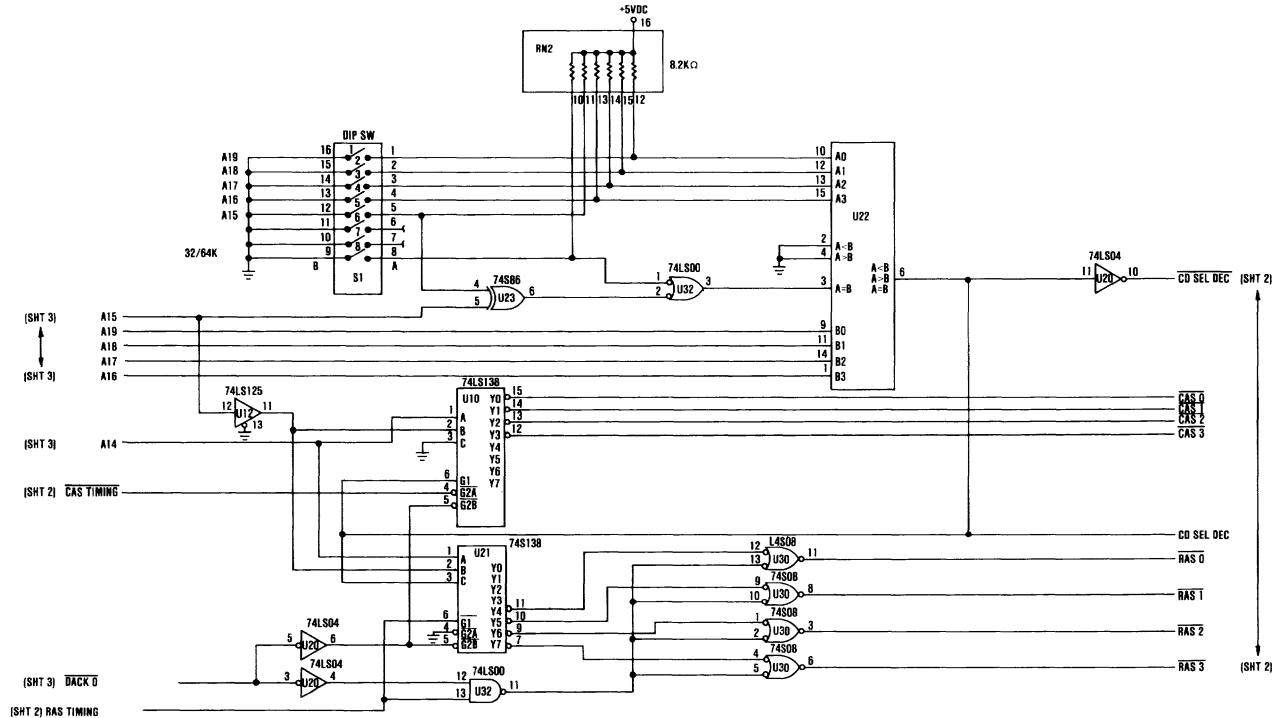
32K Memory Expansion Option (Sheet 1 of 3)



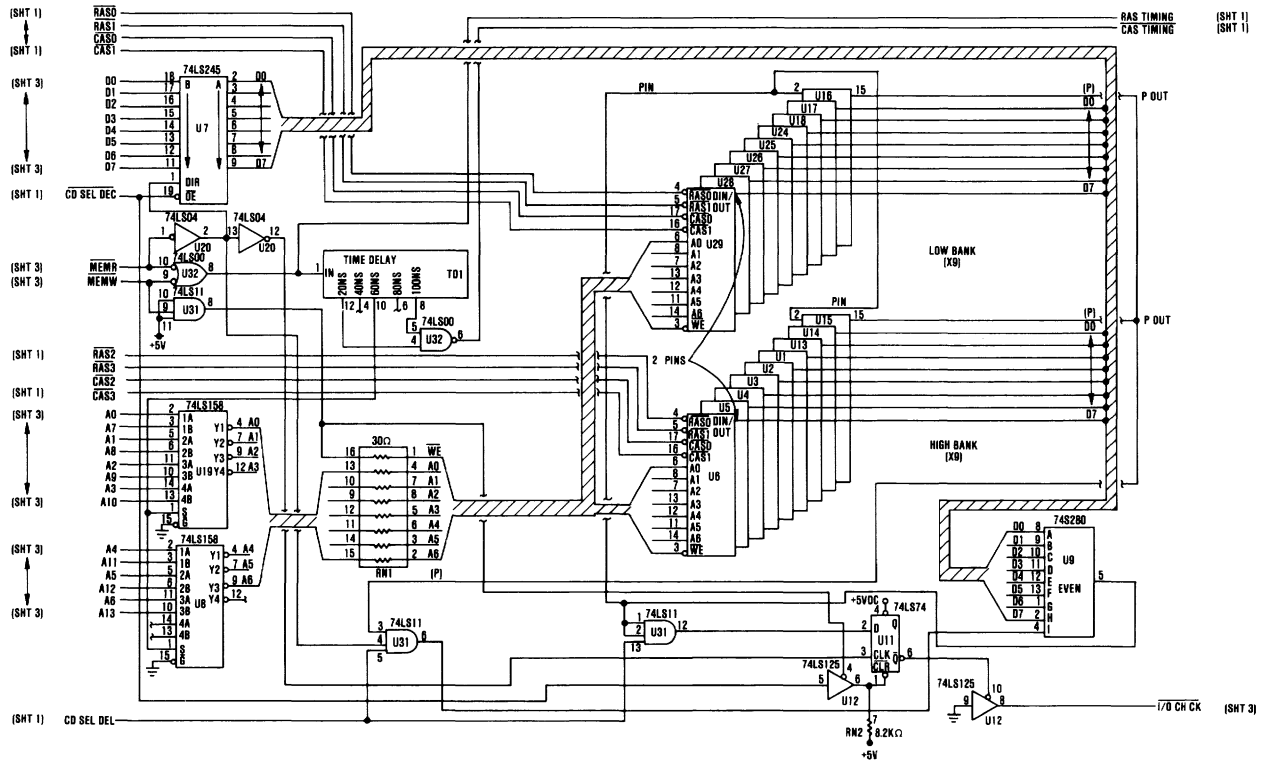
32K Memory Expansion Option (Sheet 2 of 3)



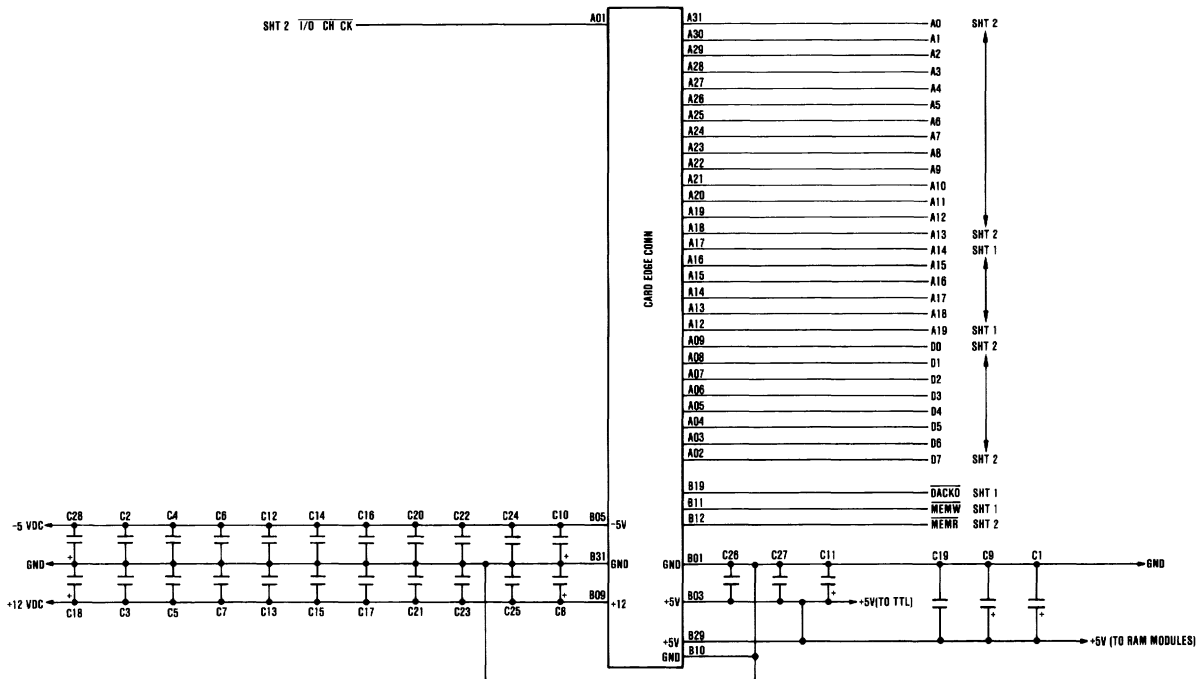
32K Memory Expansion Option (Sheet 3 of 3)



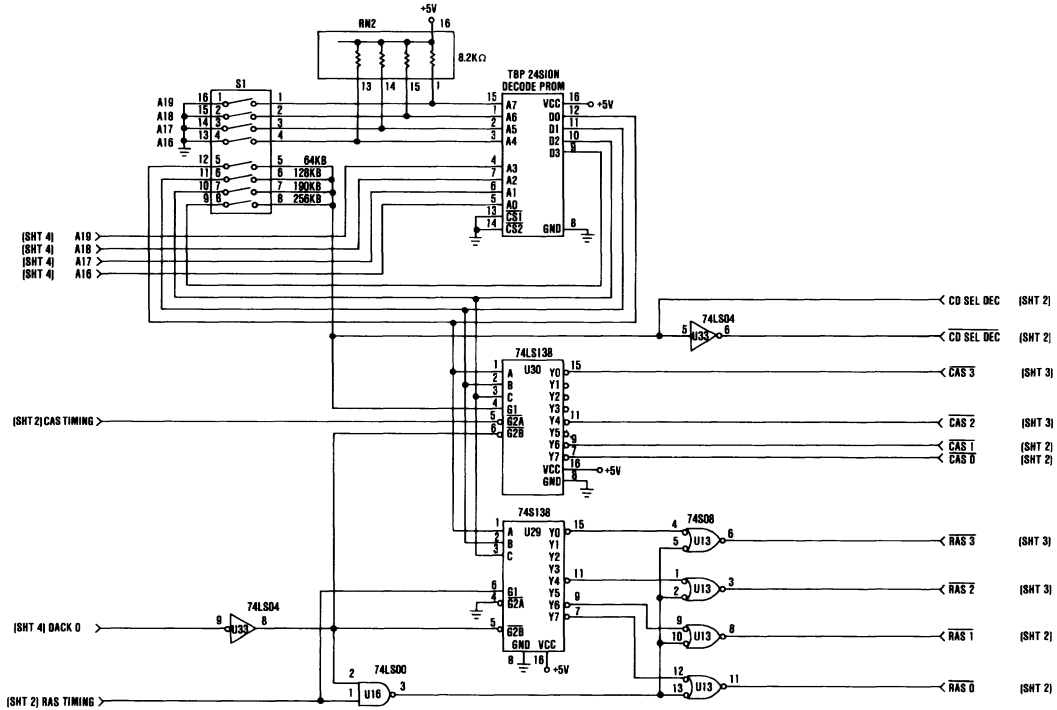
64K Memory Expansion Option (Sheet 1 of 3)



