Personal Computer Hardware Reference Library

Technical Reference

6361459

Personal Computer Hardware Reference Library

Technical Reference

Revised Edition (April 1984)

The following paragraph does not apply to the United Kingdom or any country where such provisions are inconsistent with local law: International Business Machines Corporation provides this manual "as is," without warranty of any kind, either expressed or implied, including, but not limited to the particular purpose. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this manual at any time.

This product could include technical inaccuracies or typographical errors. Changes are made periodically to the information herein; these changes will be incorporated in new editions of the publication.

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

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.

The following paragraph applies only to the United States and Puerto Rico: A Reader's Comment Form is provided at the back of this publication. If the 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, 1984

Federal Communications Commission Radio Frequency Interference Statement

Warning: The equipment described herein has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of Part 15 of the FCC rules. Only peripherals (computer input/output devices, terminals printers, etc.) certified to comply with the Class B limits may be attached to the computer. Operation with non-certified peripherals is likely to result in interference to radio and TV reception. If peripherals not offered by IBM are used with the equipment, it is suggested to use shielded grounded cables with in-line filters if necessary.

CAUTION
The product described herein is equipped with a grounded plug for the user's safety. It is to be used in conjunction with a properly grounded receptacle to avoid electrical shock.

Preface

This publication describes the various units of the IBM Personal Computer XT and IBM Portable Personal Computer; and the interaction of each.

The information in this publication is for reference, and is intended for hardware and program designers, programmers, engineers, and anyone else with a knowledge of electronics and/or programming who needs to understand the design and operation of the IBM Personal Computer XT or IBM Portable Personal Computer.

This publication consists of two parts: a system manual and an options and adapters manual.

The system manual is divided into the following sections:

Section 1, "System Board," discusses the component layout, circuitry, and function of the system board.

Section 2, "Coprocessor," describes the Intel 8087 coprocessor and provides programming and hardware interface information.

Section 3, "Power Supply," provides electrical input/output specifications as well as theory of operation for both the IBM Personal Computer XT power supply and the IBM Portable Personal Computer power supply.

Section 4, "Keyboard," discusses the hardware make up, function, and layouts of the IBM Personal Computer XT and the IBM Portable Personal Computer keyboards.

Section 5, "System BIOS," 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.

Section 6, "Instruction Set," provides a quick reference for the 8088 assembly instruction set.

Section 7, "Characters, Keystrokes, and Colors," supplies the decimal and hexadecimal values for characters and text attributes.

Section 8, "Communications,"describes communications hardware and discusses communications interface standards and the sequence of events to establish communications.

A glossary, bibliography, and index are also provided.

The *Technical Reference* options and adapters manual provides information, logic diagrams, and specifications pertaining to the options and adapters available for the IBM Personal Computer family of products. The manual is modular in format, with each module providing information about a specific option or adapter. Modules having a large amount of text contain individual indexes. The modules are grouped by type of device into the following categories:

- Expansion Unit
- Displays
- Printers
- Storage Devices
- Memory Expansion
- Adapters
- Miscellaneous
- Cables and Connectors

Full page length hard tabs with the above category descriptions, separate the groups of modules.

The term "*Technical Reference* manual" in the option and adapter manual, refers to the IBM Personal Computer XT/IBM Portable Personal Computer *Technical Reference* system manual.

The term "*Guide to Operations* manual" in the option and adapter manual, refers to either the IBM Personal Computer XT *Guide to Operations* manual or the IBM Portable Personal Computer *Guide to Operations* manual.

Prerequisite Publications

- IBM Personal Computer XT *Guide to Operations*
- IBM Portable Personal Computer *Guide to Operations*

Suggested Reading

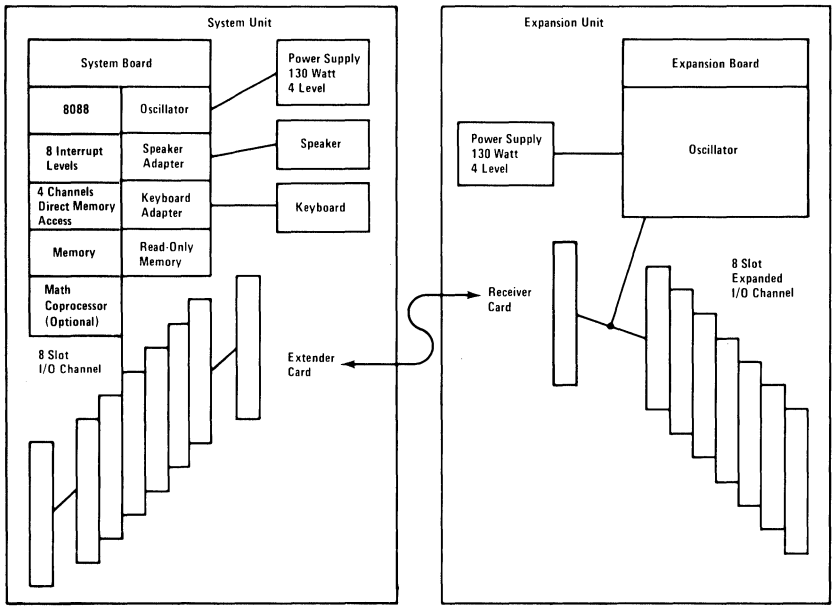
- *BASIC for the IBM Personal Computer*
- *Disk Operating System (DOS)*, Version 2.1
- IBM Personal Computer XT *Hardware Maintenance and Service*
- IBM Portable Personal Computer *Hardware Maintenance and Service*
- *MACRO Assembler for the IBM Personal Computer*

Contents

INDEX TAB LISTING

System Block Diagram (XT)

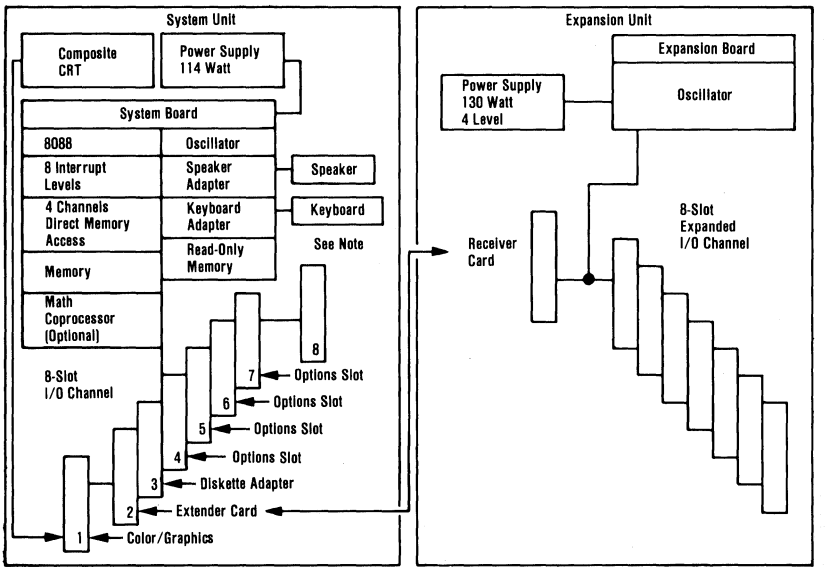
The following is a system block diagram of the IBM Personal Computer XT.



Note: A "System to Adapter Compatibility Chart," to identify the adapters supported by each system, and an "Option to Adapter Compatibility Chart," to identify the options supported by each adapter, can be found in the front matter of the *Technical Reference* options and adapters manual, Volume 1.

System Block Diagram (Portable)

The following is a system block diagram of the IBM Portable Personal Computer.



Note: A "System to Adapter Compatibility Chart," to identify the adapters supported by each system, and an "Option to Adapter Compatibility Chart," to identify the options supported by each adapter, can be found in the front matter of the *Technical Reference* options and adapters manual, Volume 1.

SECTION 1. SYSTEM BOARD

Contents

Description

The system board fits horizontally in the base of the system unit of the Personal Computer XT and Portable Personal Computer and is approximately 215.9 mm by 304.8 mm (8-1/2 x 12 in.). 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 6-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 programs 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 contains the adapter circuits for attaching the serial interface from the keyboard. These circuits generate an interrupt to the microprocessor when a complete scan code is received. The interface can request execution of a diagnostic test in the keyboard.