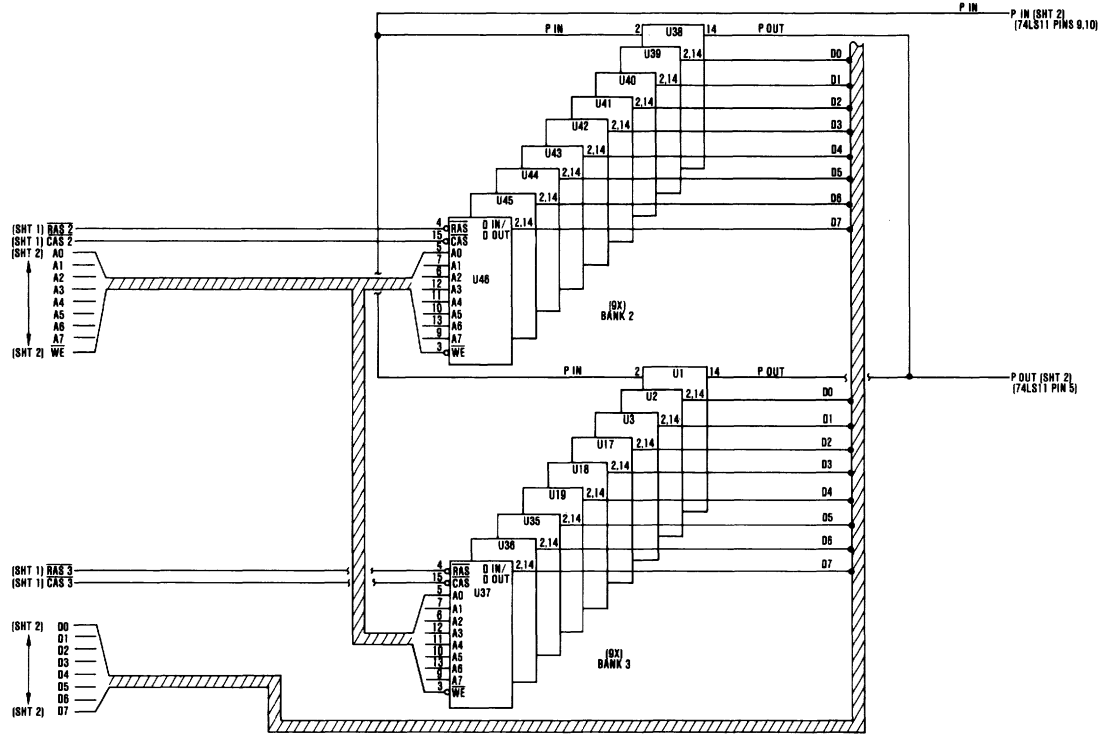
64K Memory Expansion Option (Sheet 2 of 3)



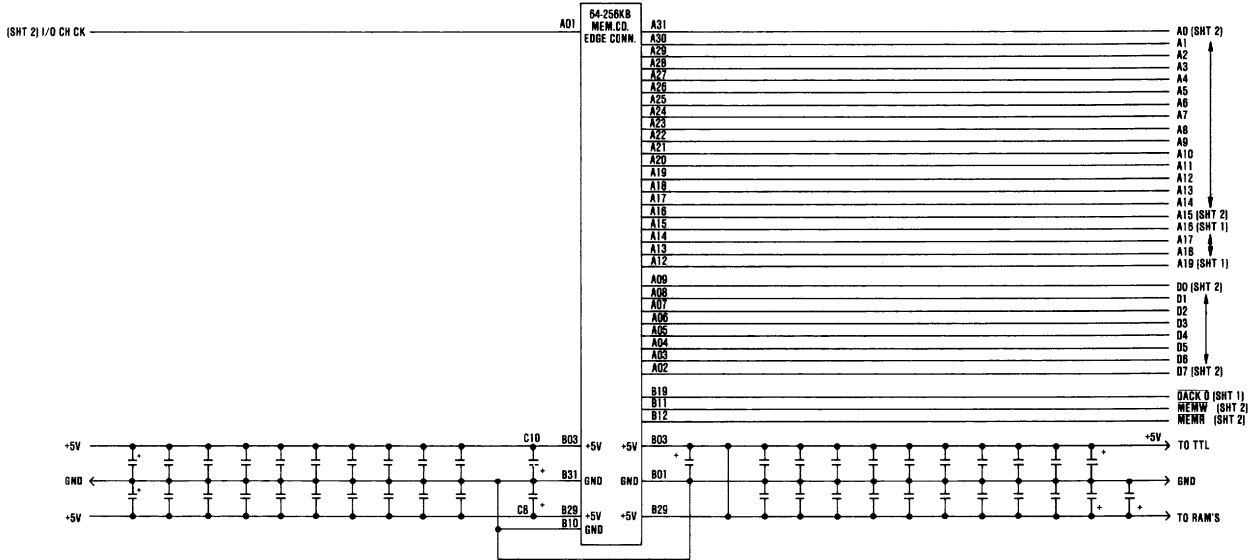
64K Memory Expansion Option (Sheet 3 of 3)



64/256K Memory Expansion Option (Sheet 1 of 4)

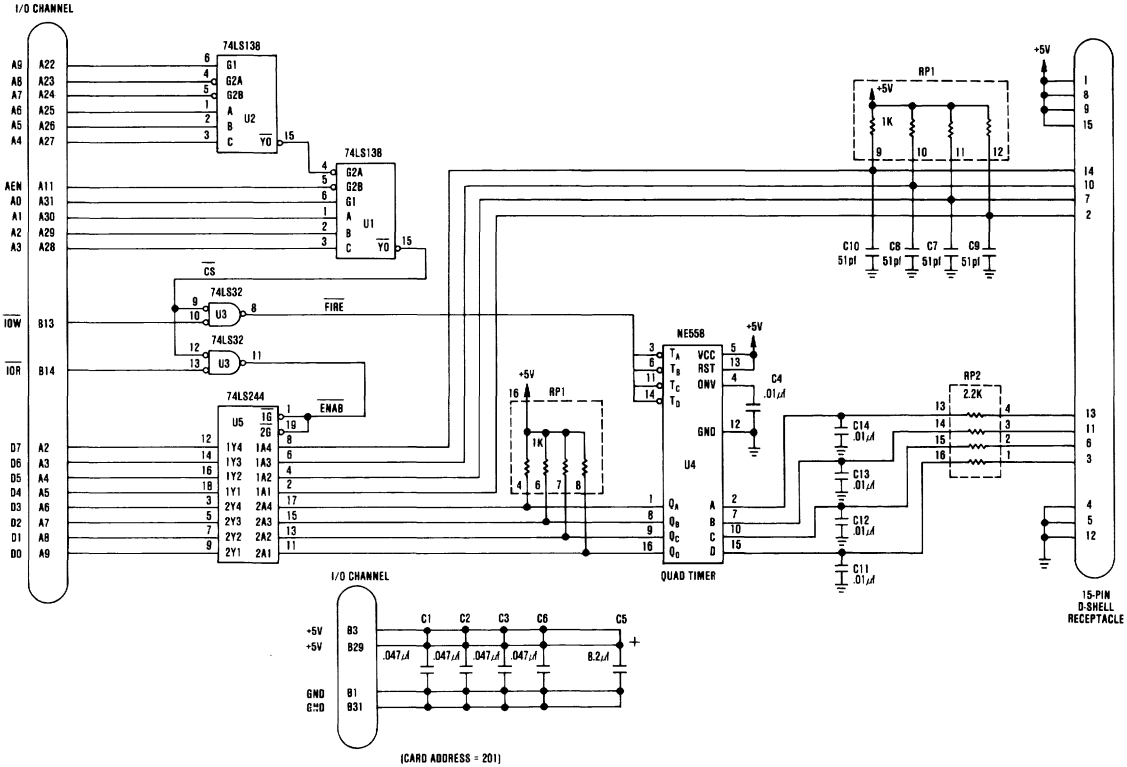


64/256K Memory Expansion Option (Sheet 3 of 4)

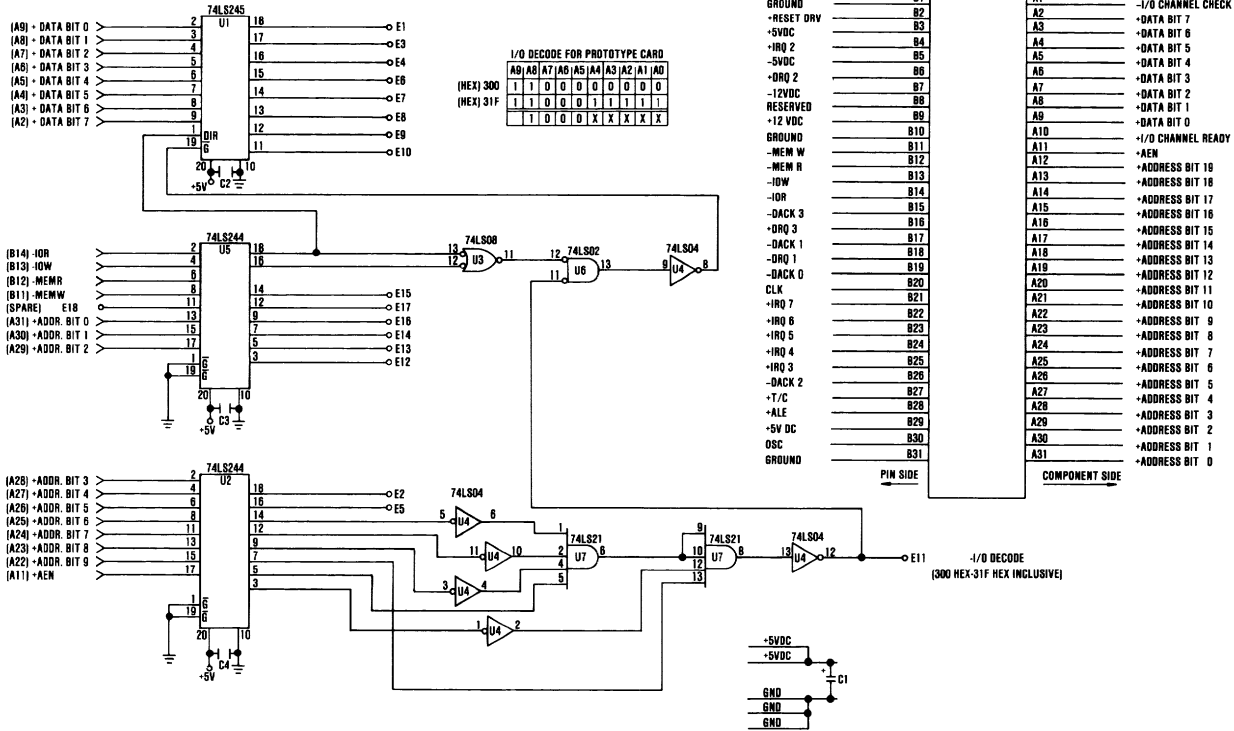


64/256K Memory Expansion Option (Sheet 4 of 4)