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

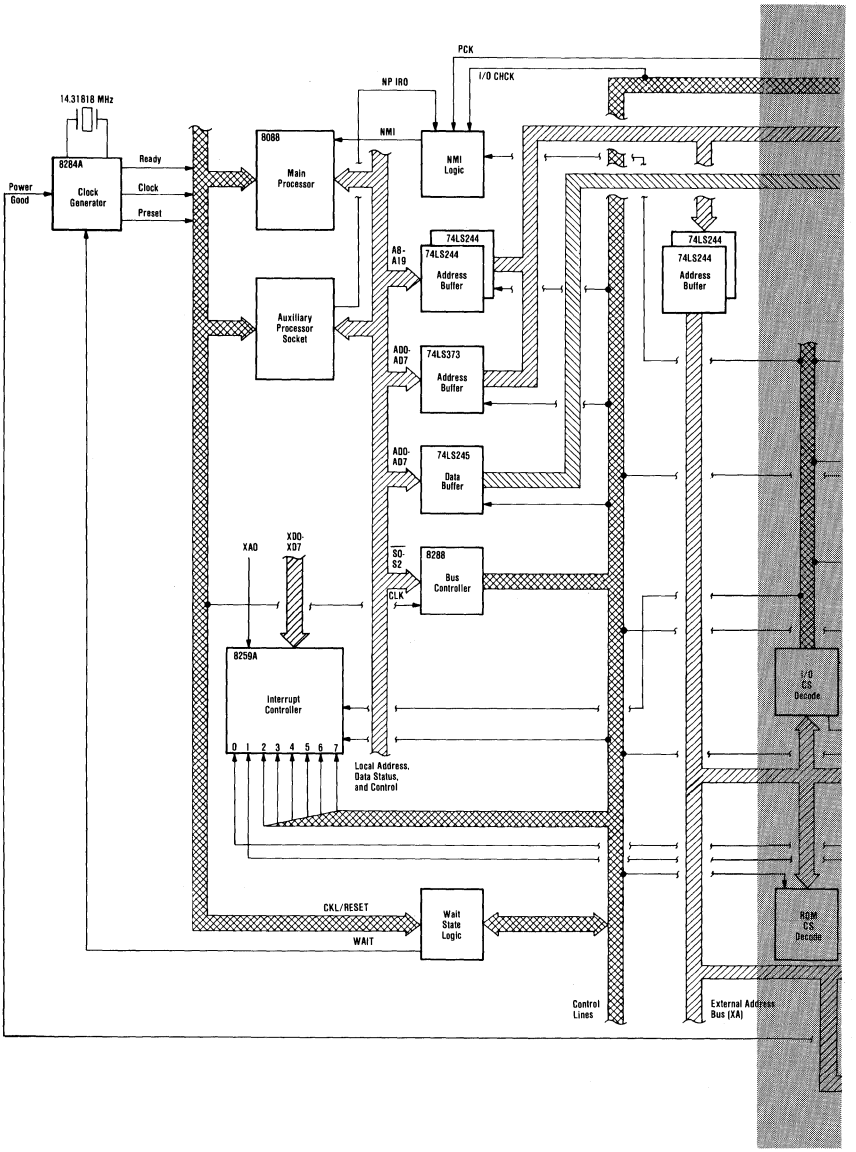
Microprocessor

The heart of the system board is the Intel 8088 Microprocessor. This is an 8-bit external-bus version of Intel's 16-bit 8086 Microprocessor, 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 M byte of storage). It also operates in maximum mode, so a coprocessor can be added as a feature. The microprocessor operates at 4.77-MHz. This frequency is derived from a 14.31818-MHz crystal, the frequency of which is divided by 3 for the microprocessor clock, and divided by 4 to obtain the 3.58-MHz color-burst signal required for color televisions.

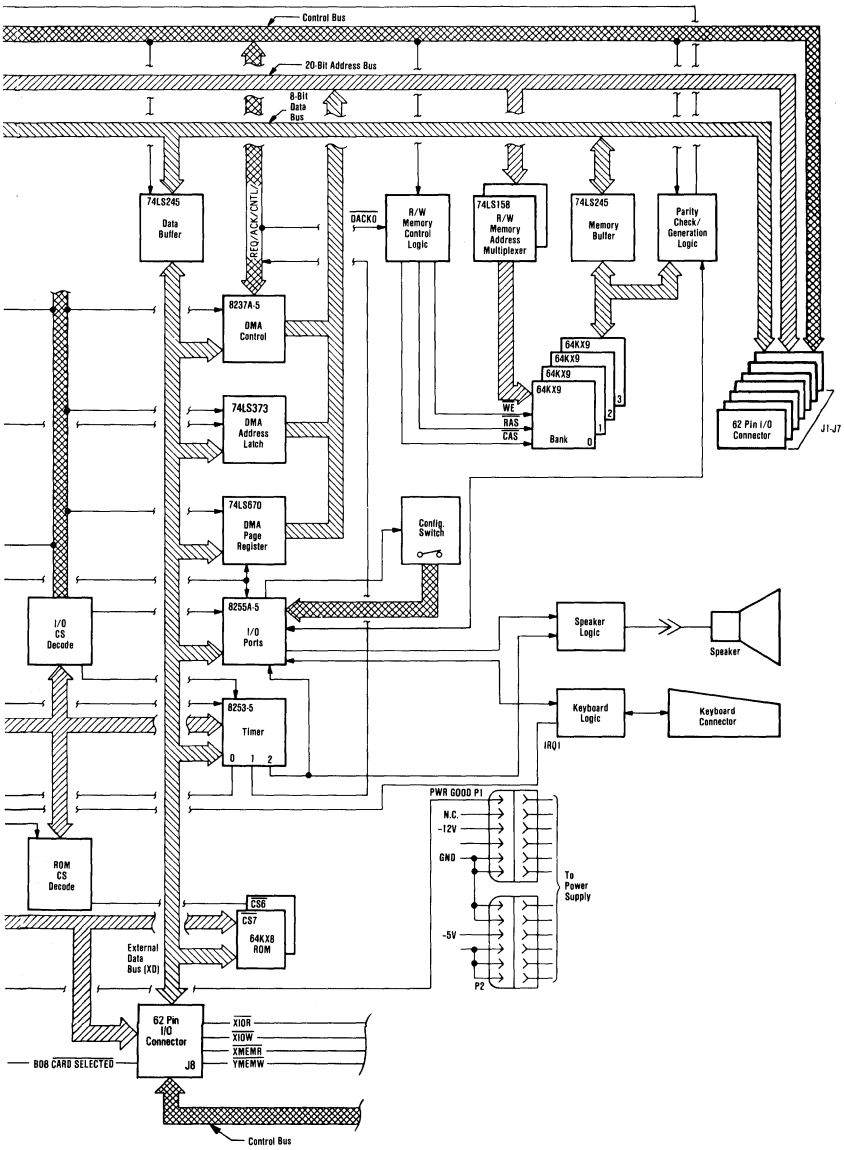
At the 4.77-MHz clock rate, the 8088 bus cycles are four clocks of 210 nanoseconds (ns), or 840-ns. I/O cycles take five 210-ns clocks or 1.05 microseconds (μs).

Data Flow Diagrams

The system board data flow diagram follows.



System Board Data Flow (Part 1 of 2)



System Board Data Flow (Part 2 of 2)

System Memory Map

Start Address Decimal	Start Address Hex	Function
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 Decimal	Start Address Hex	Function
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 Timers

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-μs.

System Interrupts

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 higher 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 following diagram contains the System Interrupt Listing.

Number	Usage
NMI	Parity 8087
0	Timer
1	Keyboard
2	Reserved
3	Asynchronous Communications (Alternate) SDLC Communications BSC Communications Cluster (Primary)
4	Asynchronous Communications (Primary) SDLC Communications BSC Communications
5	Fixed Disk
6	Diskette
7	Printer Cluster (Alternate)

8088 Hardware Interrupt Listing

ROM

The system board supports both read only memory (ROM) and read/write (R/W) memory. It has space for 64K by 8 of ROM or erasable programmable read-only memory (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.

RAM

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.