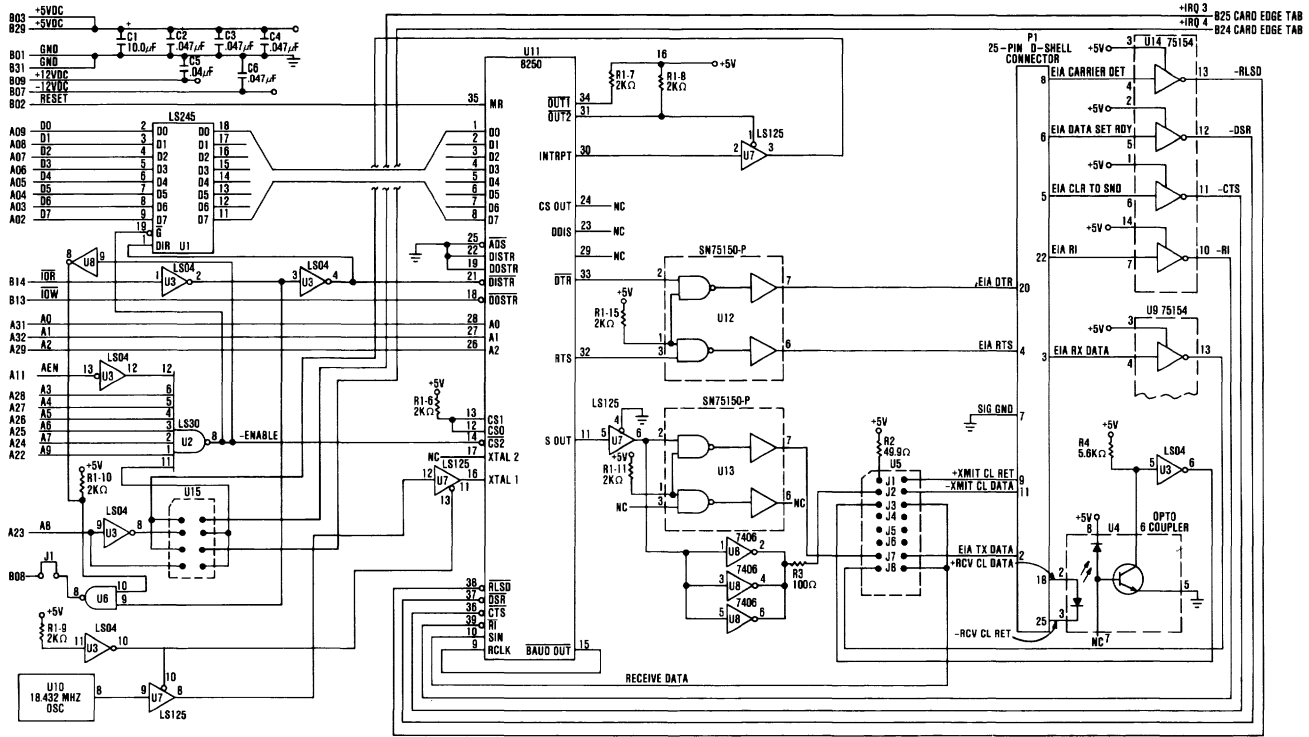
D-76 Logic Diagrams



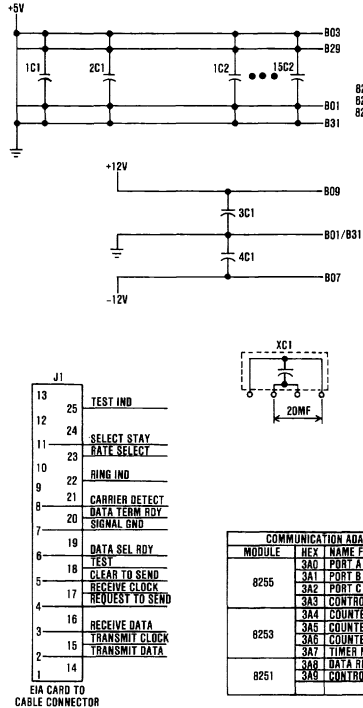
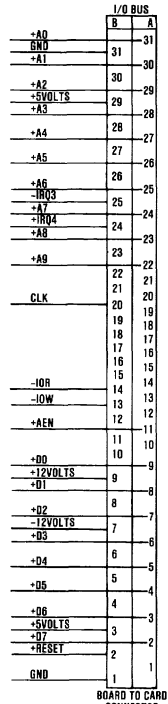
Game Control Adapter (Sheet 1 of 1)



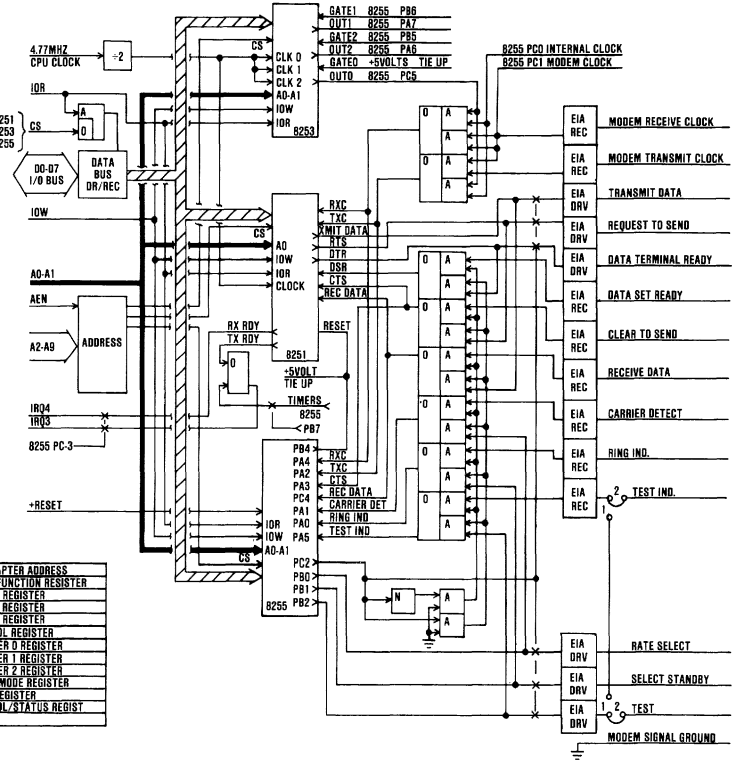
Prototype Card (Sheet 1 of 1)



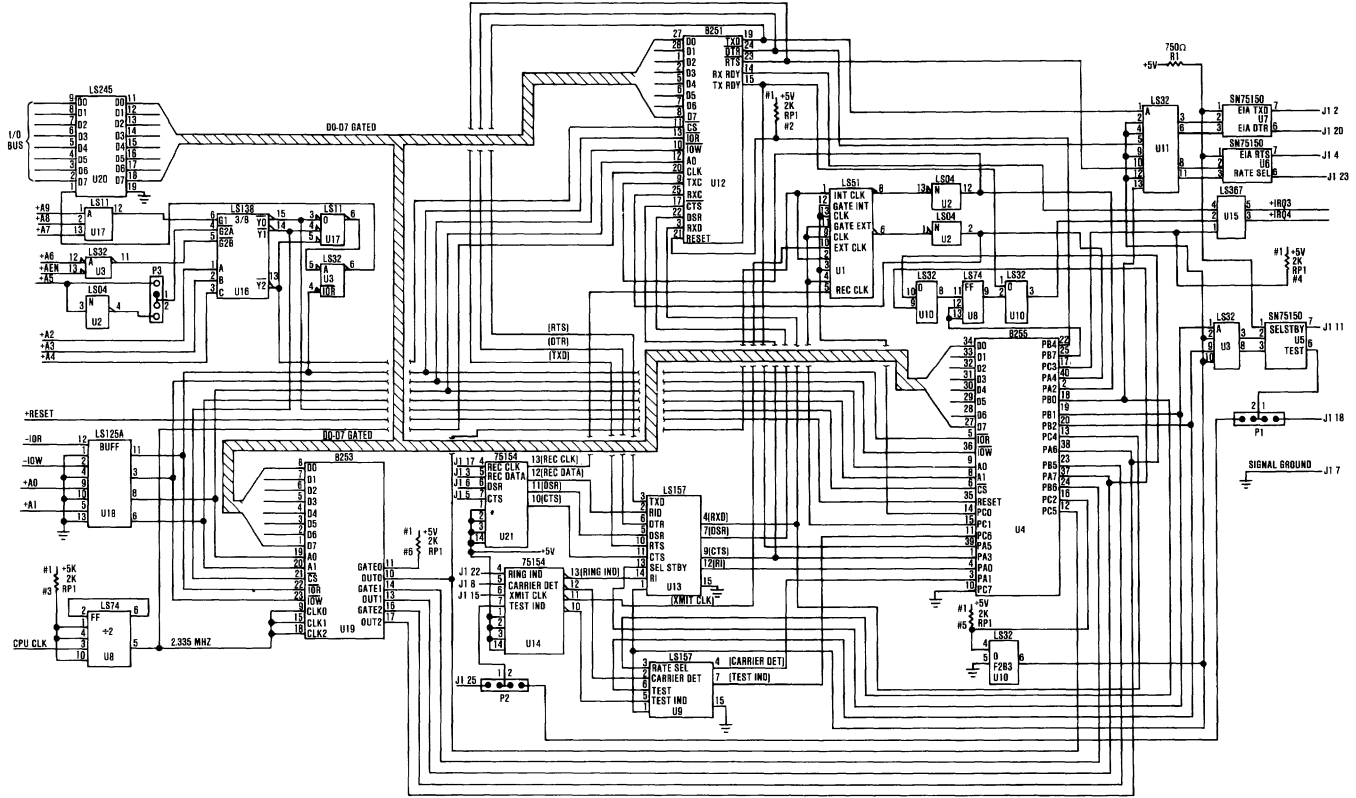
Asynchronous Communications Adapter (Sheet 1 of 1)



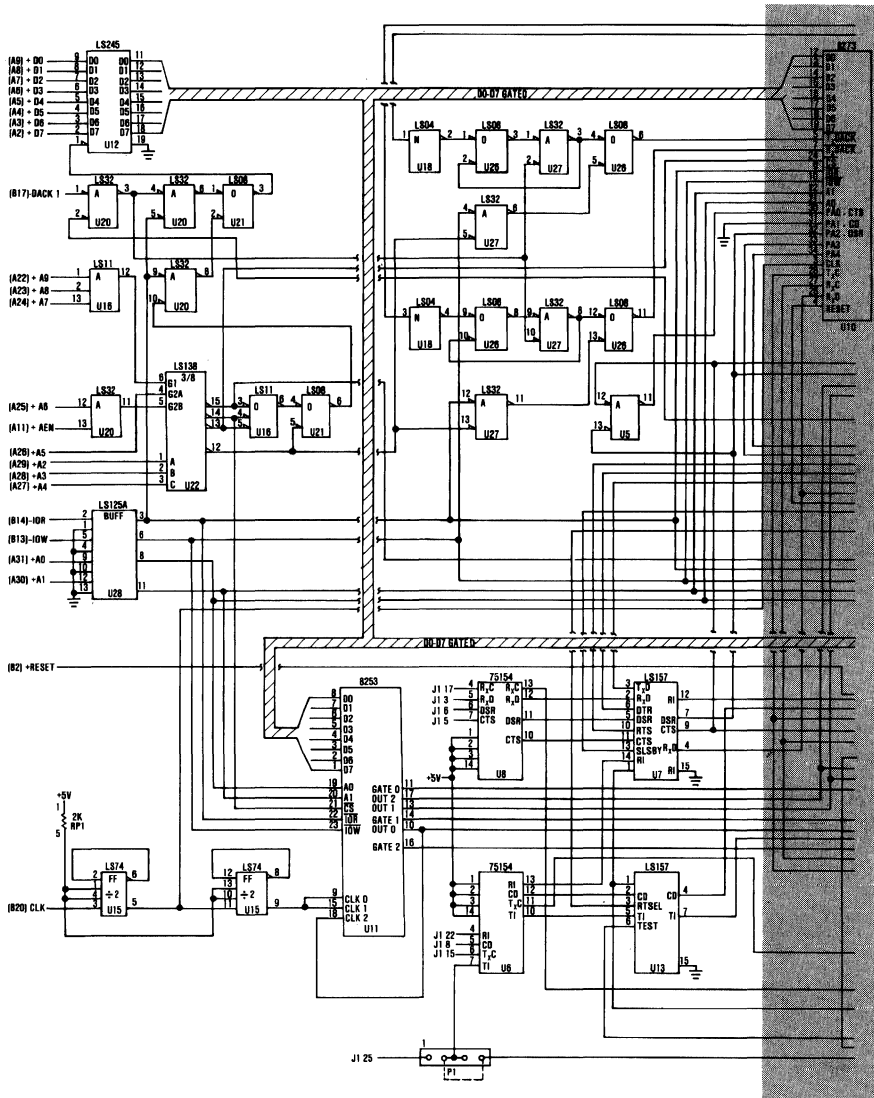
COMMUNICATION ADAPTER ADDRESS			
MODULE	HEX	NAME	FUNCTION REGISTER
8255	3A0	PBRT A REGISTER	
	3A1	PBRT B REGISTER	
	3A2	PBRT C REGISTER	
	3A3	CONTROL REGISTER	
8253	3A4	COUNTER 0 REGISTER	
	3A5	COUNTER 1 REGISTER	
	3A6	COUNTER 2 REGISTER	
	3A7	TIMER MODE REGISTER	
8251	3A8	DATA REGISTER	
	3A9	CONTROL/STATUS REGIST	



Binary Synchronous Communications Adapter (Sheet 1 of 2)



Binary Synchronous Communications Adapter (Sheet 2 of 2)



SDLC Communications Adapter (Sheet 2 of 2)

Notes:

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:

32 lb (14.5 kb)

Power Cables:

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:

717 BTU/hr

Noise Level:

49.5 dB(a) (System unit with monochrome display and expansion unit attached.)