DMA

The microprocessor 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 microprocessor intervention. The fourth DMA channel is programmed to refresh the system's 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 memory both on the system board and in the system expansion slots. All DMA data transfers, except the refresh channel, take five microprocessor clocks of 210-ns, or 1.05-μs if the microprocessor 'ready' line is not deactivated. Refresh DMA cycles take four clocks or 840-ns.

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 microprocessor-generated memory read and write cycles take four 210-ns clock cycles or 840-ns/byte. All microprocessor-generated I/O read and write cycles require five clocks for a cycle time of 1.05-μs/byte. All DMA transfers require five clocks for a cycle time of 1.05-μs/byte. Refresh cycles occur once every 72 clocks (approximately 15-μs) 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.

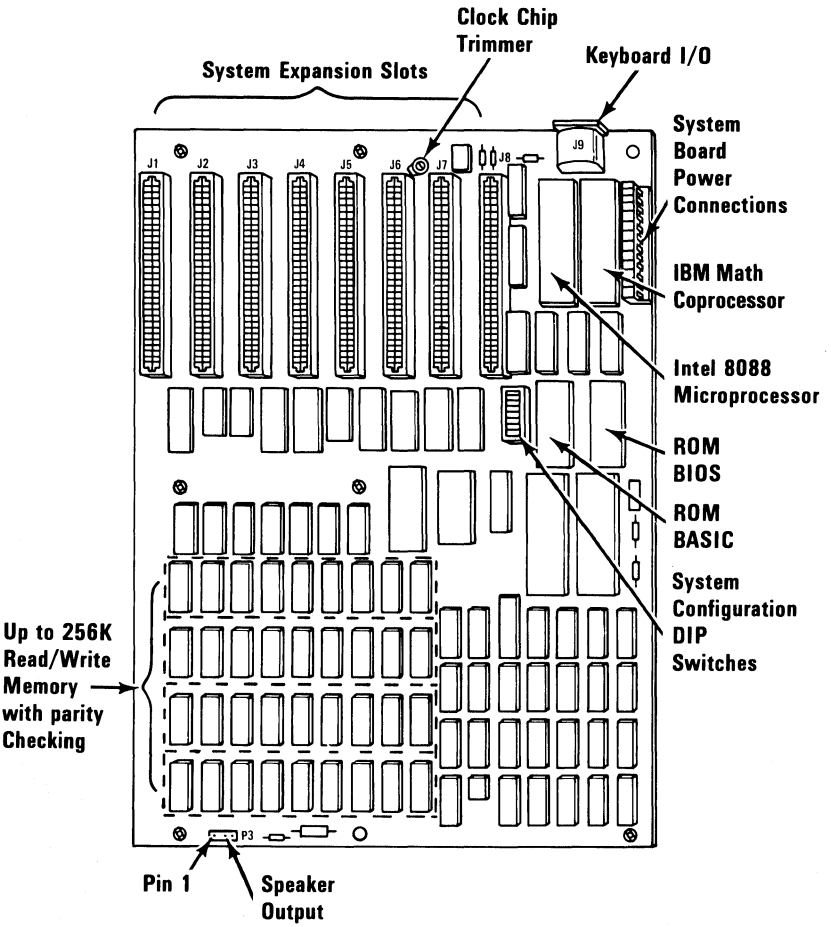
A 'channel check' line exists for reporting error conditions to the microprocessor. Activating this line results in a non-maskable interrupt (NMI) to the 8088 microprocessor. Memory expansion options use this line to report parity errors.

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.

System Board Diagram

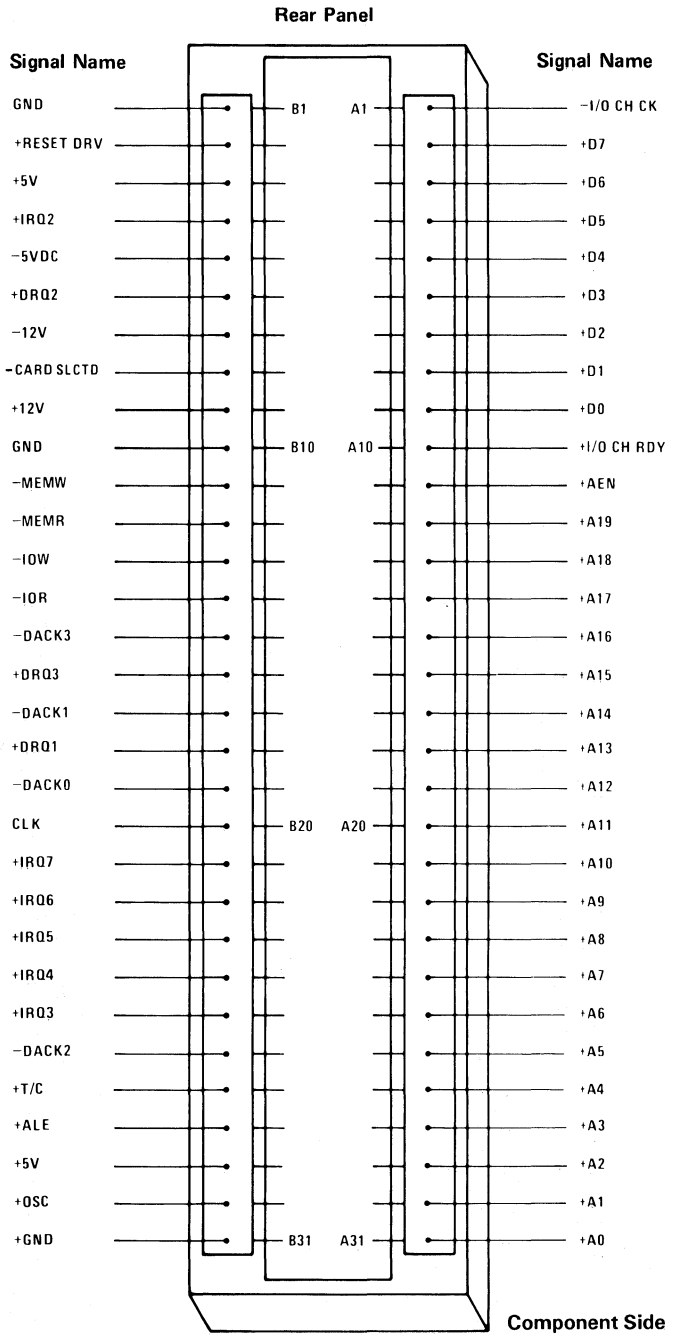
The following diagram shows the component layout for the system board.



System Board Component Diagram

I/O Channel Diagram

The following page contains the I/O channel diagram. All lines are TTL-compatible.



I/O Channel Diagram

I/O Channel Description

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

Signal	I/O	Description
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 1M byte of memory. A0 is the least significant bit (LSB) and A19 is the most significant bit (MSB). These lines are generated by either the microprocessor or DMA controller. They are active high.
AEN	**O**	Address Enable: This line is used to de-gate the microprocessor 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).
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 microprocessor. It is available to the I/O channel as an indicator of a valid microprocessor address (when used with AEN). Microprocessor addresses are latched with the falling edge of ALE.
-CARD SLCTD	**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.
CLK	**O**	System clock: It is a divide-by-3 of the oscillator and has a period of 210-ns (4.77-MHz). The clock has a 33% duty cycle.
D0–D7	**I/O**	Data Bits 0 to 7: These lines provide data bus bits 0 to 7 for the microprocessor, memory, and I/O devices. D0 is the LSB and D7 is the MSB. These lines are active high.
-DACK0 to -DACK3	**O**	-DMA Acknowledge 0 to 3: These lines are used to acknowledge DMA requests (DRQ1–DRQ3) and refresh system dynamic memory (-DACK0). They are 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.
-I/O CH CK	**I**	-I/O Channel Check: This line provides the microprocessor 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 clock cycles (210-ns).
-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 microprocessor or the DMA controller. This signal is active low.
-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 microprocessor or the DMA controller. This signal is active low.
IRQ2–IRQ7	**I**	Interrupt Request 2 to 7: These lines are used to signal the microprocessor 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 microprocessor (interrupt service routine).
-MEMR	**O**	-Memory Read Command: This command line instructs the memory to drive its data onto the data bus. It may

		be driven by the microprocessor or the DMA controller. This signal is active low.
-MEMW	**O**	-Memory Write Command: This command line instructs the memory to store the data present on the data bus. It may be driven by the microprocessor or the DMA controller. This signal is active low.
OSC	**O**	Oscillator: High-speed clock with a 70-ns period (14.31818-MHz). It has a 50% duty cycle.
RESET DRV	**O**	Reset Drive: 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 CLK and is active high.
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.

I/O Address Map

The following page contains the I/O Address Map.

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
200-20F	Game Control
210-217	Expansion Unit
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)
390-393	Cluster
3A0-3A9	Binary Synchronous Communications (Primary)
3B0-3BF	IBM Monochrome Display/Printer
3D0-3DF	Color/Graphics
3F0-3F7	Diskette
3F8-3FF	Asynchronous Communications (Primary)
790-793	Cluster (Adapter 1)
B90-B93	Cluster (Adapter 2)
1390-1393	Cluster (Adapter 3)
2390-2393	Cluster (Adapter 4)

* These are the addresses decoded by the current set of adapter cards. IBM may use any of the unlisted addresses for future use.

** 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.

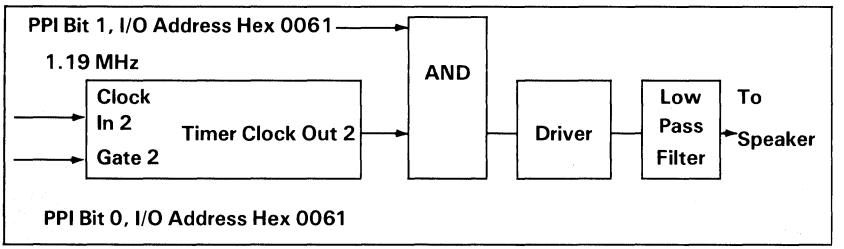
I/O Address Map

Other Circuits

Speaker Circuit

The system unit has a 57.15 mm (2-1/4 in.) 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.



Speaker Drive System Block Diagram

Channel 2 (Tone generation for speaker)	
Gate 2	— Controlled by 8255A-5 PPI Bit (See I/O Map)
Clock In 2	— 1.19318 - MHz OSC
Clock Out 2	— Used to drive speaker

Speaker Tone Generation

The speaker connection is a 4-pin Berg connector.

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

Speaker Connector

8255A I/O Bit Map

The 8255A I/O Bit Map shows the inputs and outputs for the Command/Mode register on the system board. Also shown are the switch settings for the memory, display, and number of diskette drives. The following page contains the I/O bit map.

Hex Port Number				
0060	INPUT	PA0	+ Keyboard Scan Code 0	Or Diagnostic Outputs 0
		PA1	+ Keyboard Scan Code 1	Diagnostic Outputs 1
		PA2	+ Keyboard Scan Code 2	Diagnostic Outputs 2
		PA3	+ Keyboard Scan Code 3	Diagnostic Outputs 3
		PA4	+ Keyboard Scan Code 4	Diagnostic Outputs 4
		PA5	+ Keyboard Scan Code 5	Diagnostic Outputs 5
		PA6	+ Keyboard Scan Code 6	Diagnostic Outputs 6
		PA7	+ Keyboard Scan Code 7	Diagnostic Outputs 7
0061	OUTPUT	PB0	+ Timer 2 Gate Speaker	
		PB1	+ Speaker Data	
		PB2	Spare	
		PB3	Read High Switches Or Read Low Switches	
		PB4	– Enable RAM Parity Check	
		PB5	– Enable I/O Channel Check	
		PB6	– Hold Keyboard Clock Low	
		PB7	– (Enable Keyboard Or + (Clear Keyboard)	
0062	INPUT	PC0	Loop on POST	Sw–1 Or Display 0 **Sw–5
		PC1	+ Co-Processor Installed	Sw–2 Display 1 **Sw–6
		PC2	+ Planar RAM Size 0	*Sw–3 #5-1/4 Drives ***Sw–7
		PC3	+ Planar RAM Size 1	*Sw–4 #5-1/4 Drives 1 ***Sw–8
		PC4	Spare	
		PC5	+ Timer Channel 2 Out	
		PC6	+ I/O Channel Check	
		PC7	+ RAM Parity Check	

0063 Command/Mode Register

Mode Register Value

	Hex 99							
	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

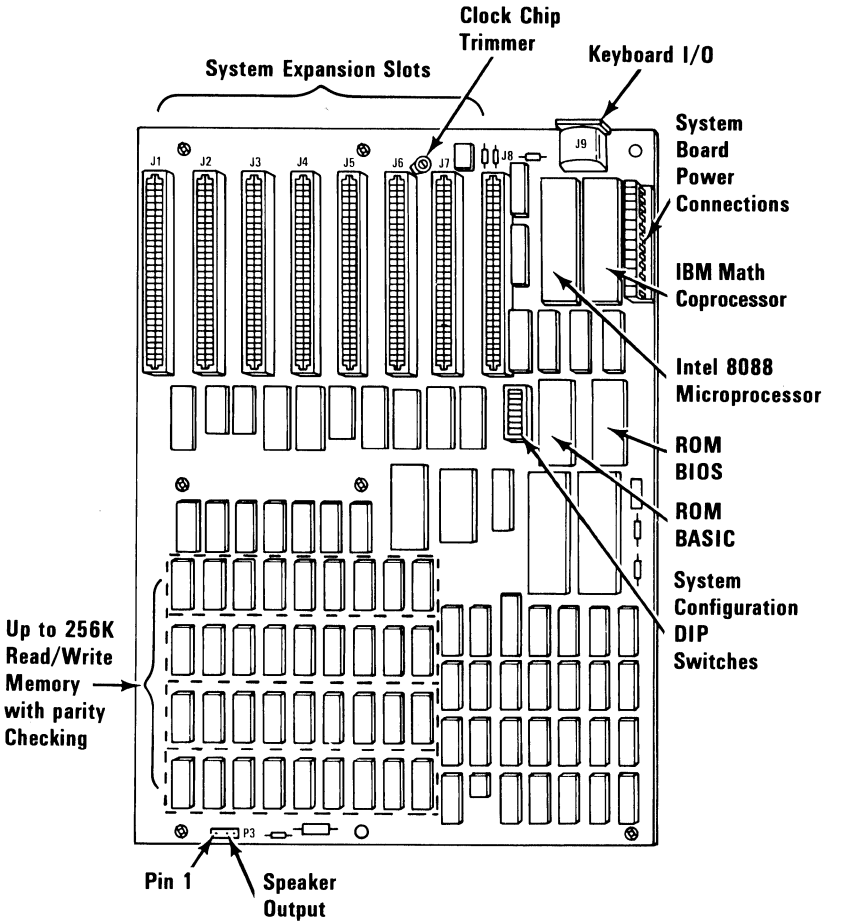
***			
	Sw–8	Sw–7	Number of 5-1/4" Drives In System
	0	0	1
	0	1	2
	1	0	3
	1	1	4

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

System-Board Switch Settings

All system board switch settings for total system memory, number of diskette drives, and type of display adapter are described under "Switch Settings" in the IBM Personal Computer XT *Guide to Operations*. The diagram showing the system board switch location follows.