Electrical:

Nominal--120 Vac

Minimum--104 Vac

Maximum--127 Vac

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 Cables:

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 Cables:

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

Minimum--104 Vac

Maximum--127 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 (.914 m)

Size--18 AWG

Signal Cable:

Length--4 ft (1.22 m)

Size--22 AWG

80 CPS Printers

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 m)

Size--22 AWG

Heat Output:

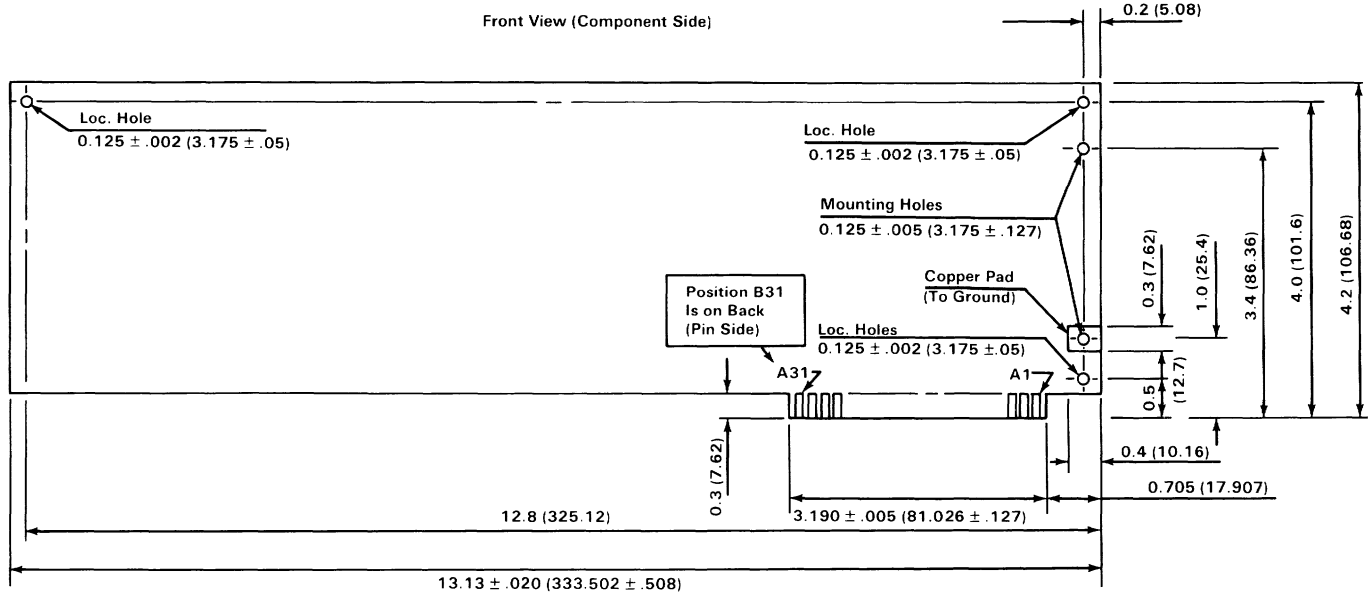
341 BTU/hr (maximum)

Electrical:

Nominal--120 Vac

Minimum--104 Vac

Maximum--127 Vac



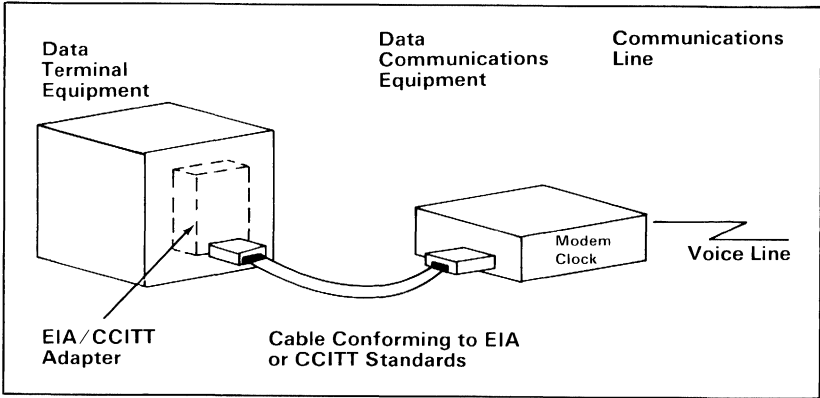
Notes:

- All Card Dimensions are $\pm .010$ (.254) Tolerance (With Exceptions Indicated on Drawing or in Notes).
- Max. Card Length is 13.15 (334.01) Smaller Length is Permissible.
- Loc. and Mounting Holes are Non-Plated Thru. (Loc. 3X, Mtg. 2X).
- 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.
- 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:



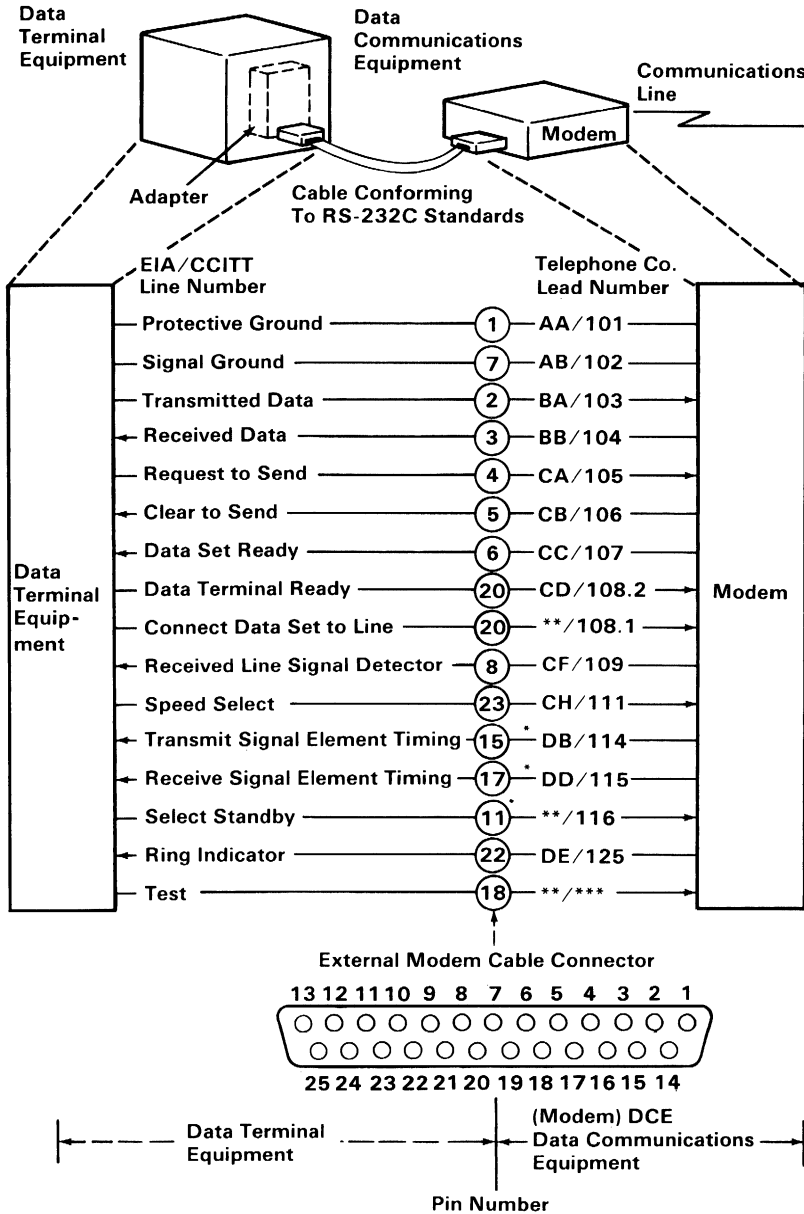
The EIA/CCITT adapter allows data terminal equipment to be connected to data communications equipment using EIA or CCITT standardized connections. An external modem is shown in this example; however, other types of data communications equipment can also be connected to data terminal equipment using EIA 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:



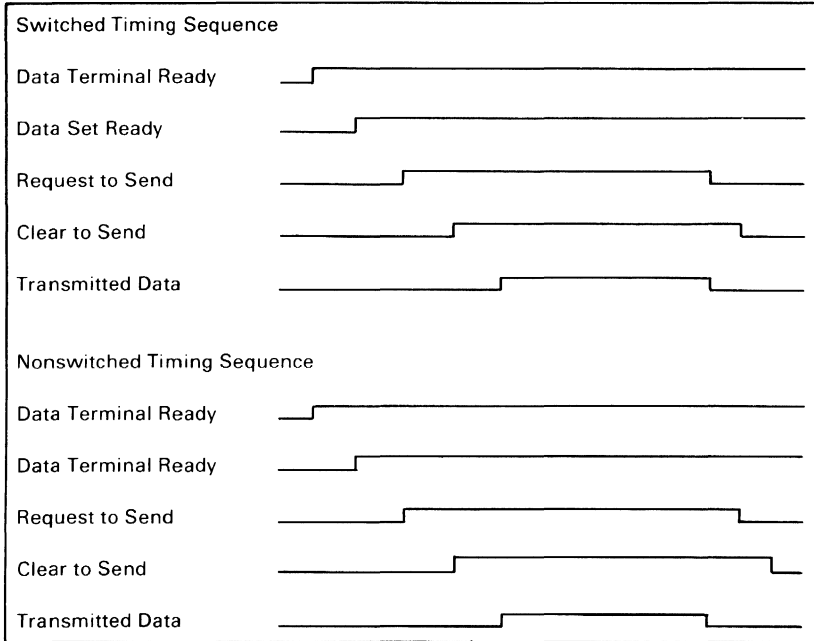
*Not used when business machine clocking is used.

**Not standardized by EIA (Electronics Industry Association).

***Not standardized by CCITT

Establishing a Communications Link

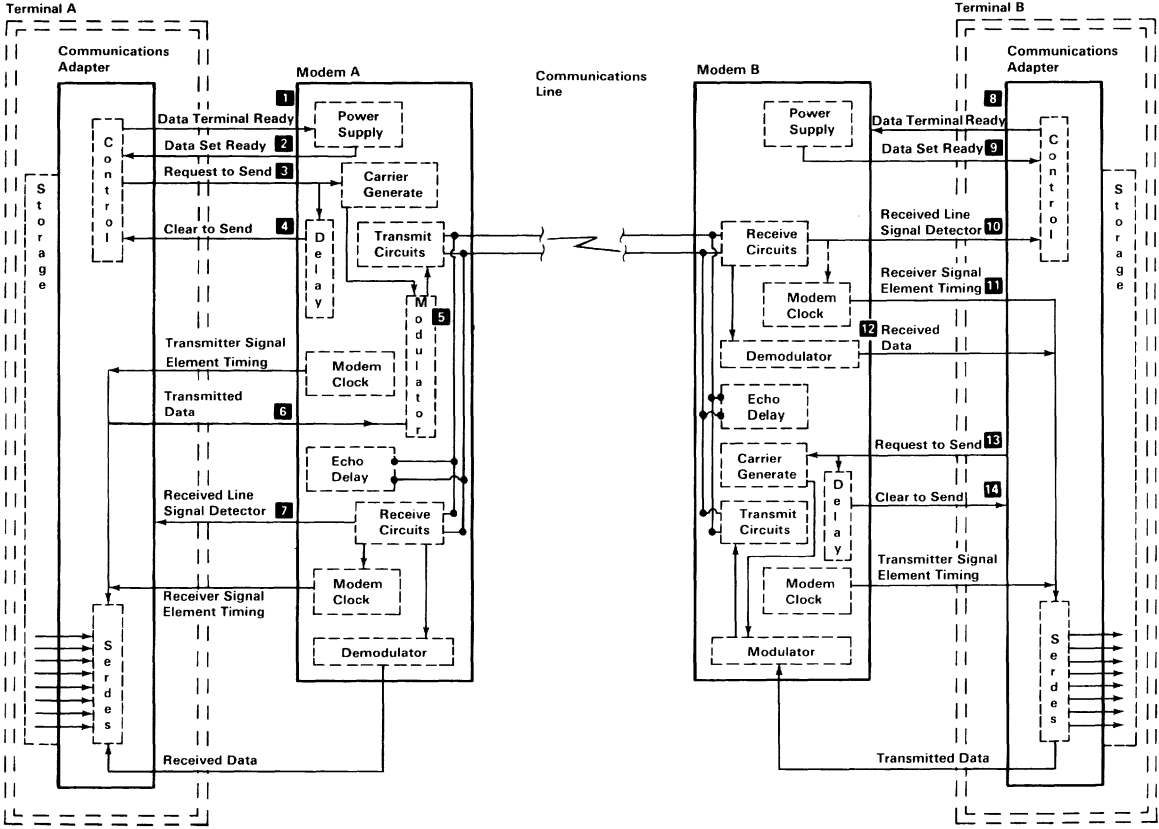
The following bar graphs represent normal timing sequences of operation during the establishment of communications for both switched (dial-up) and nonswitched (direct line) networks.



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.

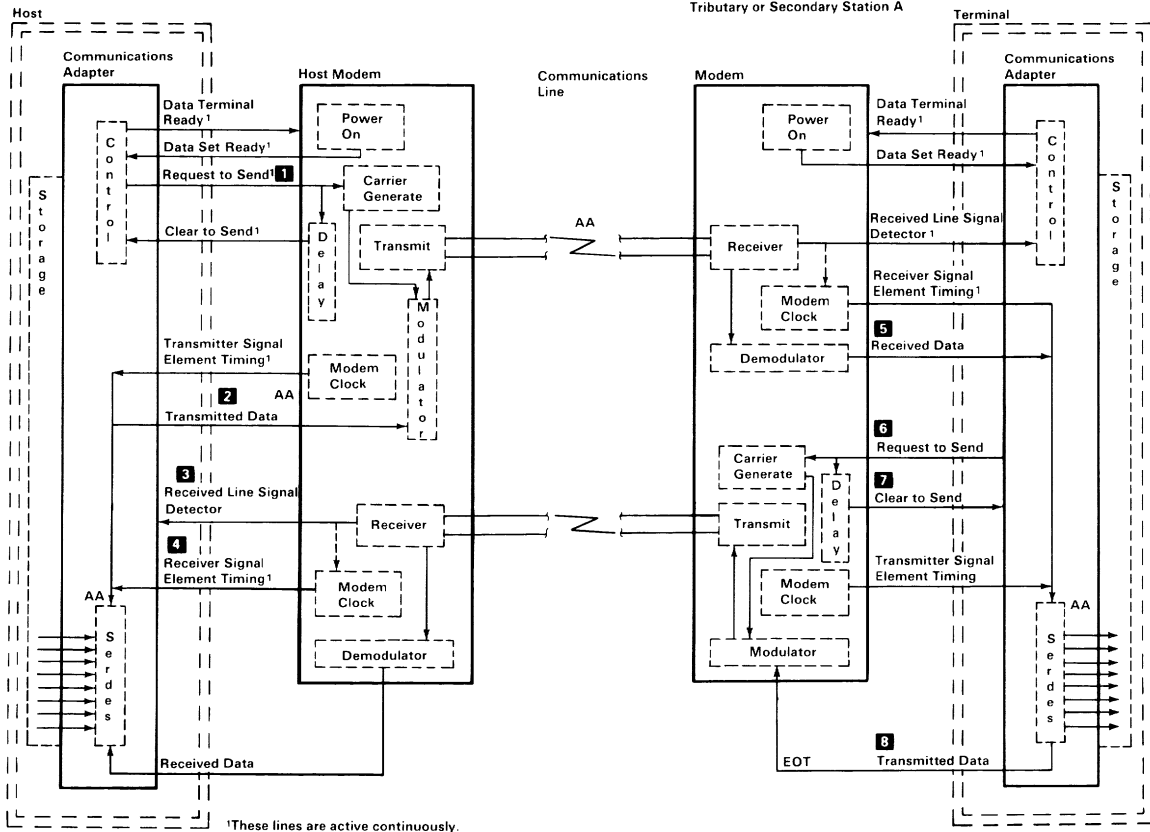
Establishing a Link on a Nonswitched Point-to-Point Line

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

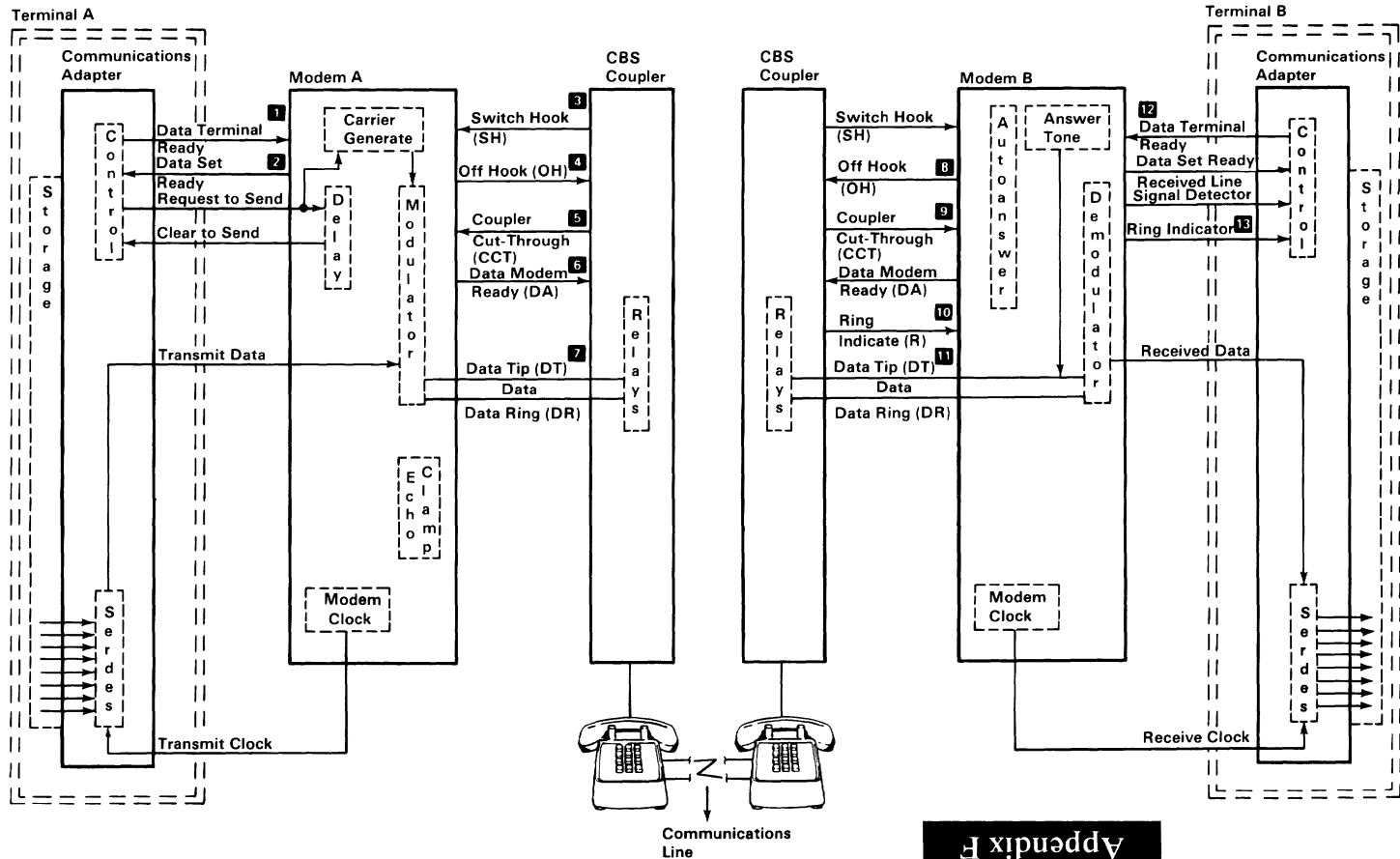
1. The control station serializes the address for the tributary or secondary station (AA) and sends its address to the modem on the 'transmitted data' line **2**.
2. Since the 'request to send' line and, therefore, the modem carrier, is active continuously **1**, the modem immediately modulates the carrier 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 through 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:

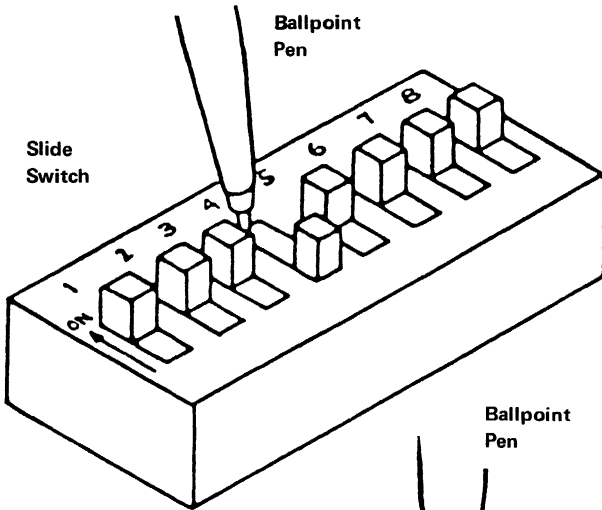
APPENDIX G: SWITCH SETTINGS

System Board Switch Settings	G-3
System Board Switch	G-3
Math Coprocessor Switch Setting	G-3
System Board Memory Switch Settings	G-4
Monitor Type Switch Settings	G-4
5-1/4" Diskette Drive Switch Settings	G-5
Extender Card Switch Settings	G-6
Memory Option Switch Settings	G-7
288K Total Memory	G-7
320K Total Memory	G-8
352K Total Memory	G-9
384K Total Memory	G-10
416K Total Memory	G-11
448K Total Memory	G-12
480K Total Memory	G-13
512K Total Memory	G-14
544K Total Memory	G-15
576K Total Memory	G-16
608K Total Memory	G-17
640K Total Memory	G-18

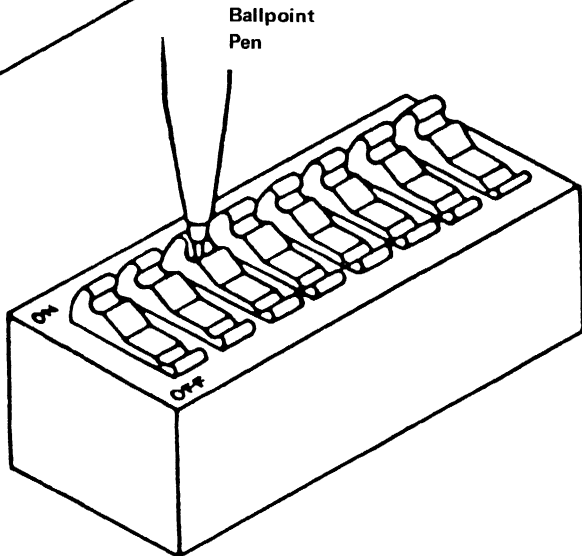
Switches in your system are set to reflect the addition of memory and other installed options. Switches are located on the system board, extender card, and memory expansion options.

The switches are dual inline pin (dip) switches that can be easily set with a ballpoint pen. Refer to the diagrams below to familiarize yourself with the different types of switches that may be used in your system.

Refer to the charts on the following pages to determine the correct switch settings for your system.



or
Rocker
Switch

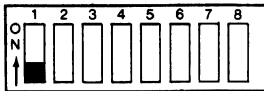


Note: Set a rocker switch by pressing down the rocker to the desired position.

G-2 Switch Settings

System Board Switch Settings

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

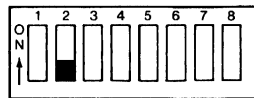


Position	Function
1	Normal operation, Off (set to On to loop POST)
2	Used for Math Coprocessor
3-4	Amount of memory on the system board
5-6	Type of monitor you are using
7-8	Number of 5-1/4 inch diskette drives attached

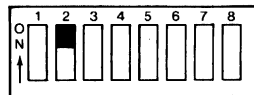
Math Coprocessor Switch Settings

The following figure shows the settings for position 2.