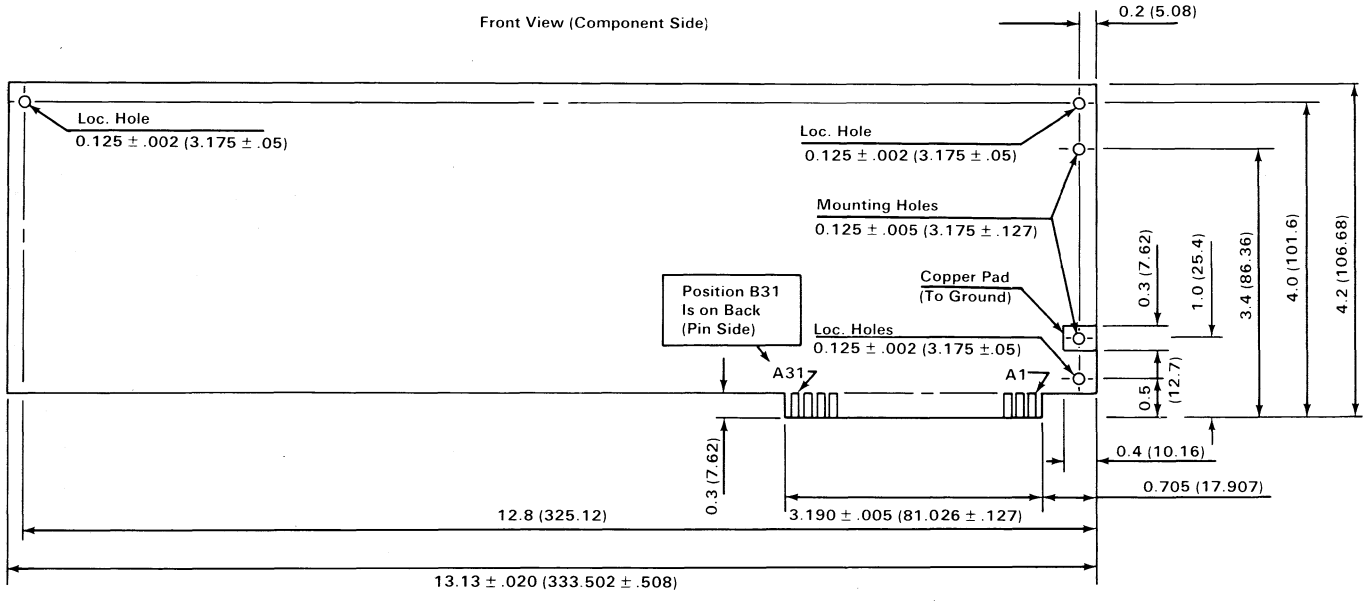
Specifications

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

+ 5 Vdc ± 5% on 2 connector pins
− 5 Vdc ±10% on 1 connector pin
+12 Vdc ± 5% on 1 connector pin
−12 Vdc ±10% on 1 connector pin
GND (Ground) on 3 connector pins

Card Specifications

The specifications for option cards follow.

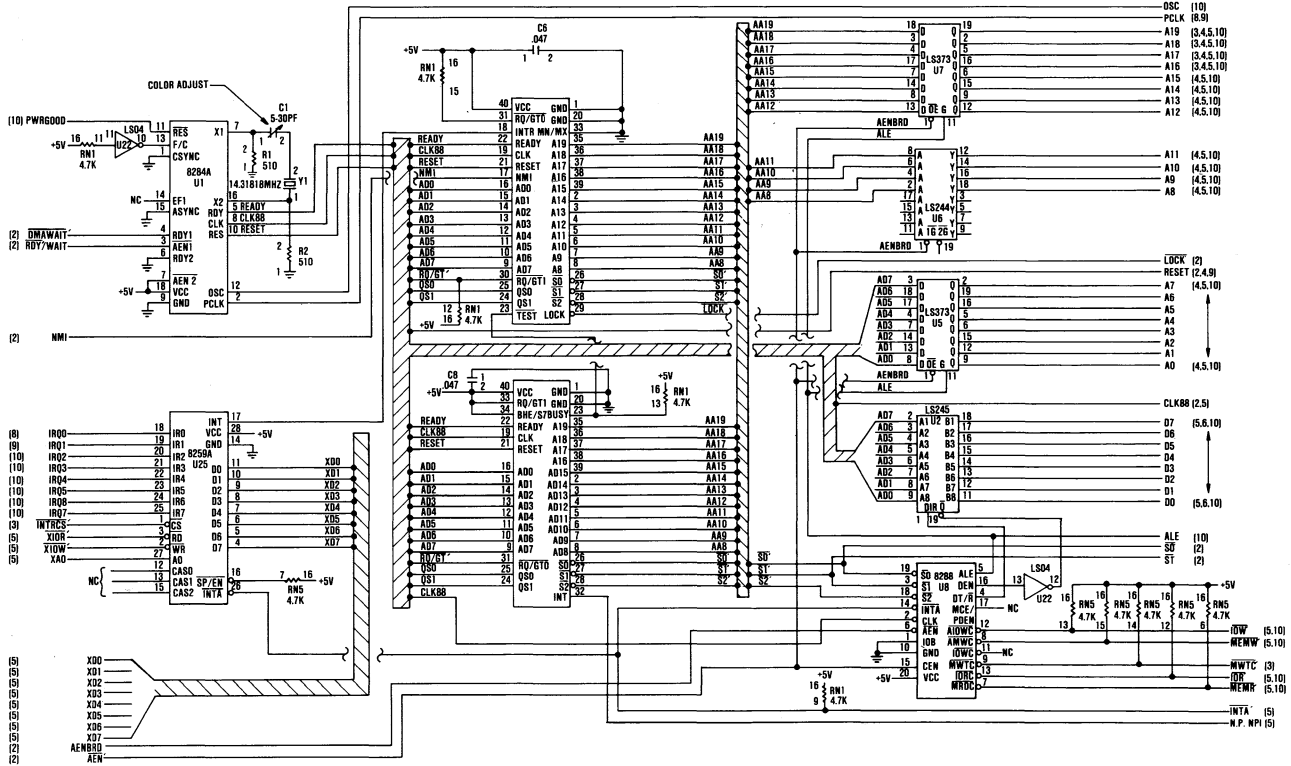


Notes:

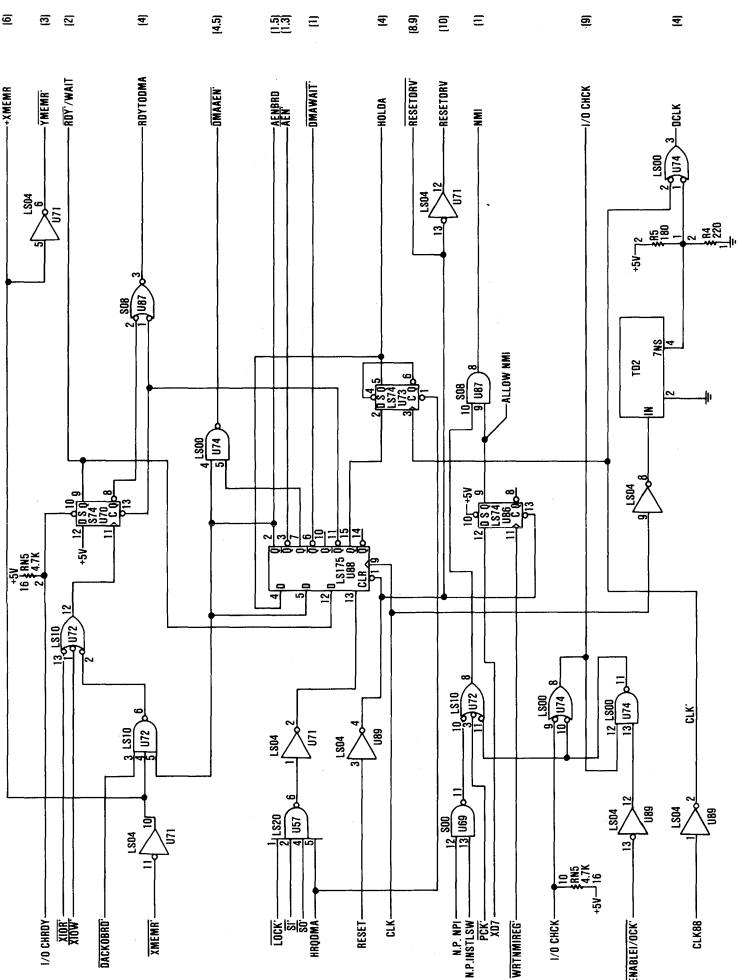
1. All Card Dimensions are ± .010 (.254) Tolerance (With Exceptions Indicated on Drawing or in Notes).
2. Max. Card Length is 13.15 (334.01) Smaller Length is Permissible.
3. Loc. and Mounting Holes are Non-Plated Thru. (Loc. 3X, Mtg. 2X).
4. 31 Gold Tabs Each Side, 0.100 ± .0005 (2.54 ± .0127) Center to Center, 0.06 ± .0005 (1.524 ± .0127) Width.
5. Numbers in Parentheses are in Millimeters. All Others are in Inches.