Math Coprocessor installed

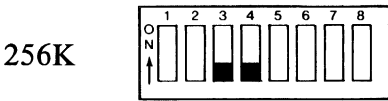
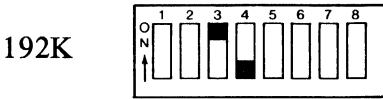
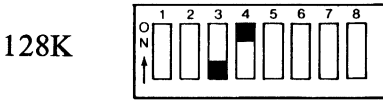


Math Coprocessor not installed



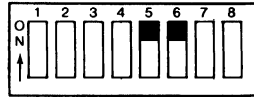
System Board Memory Switch Settings

The following figure shows the settings for positions 3 and 4 for the amount of memory on the system board.

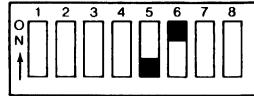


Monitor Type Switch Settings

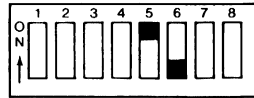
No Monitor



IBM Color Display or other color monitor in the 40x25 Color mode

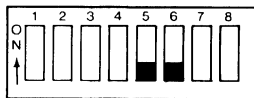


IBM Color Display or other color monitor in the 80x25 Color mode

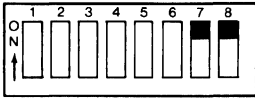


Note: The 80x25 color setting, when used with your television and other monitors, can cause loss of character quality.

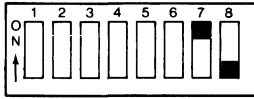
IBM Monochrome Display or more than one monitor



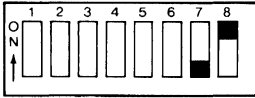
5 1/4" Diskette Drive Switch Settings



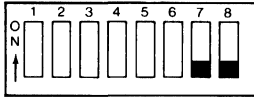
1 DRIVE



3 DRIVES

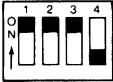
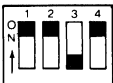
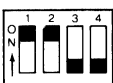
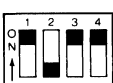
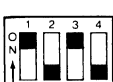

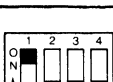
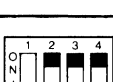
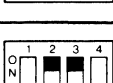



2 DRIVES




4 DRIVES

Extender Card Switch Settings




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

Memory Option Switch Settings




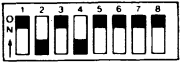
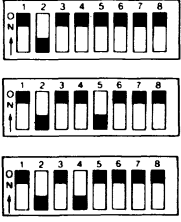
288K Total Memory
32K + (256K on System Board)

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 32K option			





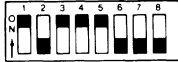


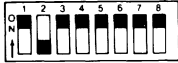



**320K Total Memory
64K + (256K on System Board)**

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 64K installed			
1 - 64K option			
2 - 32K options			

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

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 64K installed 1 - 32K option			
1 - 64K option 1 - 32K option			
3 - 32K options			

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

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 64K option 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			


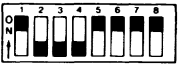



**416K Total Memory
160K + (256K on System Board)**

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
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			

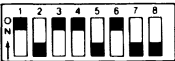
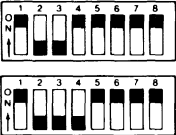
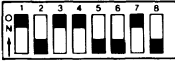


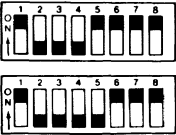

**448K Total Memory
192K + (256K on System Board)**

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
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 128 installed 2 - 32K options			

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

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 192K installed 1 - 32K option			
1 - 64/256K option with 128K installed 1 - 64K option 1 - 32K option			

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

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
<p>1 - 64/256K option with 128K installed 2 - 64K options</p>			
<p>1 - 64/256K option with 192K installed 1 - 64K option</p>			
<p>1 - 64/256K option with 192K installed 2 - 32K options</p>			
<p>1 - 64/256K option with 256K installed</p>			

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

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
1 - 64/256K option with 192K installed 1 - 64K option 1 - 32K option			
1 - 64/256K option with 256K installed 1 - 32K option			

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

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
<p>1 - 64/256K option with 192K installed 2 - 64K options</p>			
<p>1 - 64/256K option with 256K installed 1 - 64/256K option with 64K installed</p>			
<p>1 - 64/256K option with 256K installed 1 - 64K option</p>			
<p>1 - 64/256K option with 256K installed 2 - 32K options</p>			

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

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
<p>1 - 64/256K option with 256K installed 1 - 64/256K option with 64K installed 1 - 32K option</p>			
<p>1 - 64/256K option with 256K installed 1 - 64K option 1 - 32K option</p>			

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

	64/256K Option Card Switches	64K Option Card Switches	32K Option Card Switches
<p>1 - 64/256K option with 256K installed 1 - 64/256K option with 64K installed 1 - 64K option</p>			
<p>1 - 64/256K option with 256K installed 2 - 64K options</p>			
<p>1 - 64/256K option with 256K installed 1 - 64/256K option with 128K installed</p>			

GLOSSARY

μ s: 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 alphanumeric.

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.

binary synchronous communications (BSC): A standardized procedure, using a set of control characters and control character sequences for synchronous transmission of binary-coded data between stations.

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.

BSC: Binary synchronous communications.

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 Telegraphique 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 a “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: 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: 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 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 (SLDC): 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) 1-123
- A0-A19 (Address Bits 0 to 19), I/O channel 1-15
- adapter card with ROM 2-10
- adapter,
 - asynchronous communication 1-215
 - binary synchronous communication 1-245
 - color/graphics monitor 1-123
 - diskette drive 1-151
 - fixed disk drive 1-179
 - game control 1-203
 - monochrome display and printer 1-115
 - printer 1-113
 - synchronous data link control 1-265
- Address Bits 0 to 19 (A0-A19), I/O channel 1-15
- address bits (asynchronous communication) 1-217
- Address Enable (AEN), I/O channel 1-18
- Address Latch Enable (ALE), I/O channel 1-15
- address map, I/O 1-8
- AEN (Address Enable), I/O channel 1-18
- ALE (Address Latch Enable), I/O channel 1-15
- all points addressable mode 1-129, 1-123
- alphanumeric mode, 1-128
 - high resolution 1-135
 - low resolution 1-132
- alt (keyboard extended code) 2-15
- APA mode (all points addressable mode) 1-124
- asynchronous communications adapter, 1-215
 - adapter address jumper module 1-242
 - address bits 1-217
 - block diagram 1-216
 - connector specifications 1-243
 - current loop interface 1-219
 - divisor latch least significant bit 1-229
 - divisor latch most significant bit 1-230
 - I/O decode 1-217
 - INS8250 functional pin description 1-221
 - INS8250 input signals 1-221
 - INS8250 input/output signals 1-225

- INS8250 output signals 1-224
- interface descriptions 1-218
- interface format jumper module 1-242
- interrupt control functions 1-234
- interrupt enable register 1-235
- interrupt identification register 1-233
- interrupts 1-218
- line control register 1-227
- line status register 1-231
- modem control register 1-236
- modem status register 1-238
- modes of operation 1-216
- programmable baud rate generator 1-229
- programming considerations 1-226
- receiver buffer register 1-240
- reset functions 1-226
- transmitter holding register 1-241
- voltage interchange information 1-220
- attributes, character
 - (see character attributes)

B

- BASIC reserved interrupts 2-7
- BASIC,
 - DEF SEG 2-8
 - reserved interrupt 2-7
 - screen editor keyboard functions 2-20
 - workspace variables 2-8
- baud rate generator 1-231
- bell (printer) 1-92
- bibliography I-1
- binary synchronous communications adapter, 1-245
 - 8252A programming procedures 1-257
 - 8252A universal synchronous/asynchronous
 - receiver/transmitter 1-241
 - 8253-5 programmable interval timer 1-251
 - 8255A-5 programmable peripheral interface 1-251
- block diagram 1-246
- command instruction format 1-258
- connector information 1-257
- data bus buffer 1-247
- interface signal information 1-260
- interrupt information 1-262
- mode instruction definition 1-257

- read/write control logic 1-247
- receive 1-252
- receiver buffer 1-249
- receiver control 1-249
- status read definition 1-259
- transmit 1-251
- transmitter buffer 1-248
- transmitter control 1-249
- typical programming sequence 1-253
- BIOS,**
 - fixed disk ROM A-84
 - memory map 2-9
 - parameter passing 2-3
 - software interrupt listing 2-4
 - system ROM A-2
 - use of 2-2
- bisync communications
(see binary synchronous communications adapter)
- block diagram**
 - 8252A universal synchronous/asynchronous receiver/transmitter 1-246
 - 8273 SDLC protocol controller 1-267
 - asynchronous communications adapter 1-215
 - color/graphics monitor adapter 1-125
 - coprocessor 1-25
 - diskette drive adapter 1-151
 - expansion board 1-71
 - extender card 1-74
 - fixed disk drive adapter 1-179
 - game control adapter 1-203
 - keyboard interface 1-67
 - monochrome display adapter 1-114
 - printer adapter 1-108
 - prototype card 1-118
 - receiver card 1-77
 - speaker drive system 1-20
 - synchronous data link control adapter 1-265
 - system 1-2
- break (keyboard extended code) 2-14
- BSC adapter**
(see binary synchronous communications)

C

cable

- communications adapter 1-285
- expansion unit 1-171
- printer 1-81

cancel (printer) 1-93

cancel ignore paper end (printer) 1-95

cancel skip perforation (printer) 1-99

caps lock (keyboard extended code) 2-16

card dimensions and specifications E-4

card selected 1-19

CARD SLCTD (card selected), I/O channel 1-19

card,

- dimensions and specifications E-4
- extender 1-74
- prototype 1-209
- receiver 1-77

carriage return (printer) 1-92

CCITT, F-1

- standards F-1

character attributes

- color/graphics monitor adapter 1-130
- monochrome display adapter 1-130

character codes

- keyboard 2-11

character set,

- graphics printer (set 1) 1-103
- graphics printer (set 2) 1-105
- matrix printer 1-101
- quick reference C-12

clear printer buffer (printer) 1-100

CLK (system clock), I/O channel 1-16

color display 1-149

- operating characteristics 1-150
- specifications E-2

color select register 1-141

color/graphics monitor adapter 1-123

- 6845 register description 1-136
- alphanumeric mode 1-127
- alphanumeric mode (high-resolution) 1-128
- alphanumeric mode (low-resolution) 1-128
- block diagram 1-126
- character attributes 1-130
- color-select register 1-141
- composite connector specifications 1-146

- connector specifications 1-146
- direct-drive connector specifications 1-146
- display buffer basic operation 1-127
- graphics mode 1-132
- graphics mode (high resolution) 1-135
- graphics mode (low resolution) 1-132
- graphic mode (medium resolution) 1-133
- light pen connector specifications 1-147
- major components 1-126
- memory requirements 1-145
- mode control and status register 1-143
- mode register summary 1-143
- mode select register 1-141
- programming considerations 1-137
- RF modulator connector specifications 1-147
- sequence of events 1-144
- status register 1-143
- summary of available color 1-135
- colors, summary of available 1-137
- command status register 0 1-164
- command status register 1 1-165
- command status register 2 1-166
- command status register 3 1-167
- command summary,
 - diskette drive adapter 1-151
 - fixed disk drive adapter 1-179
- communications adapter cable 1-285
 - connector specifications 1-286
- communications F-1
 - establishing a link F-3
- component diagram,
 - system board 1-13
- compressed (printer) 1-93
- compressed off (printer) 1-93
- connector specifications,
 - asynchronous communications adapter 1-215
 - binary synchronous communications 1-245
 - color/graphics monitor adapter 1-147
 - communications adapter cable 1-286
 - diskette drive adapter (external) 1-174
 - diskette drive adapter (internal) 1-173
 - game control adapter 1-203
 - keyboard interface 1-67
 - monochrome display adapter 1-115
 - printer adapter 1-81
 - synchronous data link control adapter 1-293

- connectors,
 - power supply (system unit) 1-21
- consideration, programming
 - (see programming considerations)
- control byte, fixed disk drive adapter 1-186
- control codes, printer 1-91
- control/read/write logic 1-268
- coprocessor,
 - (see math coprocessor)
- ctrl (keyboard extended code) 2-13
- current loop interface 1-219

D

- D0-D7 (data bits 0 to 7), I/O channel 1-16
- DACK0-DACK3 (DMA Acknowledge 0 to 3), I/O channel 1-18
- Data Bits 0 to 7 (D0-D7), I/O channel 1-16
- data flow,
 - system board 1-6
- data register 1-185
- data transfer mode register 1-282
- DEF SEG (default segment workspace) 2-8
- default workspace segment (DEF SEG) 2-8
- diagram, block (see block diagram)
- digital output register 1-153
- diskette drive adapter 1-151
 - adapter input 1-171
 - adapter output 1-170
 - block diagram 1-152
 - command status register 0 1-164
 - command status register 1 1-165
 - command status register 2 1-166
 - command status register 3 1-167
 - command summary 1-158
 - connector specifications (external) 1-174
 - connector specifications (internal) 1-173
 - digital-output register 1-153
 - DPC registers 1-167
 - drive A and B interface 1-170
 - drive constants 1-168
 - FDC constants 1-168
 - floppy disk controller 1-154
 - functional description 1-153
 - programming considerations 1-156
 - programming summary 1-167
 - symbol descriptions 1-156
 - system I/O channel interface 1-168

- diskette drive, 1-175
 - electrical specifications 1-176
 - mechanical specifications 1-176
 - switch settings G-1
- diskettes 1-177
- display adapter type switch settings G-1
- display,
 - color 1-149
 - monochrome 1-121
- divisor latch,
 - least significant bit 1-229
 - most significant bit 1-230
- DMA Acknowledge 0 to 3 (DACK0-DACK3),
 - I/O channel 1-18
- DMA Request 1 to 3 (DRQ1-DRQ3), I/O channel 1-18
- DOS reserved interrupts 2-7
- DOS,
 - keyboard functions 2-20
 - reserved interrupts 2-7
- double strike (printer) 1-96
- double strike off (printer) 1-96
- double width (printer) 1-93
- double width off (printer) 1-93
- DPC registers 1-167
- DRQ1-DRQ3 (DMA Request 1 to 3), I/O channel 1-18

E

- EIA, F-1
 - standards F-1
- emphasized (printer) 1-96
- emphasized off (printer) 1-96
- escape (printer) 1-93
- establishing a communications link F-3
- expansion board, 1-71
 - block diagram 1-72
- expansion channel 1-73
- expansion unit, 1-71
 - cable 1-71
 - expansion board 1-71
 - expansion channel 1-73
 - extender card 1-74
 - interface information 1-79
 - power supply 1-71
 - receiver card 1-77
 - specifications E-2

- extender card, 1-74
 - block diagram 1-76
 - programming considerations 1-75
 - switch settings G-1

F

- FABS (coprocessor) 1-36
- FADD (coprocessor) 1-36
- FBLD (coprocessor) 1-37
- FBSTP (coprocessor) 1-37
- FCHS (coprocessor) 1-38
- FCLEX/FNCLEX (coprocessor) 1-38
- FCOM (coprocessor) 1-38
- FCOMP (coprocessor) 1-39
- FCOMPP (coprocessor) 1-39
- FDECSTP (coprocessor) 1-40
- FDISI/FNDISI (coprocessor) 1-40
- FDIV (coprocessor) 1-41
- FDIVR (coprocessor) 1-42
- FENI/FNENI (coprocessor) 1-43
- FFREE (coprocessor) 1-43
- FICOM (coprocessor) 1-43
- FICOMP (coprocessor) 1-44
- FILD (coprocessor) 1-44
- FINCSTP (coprocessor) 1-44
- FINIT/FNINIT (coprocessor) 1-45
- FIST (coprocessor) 1-46
- FISTP (coprocessor) 1-46
- fixed disk controller 1-179
- fixed disk drive 1-195
- fixed disk drive adapter 1-179
 - block diagram 1-180
 - command summary 1-187
 - control byte 1-186
 - data register 1-185
 - fixed disk controller 1-179
 - interface specifications 1-193
 - programming considerations 1-181
 - programming summary 1-191
 - ROM BIOS listing A-84
 - sense bytes 1-181
 - status register 1-181
 - system I/O channel interface 1-192

fixed disk drive, 1-195
 electrical specifications 1-196
 mechanical specifications 1-196
fixed disk ROM BIOS A-84
FLD (coprocessor) 1-47
FLDCW (coprocessor) 1-47
FLDENV (coprocessor) 1-48
FLDLG2 (coprocessor) 1-48
FLDLN2 (coprocessor) 1-48
FLDL2E (coprocessor) 1-49
FLDL2T (coprocessor) 1-49
FLDPI (coprocessor) 1-49
FLDZ (coprocessor) 1-50
FLD1 (coprocessor) 1-50
floppy disk controller 1-154
form feed (printer) 1-92
FMUL 1-51
FNOP 1-52
FPATAN 1-52
FPREM 1-52
FPTAN 1-53
FRNDINT 1-53
FRSTOR 1-53
FSAVE/FNSAVE (coprocessor) 1-54
FSCALE (coprocessor) 1-54
FSQRT (coprocessor) 1-55
FST (coprocessor) 1-55
FSTCW/FNSTCW (coprocessor) 1-56
FSTENV/FNSTENV (coprocessor) 1-56
FSTP (coprocessor) 1-57
FSTSW/FNSTSW (coprocessor) 1-57
FSUB (coprocessor) 1-58
FSUBR (coprocessor) 1-59
FTST (coprocessor) 1-60
FWAIT (coprocessor) 1-60
FXAM (coprocessor) 1-61
FXCH (coprocessor) 1-62
EXTRACT (coprocessor) 1-62
FYL2X (coprocessor) 1-63
FYL2XP1 (coprocessor) 1-63
F2XM1 (coprocessor) 1-64

G

- game control adapter, 1-203
 - block diagram 1-203
 - connector specifications 1-208
 - functional description 1-204
 - I/O channel description 1-205
 - interface description 1-206
 - joy stick schematic diagram 1-207
- glossary, H-1
- graphics mode, 1-123
 - high resolution 1-124
 - low resolution 1-123
 - medium resolution 1-124

H

- hardware interrupt listing 1-9
- home head (printer) 1-95
- horizontal tab (printer) 1-92

I

- I/O address map 1-8
- I/O bit map, 8255A 1-10
- I/O CH CK (I/O Channel Check), I/O channel 1-17
- I/O CH RDY (I/O Channel Ready), I/O channel 1-17
- I/O Channel Check (I/O CH CK), I/O channel 1-17
- I/O channel interface,
 - diskette drive adapter 1-168
 - fixed disk drive adapter 1-192
 - prototype card 1-211
- I/O Channel Ready (I/O CH RDY), I/O channel 1-17
- I/O channel, 1-14
 - I/O Channel Check (I/O CH CK) 1-17
 - I/O Read Command (IOR) 1-17
 - I/O Write Command (IOW) 1-17
 - Address Bits 0 to 19 (A0-A19) 1-16
 - Address Enable (AEN) 1-18
 - Address Latch Enable (ALE) 1-16
 - Data Bits 0 to 7 (D0-D7) 1-16
 - description 1-16
 - diagram 1-15
 - DMA Request 1 to 3 (DRQ1-DRQ3) 1-18
 - I/O Channel Ready (I/O CH RDY) 1-17

- Interrupt Request 2 to 7 (IRQ2-IRQ7) 1-17
- Memory Read Command (MEMR) 1-18
- Memory Write Command (MEMW) 1-18
- Oscillator (OSC) 1-16
- Reset Drive (RESET DRV) 1-16
- System Clock (CLK) 1-16
- Terminal Count (T/C) 1-18
- I/O Read Command (IOR), I/O channel 1-17
- I/O Write Command (IOW), I/O channel 1-17
- IBM 10MB Fixed Disk Drive 1-195
- IBM 5-1/4" Diskette Drive 1-175
- IBM 5-1/4" Diskette Drive Adapter 1-151
- IBM 80 CPS Graphics Printer 1-81
- IBM 80 CPS Matrix Printer 1-81
- IBM 80 CPS Printers 1-81
- IBM Asynchronous Communications Adapter 1-215
- IBM Binary Synchronous Communications Adapter 1-245
- IBM Color Display 1-149
- IBM Color/Graphics Monitor Adapter 1-123
- IBM Communicatons Adapter Cable 1-295
- IBM Fixed Disk Drive Adapter 1-179
- IBM Game Control Adapter 1-203
- IBM Memory Expansion Options 1-197
- IBM Monochrome Display and Printer Adapter 1-113
- IBM Monochrome Display 1-121
- IBM Personal Computer Math Coprocessor 1-25
- IBM Printer Adapter 1-107
- IBM Prototype Card 1-209
- IBM Synchronous Data Link Controller Adapter 1-265
- ignore paper end (printer) 1-94
- 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
 - operand summary B-3
 - processor control B-14
 - register model B-2

- second instruction byte summary B-3
- segment override prefix B-4
- software interrupt listing 2-4
- string manipulation B-11
- 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 1-10
- 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 1-265
- interrupt enable register 1-235
- interrupt identification register 1-233
- interrupt listing,
 - 8088 hardware 1-9
 - 8088 software 2-4
- Interrupt Request 2 to 7 (IRQ2-IRQ7), I/O channel 1-17
- interruptions,
 - 8088 hardware 1-9
 - 8088 software 2-4
 - asynchronous communications adapter 1-215
 - BASIC reserved 2-7
 - DOS reserved 2-7
 - special 2-5
- IOR (I/O Read Command), I/O channel 1-17
- IOW (I/O Write Command), I/O channel 1-17
- IRQ2-IRQ7 (Interrupt Request 2 to 7), I/O channel 1-17

J

- joy stick,
 - positions 1-204
 - schematic diagram 1-207
- jumper module, asynchronous communications adapter 1-242

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 1-65
 - BASIC screen editor special functions 2-20
 - character codes 2-11
 - commonly used functions 2-18
 - diagram 1-68
 - DOS special functions 2-20
 - encoding 2-11
 - extended functions 2-14
 - interface block diagram 1-70
 - interface connector specifications 1-70
 - scan codes 1-69
 - specifications E-1

L

- light pen connector specifications 1-147
- line control register 1-227
- line feed (printer) 1-92
- line status register 1-223
- logic diagrams D-1

M

- math coprocessor 1-25
 - block diagram 1-29
 - control unit 1-29
 - control word 1-32
 - data types 1-26
 - exception pointers 1-33
- FABS 1-36
- FADD 1-36
- FBLD 1-37
- FBSTP 1-37
- FCHS 1-38
- FCLEX/FNCLEX 1-38
- FCOM 1-38
- FCOMP 1-39
- FCOMPP 1-39
- FDECSTP 1-40

FDISI/FNDISI 1-40
FDIV 1-41
FDIVR 1-42
FENI/FNENI 1-43
FFREE 1-43
FICOM 1-43
FICOMP 1-44
FILD 1-44
FINCSTP 1-44
FINIT/FNINIT 1-45
FIST 1-46
FISTP 1-46
FLD 1-47
FLDCW 1-47
FLDENV 1-48
FLDLG2 1-48
FLDLN2 1-48
FLDL2E 1-49
FLDL2T 1-49
FLDPI 1-49
FLDZ 1-50
FLD1 1-50
FMUL 1-51
FNOP 1-52
FPATAN 1-52
FPREM 1-52
FPTAN 1-53
FRNDINT 1-53
FRSTOR 1-53
FSAVE/FNSAVE 1-54
FSCALE 1-54
FSQRT 1-55
FST 1-55
FSTCW/FNSTCW 1-56
FSTENV/FNSTENV 1-56
FSTP 1-57
FSTSW/FNSTSW 1-57
FSUB 1-58
FSUBR 1-59
FTST 1-60
FWAIT 1-60
FXAM 1-61
FXCH 1-62
FXTRACT 1-62
FYL2X 1-63
FYL2XP1 1-63
F2XM1 1-64