Logic Diagrams

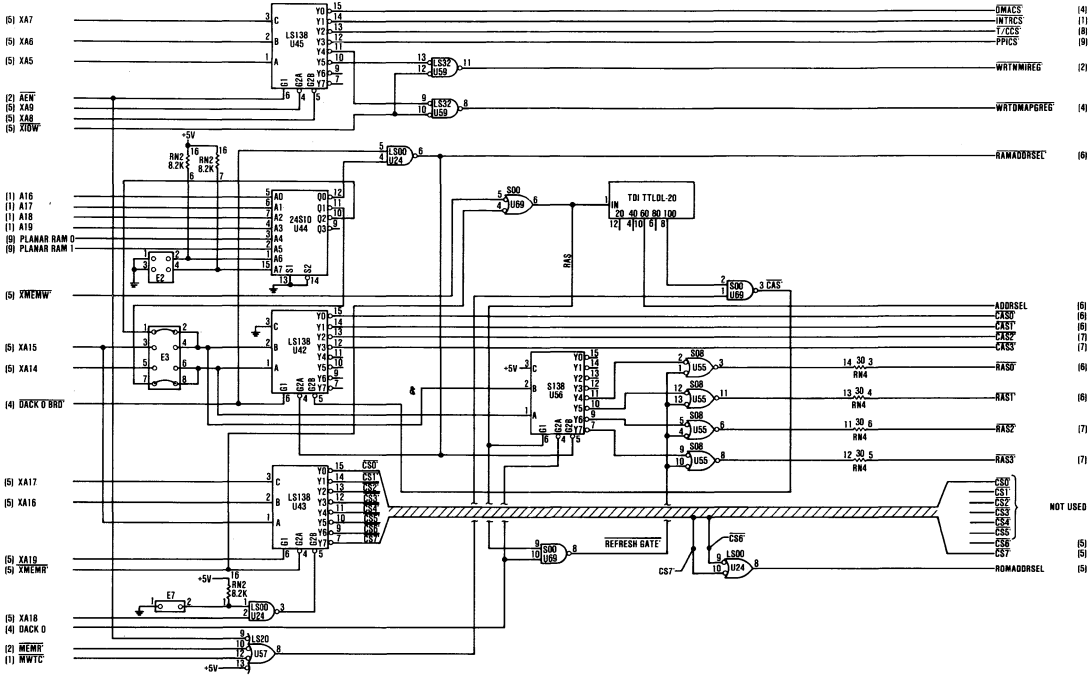
The following pages contain the logic diagrams for the system board.



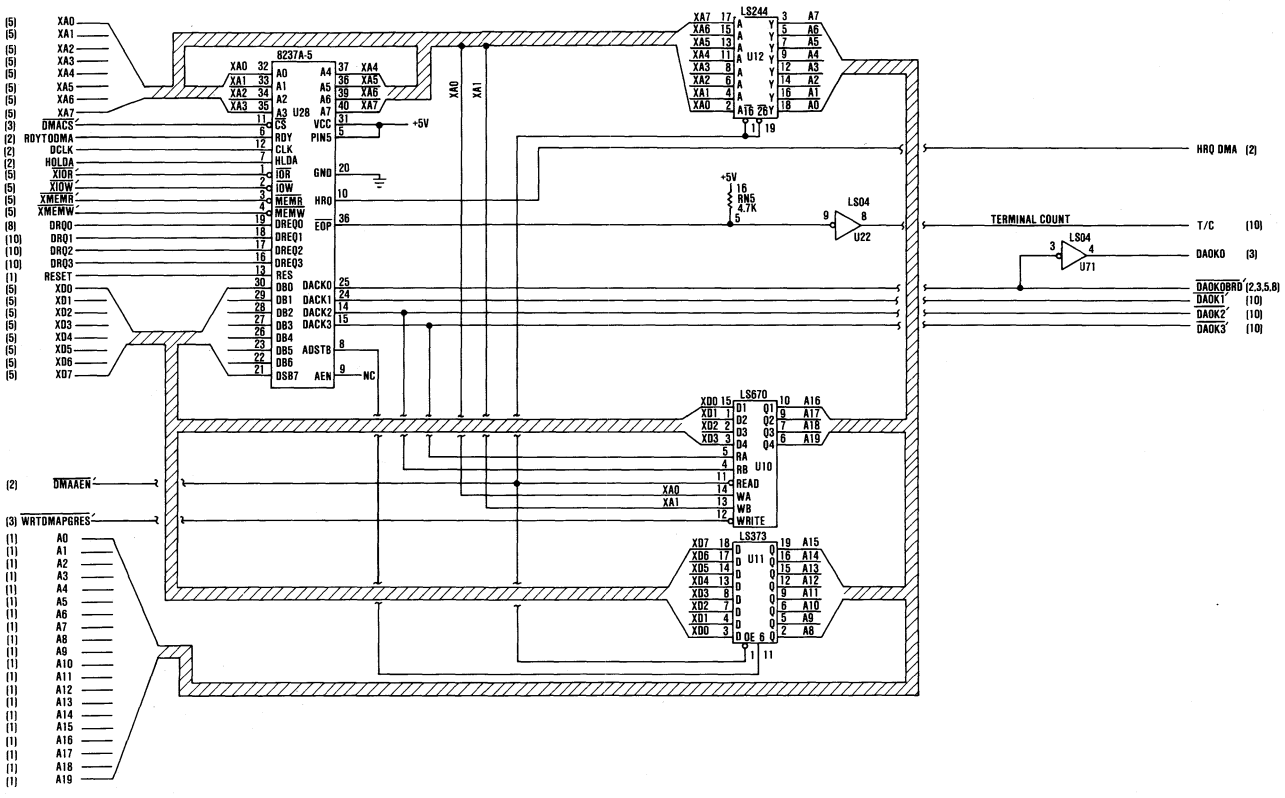
System Board (Sheet 1 of 10)



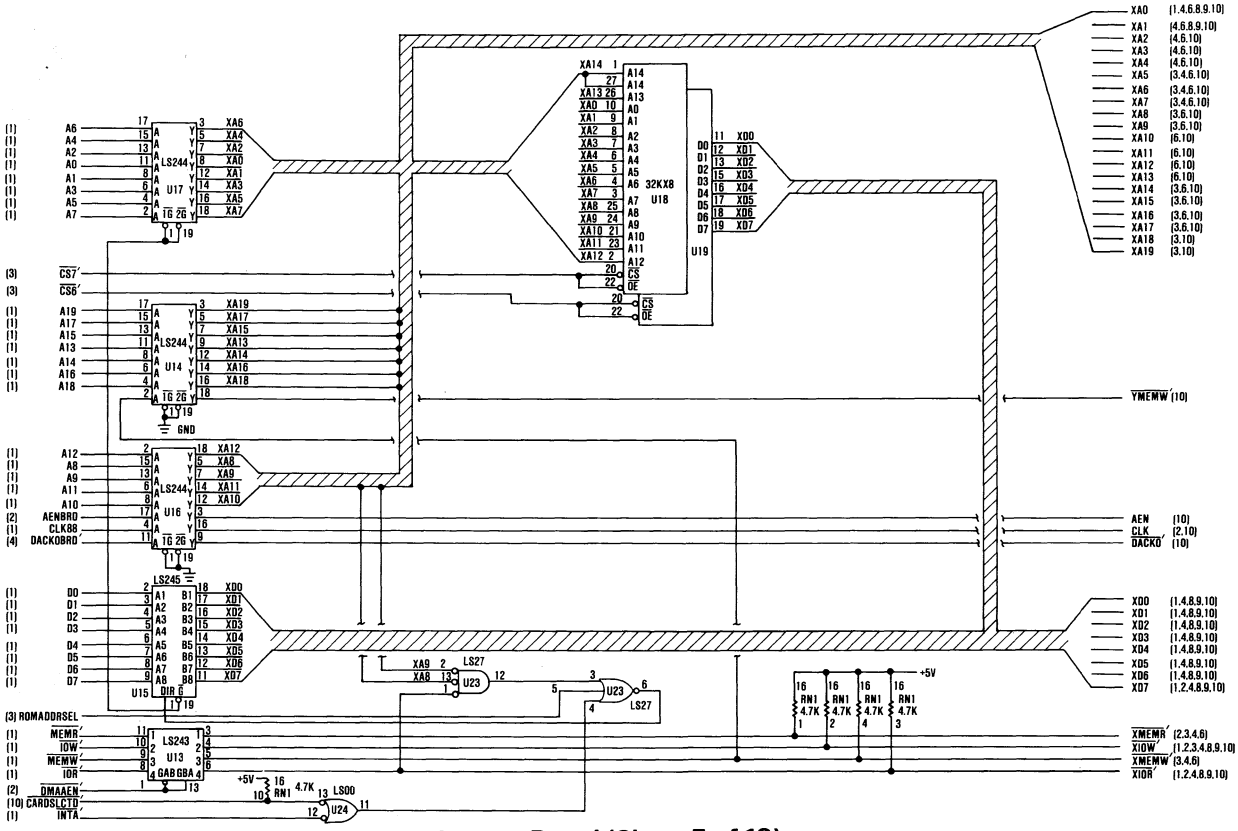
System Board (Sheet 2 of 10)



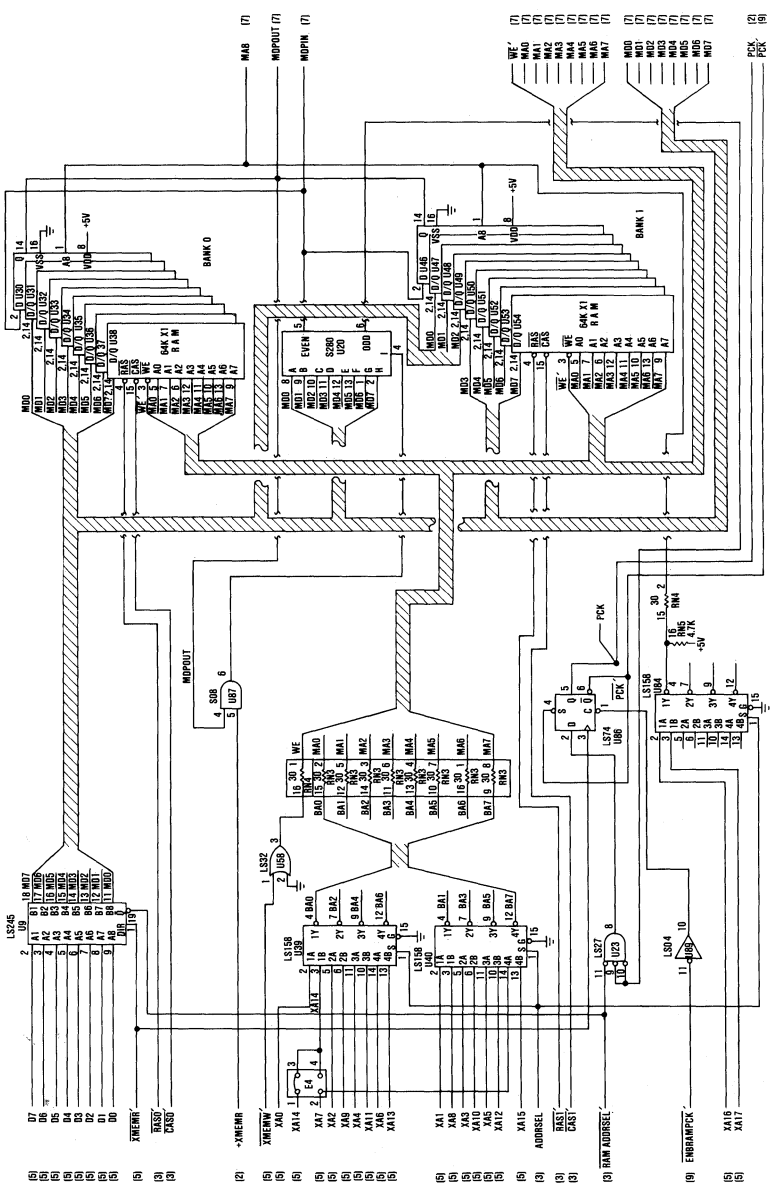
System Board (Sheet 3 of 10)



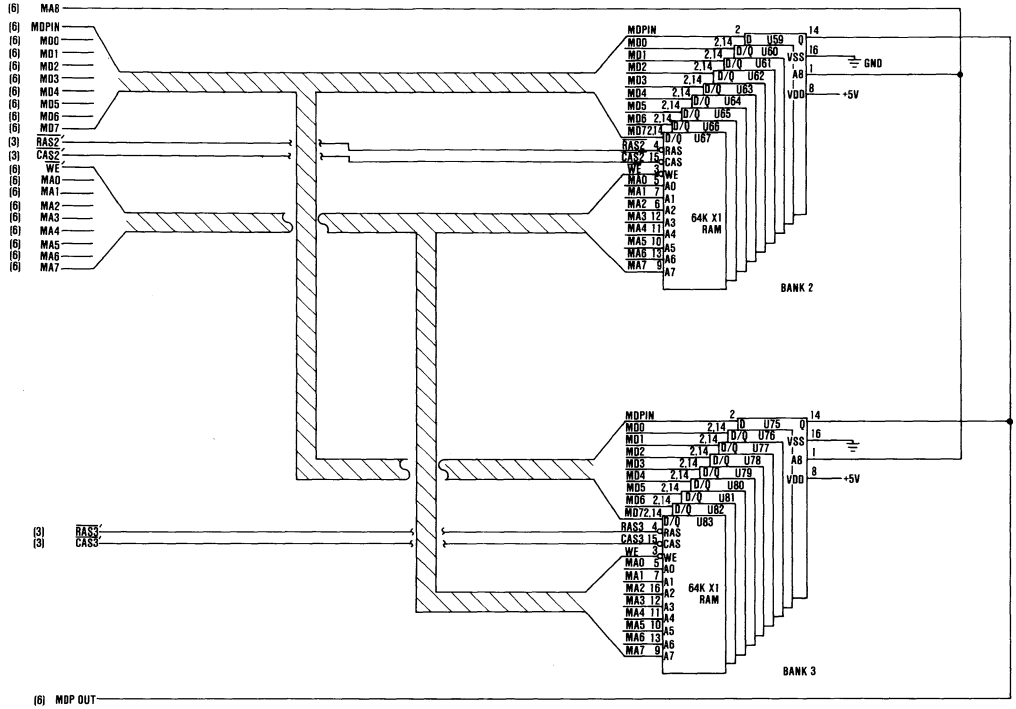
System Board Sheet (4 of 10)



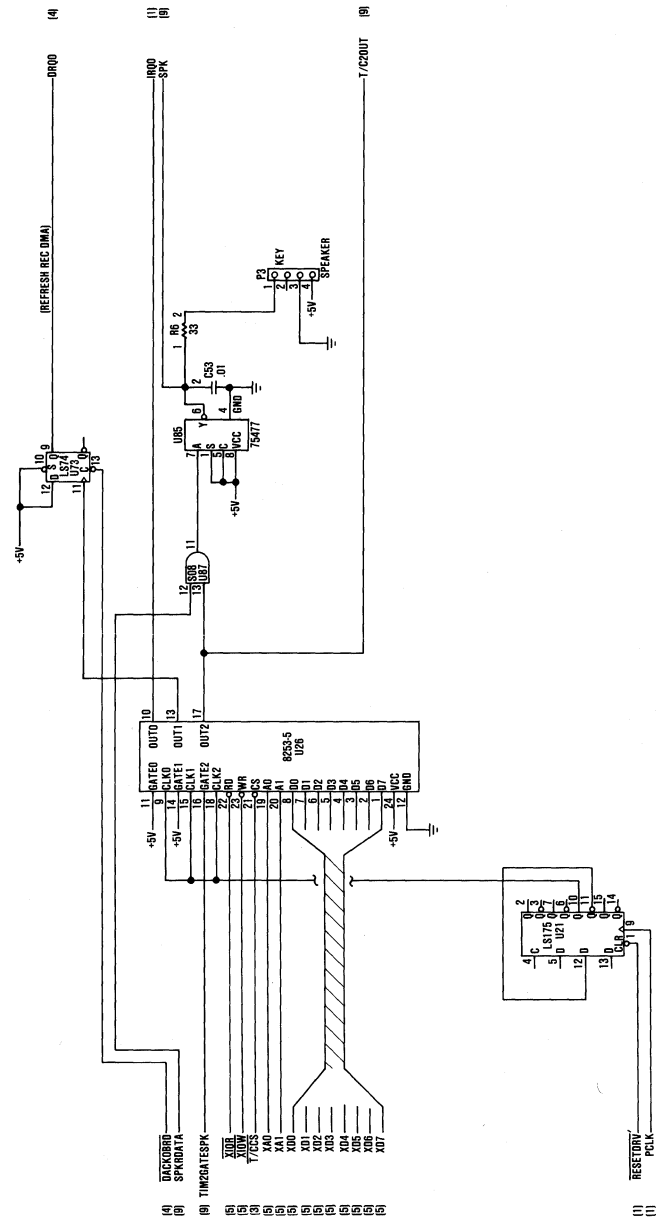
System Board (Sheet 5 of 10)