- hardware interface 1-27
- instruction set 1-35
- interconnection 1-28
- number system 1-34
- programming interface 1-26
- register stack 1-30
- status word 1-31
- tag word 1-33
- memory expansion options, 1-197
 - DIP module start address 1-201
 - memory module description 1-198
 - memory module pin configuration 1-199
 - memory option switch settings G-1
 - R/W memory operating characteristics 1-198
 - switch-configurable start address 1-200
- memory locations,
 - reserved 2-8
- memory map,
 - BIOS 2-9
 - system 1-11
- Memory Read Command (MEMR), I/O channel 1-18
- memory switch settings, G-1
 - extender card G-1
 - memory options G-1
 - system board G-1
- 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 1-139
- mode select register 1-141
- modem control register 1-236
- modem status register 1-238
- monochrome display 1-121
- monochrome display and printer adapter 1-113
- monochrome display adapter 1-115
 - 6845 CRT control port 1-118
 - 6845 CRT status port 1-118
 - block diagram 1-114
 - character attributes 1-117
 - connector specifications 1-119
 - I/O address and bit map 1-117
 - programming considerations 1-115
- monochrome display, 1-121
 - operating characteristics 1-122
 - 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 1-221
input signals 1-221
input/output signals 1-225
output signals 1-224
null (printer) 1-92

O

one bit delay mode register 1-283
operating mode register 1-280
OSC (oscillator) 1-16
Oscillator (OSC), I/O channel 1-16

P

parameter passing (ROM BIOS) 2-3
pause (keyboard extended code) 2-17
power good signal 1-24
power supply 1-21
connectors 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, 1-107
block diagram 1-108
connector specifications 1-111
programming considerations 1-109
printer control codes, 1-91
1/8-inch line feeding 1-94
1920 bit-image graphics mode 1-101
480 bit-image graphics mode 1-97
7/72-inch line feeding 1-94
960 bit-image graphics mode 1-99

960 bit-image graphics mode normal speed 1-100
 bell 1-92
 cancel 1-93
 cancel ignore paper end 1-95
 cancel skip perforation 1-99
 carriage return 1-92
 clear printer buffer 1-100
 compressed 1-93
 compressed off 1-93
 double strike 1-96
 double strike off 1-97
 double width 1-99, 1-93
 double width off 1-93
 emphasized 1-96
 emphasized off 1-96
 escape 1-93
 form feed 1-92
 home head 1-91
 horizontal tab 1-92
 ignore paper end 1-94
 line feed 1-92
 null 1-92
 printer deselected 1-93
 printer selected 1-93
 select character set 1 1-94
 select character set 2 1-94
 set horizontal tab stops 1-96
 set lines per page 1-96
 set skip perforation 1-99
 set variable line feeding 1-97
 set vertical tabs 1-95
 starts variable line feeding 1-94
 subscript/superscript 1-99
 subscript/superscript off 1-99
 underline 1-94
 unidirectional printing 1-99
 vertical tab 1-92
 printer deselected (printer) 1-93
 printer selected (printer) 1-93
 printer, 1-81
 additional specifications 1-83
 cable 1-81
 connector pin assignment 1-87
 control codes 1-91
 graphic character set 1 1-103
 graphic character set 2 1-105
 interface signal descriptions 1-87

- matrix character set 1-101
- modes 1-90
- parallel interface 1-86
- parallel interface timing diagram 1-86
- specifications 1-82, E-3
- switch locations 1-84
- switch settings 1-84
- processor (see Intel 8088 microprocessor)
- programmable baud rate generator 1-229
- programming considerations,
 - asynchronous communications adapter 1-226
 - binary synchronous communications adapter 1-253
 - color/graphics monitor adapter 1-137
 - diskette drive adapter 1-151
 - extender card 1-74
 - fixed disk drive adapter 1-179
 - monochrome display adapter 1-115
 - printer adapter 1-109
 - receiver card 1-77
 - SDLC adapter 1-275
- prototype card, 1-209
 - block diagram 1-210
 - external interface 1-213
 - I/O channel interface 1-211
 - layout 1-211
 - system loading and power limitations 1-213

Q

- quick reference, character set C-12

R

- receiver buffer register 1-249
- receiver card, 1-77
 - block diagram 1-78
 - programming considerations 1-77

- register,
 - 6845 description (color/graphic adapter) 1-139
 - color select (color/graphic adapter) 1-140
 - command status 0 (diskette drive adapter) 1-167
 - command status 1 (diskette drive adapter) 1-168
 - command status 2 (diskette drive adapter) 1-169
 - command status 3 (diskette drive adapter) 1-170
 - data (fixed disk drive adapter) 1-185
 - data transfer mode (SDLC) 1-272
 - digital output (diskette drive adapter) 1-153
 - DPC (diskette drive adapter) 1-167
 - interrupt enable (asynchronous communications) 1-235
 - interrupt identification (asynchronous communications) 1-235
 - line control (asynchronous communications) 1-227
 - line status (asynchronous communications) 1-231
 - mode control and status (color/graphics) 1-139
 - mode select (color/graphics) 1-141
 - modem control (asynchronous communications) 1-236
 - modem status (asynchronous communications) 1-238
 - one-bit delay mode (SDLC) 1-283
 - operating mode (SDLC) 1-280
 - receiver buffer (asynchronous communications) 1-240
 - serial I/O mode (SDLC) 1-282
 - status (color/graphics) 1-143
 - status (fixed disk drive adapter) 1-181
 - transmitter holding (asynchronous communications) 1-241
- reserved interrupts,
 - BASIC and DOS 2-7
- reserved memory locations 2-7
- Reset Drive (RESET DRV), I/O channel 1-16
- RESET DRV (Reset Drive), I/O channel 1-16
- RF modulator connector specifications 1-147
- ROM BIOS, 2-2
 - Fixed Disk A-84
 - System A-2
- ROM, adapter cards with 2-10
- RS-232C,
 - interface standards F-2

S

- scan codes,
 - keyboard 1-65
- scroll lock (keyboard extended code) 2-14
- SDLC (see synchronous data link control)
- select character set 1 (printer) 1-94
- select character set 2 (printer) 1-94
- sense bytes, fixed disk drive adapter 1-181
- serial I/O mode register 1-282
- set horizontal tab stops (printer) 1-96
- set lines per page (printer) 1-96
- set skip perforation (printer) 1-99
- set variable line feeding (printer) 1-95, 1-97
- set vertical tabs (printer) 1-95
- shift (keyboard extended code) 2-13
- shift key priorities (keyboard code) 2-14
- shift states (keyboard extended code) 2-13
- 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 1-82
 - printer (additional) 1-83
 - system unit E-1
- stack area 2-7
- starts variable line feeding (printer) 1-94
- status register, 1-137
 - color/graphics monitor adapter 1-143
 - fixed disk drive adapter 1-181
 - synchronous data link control adapter 1-276
- subscript/superscript (printer) 1-99
- subscript/superscript off (printer) 1-99
- switch settings, G-1
 - diskette drive G-1
 - display adapter type G-1
 - extender card G-1
 - memory options G-1
 - printer 1-84

- system board G-1
- system board memory G-1
- synchronous data link control communications adapter, 1-265
 - 8253-5 interval timer control word 1-279
 - 8253-5 programmable interval timer 1-275
 - 8255A-5 port A assignments 1-274
 - 8255A-5 port B assignments 1-274
 - 8255A-5 port C assignments 1-275
 - 8255A-5 programmable peripheral interface 1-274
 - 8273 command phase flow chart 1-286
 - 8273 commands 1-285
 - 8273 control/read/write registers 1-269
 - 8273 data interfaces 1-270
 - 8273 elements of data transfer interface 1-270
 - 8273 mode register commands 1-283
 - 8273 modem control block 1-271
 - 8273 modem control port A 1-271
 - 8273 modem control port B 1-272
 - 8273 modem interface 1-271
 - 8273 protocol controller operations 1-266
 - 8273 protocol controller structure 1-267
 - 8273 register selection 1-268
 - 8273 SDLC protocol controller block diagram 1-257
 - 8273 transmit/receiver timing 1-283
- block diagram 1-265
- command phase 1-284
- connector specifications 1-293
- control/read/write logic 1-268
- data transfer mode register 1-282
- device addresses 1-291
- execution phase 1-287
- general receive 1-288
- initialization/configuration commands 1-280
- initializing the SDLC adapter 1-278
- interface information 1-292
- interrupt information 1-291
- one bit delay code register 1-283
- operating mode register 1-290
- partial byte received codes 1-290
- processor interface 1-268
- programming considerations 1-275
- protocol control module features 1-266
- protocol controller operations 1-266
- result code summary 1-290
- result phase 1-287

- selective receive 1-289
- serial data timing block 1-273
- serial I/O mode register 1-282
- status register format 1-276
- transmit 1-288
- system block diagram 1-2
- system board, 1-3
 - component diagram 1-13
 - data flow 1-6
 - R/W memory operating characteristics 1-198
 - switch settings G-1
- System Clock (CLK), I/O channel 1-16
- system memory map 1-12
- system reset (keyboard extended code) 2-16
- system ROM BIOS A-2
- system unit, 1-3
 - I/O channel 1-14
 - I/O channel diagram 1-15
 - keyboard interface 1-67
 - 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 1-241

U

- underline (printer) 1-94
- unidirectional printer (printer) 1-99

V

- Vac output,
 - system unit 1-22
- Vdc output,
 - system unit 1-22
- vectors with special meanings 2-5
- vertical tab (printer) 1-92
- voltage interchange,
 - asynchronous communications adapter 1-215

Numerics

- 1/8 inch line feeding (printer) 1-94
- 1920 bit-image graphics mode (printer) 1-100
- 480 bit-image graphics mode (printer) 1-97
- 6845,
 - (see color/graphics monitor adapter)
 - (see monochrome display adapter)
- 7/72 inch line feeding (printer) 1-94
- 8088,
 - (see Intel 8088 microprocessor) 1-4
- 8250,
 - (see asynchronous communications adapter)
- 8253-5,
 - (see synchronous data link control adapter)
- 8255A 1-10
- 8255A-5,
 - (see synchronous data link control adapter)
- 8273,
 - (see synchronous data link control adapter)
- 960 bit-image graphics mode (printer) 1-99
- 960 bit-image graphics mode normal speed (printer) 1-100

Notes:



Reader's Comment Form

TECHNICAL REFERENCE

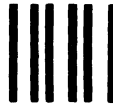
1502237

Your comments assist us in improving the usefulness of our publication; they are an important part of the input used for revisions.

IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Please do not use this form for technical questions regarding the IBM Personal Computer or programs for the IBM Personal Computer, or for requests for additional publications; this only delays the response. Instead, direct your inquiries or request to your authorized IBM Personal Computer dealer.

Comments:

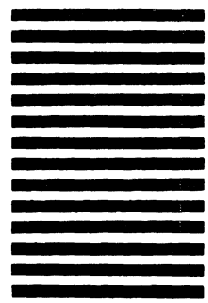


NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 123 BOCA RATON, FLORIDA 33432

POSTAGE WILL BE PAID BY ADDRESSEE

IBM PERSONAL COMPUTER
SALES & SERVICE
P.O. BOX 1328-C
BOCA RATON, FLORIDA 33432



.....
Fold here



Reader's Comment Form

TECHNICAL REFERENCE

1502237

Your comments assist us in improving the usefulness of our publication; they are an important part of the input used for revisions.

IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Please do not use this form for technical questions regarding the IBM Personal Computer or programs for the IBM Personal Computer, or for requests for additional publications; this only delays the response. Instead, direct your inquiries or request to your authorized IBM Personal Computer dealer.

Comments:



NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 123 BOCA RATON, FLORIDA 33432

POSTAGE WILL BE PAID BY ADDRESSEE

IBM PERSONAL COMPUTER
SALES & SERVICE
P.O. BOX 1328-C
BOCA RATON, FLORIDA 33432



.....
Fold here

Please do not staple

Tape



International Business Machines Corporation

P.O. Box 1328-W
Boca Raton, Florida 33432

1502237

Printed in United States of America