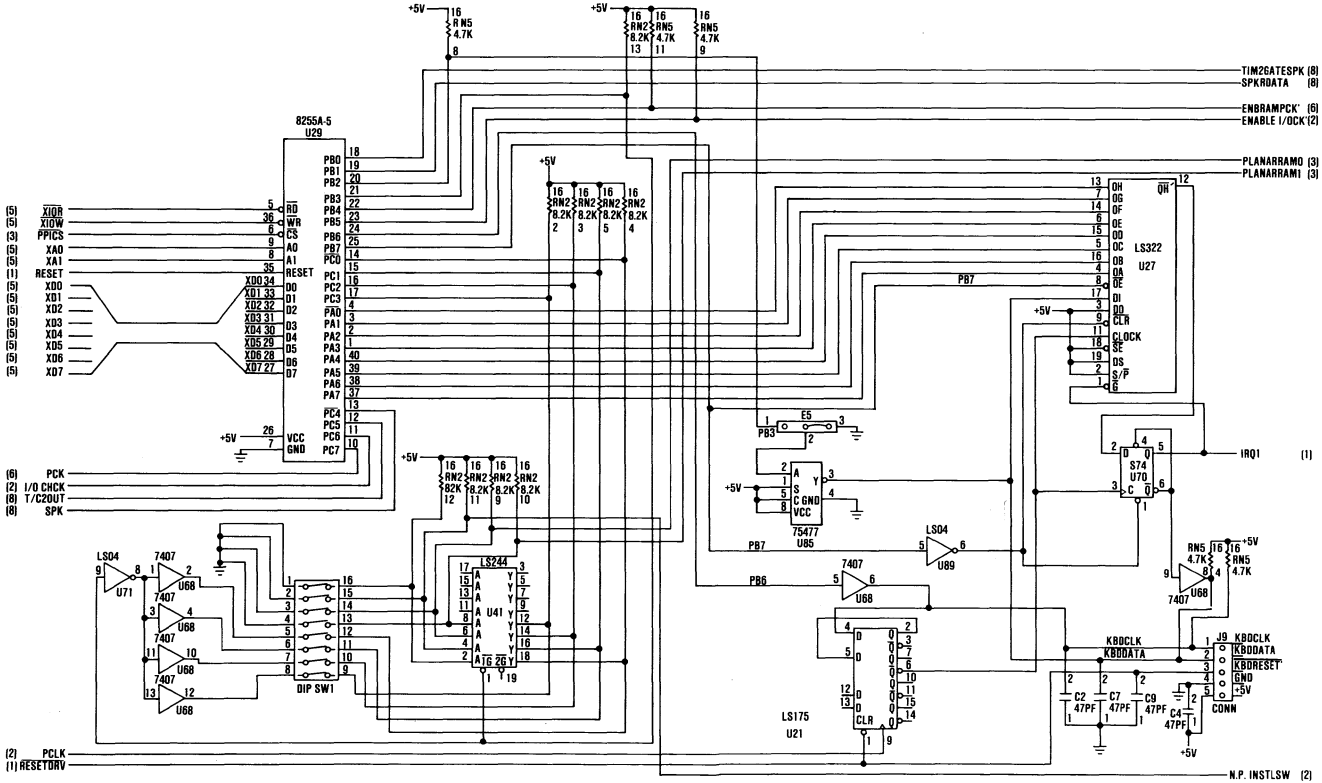
System Board (Sheet 6 of 10)



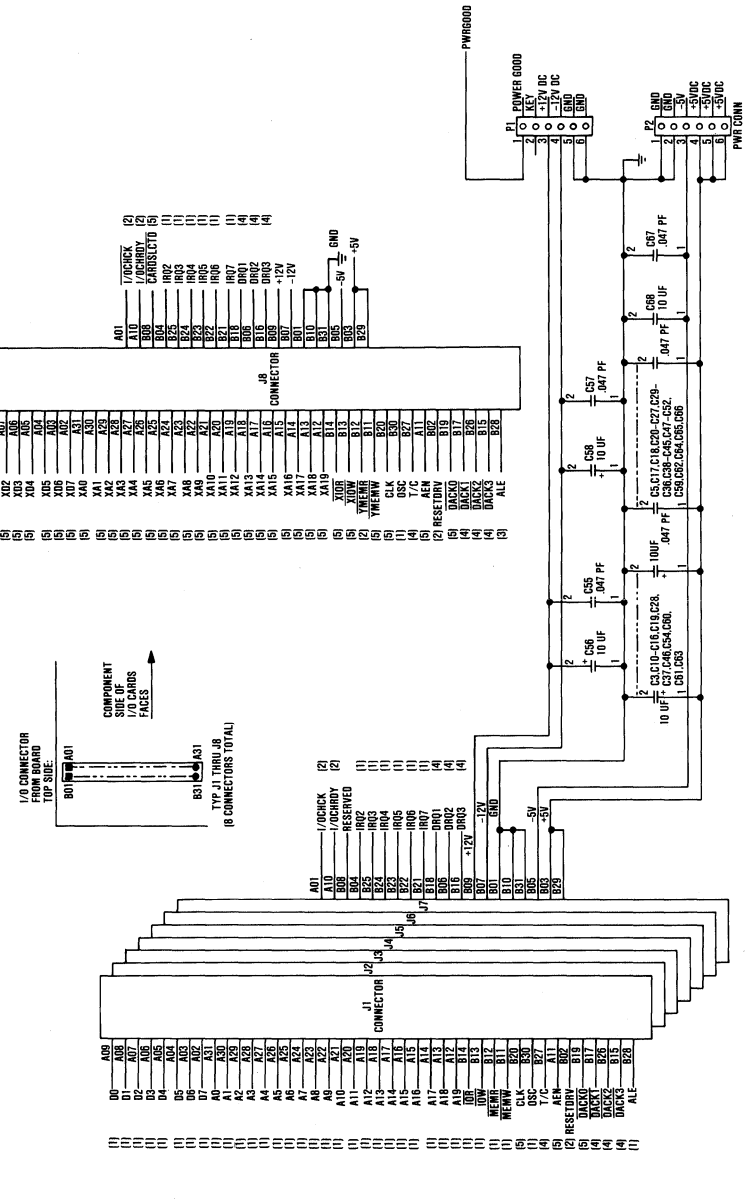
System Board (Sheet 7 of 10)



System Board (Sheet 8 of 10)



System Board (Sheet 9 of 10)



System Board (Sheet 10 of 10)

SECTION 2. COPROCESSOR

Contents

Description

The Math Coprocessor (8087) enables the IBM Personal Computer to perform high-speed arithmetic, logarithmic functions, and trigonometric operations with extreme accuracy.

The 8087 coprocessor works in parallel with the microprocessor. The parallel operation decreases operating time by allowing the coprocessor to do mathematical calculations while the microprocessor continues to do other functions.

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

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 microprocessor.

The coprocessor has eight 80-bit registers, which provide the equivalent capacity of the 40 16-bit registers found in the microprocessor. 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. 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	$-2\times10^{9} \leq X \leq +2\times10^{9}$
Long Integer	64	18	$-9\times10^{18} \leq X \leq +9\times10^{18}$
Packed Decimal	80	18	$-99\ldots99 \leq X \leq +99\ldots99$ (18 digits)
Short Real*	32	6-7	$8.43\times10^{-37} \leq \lvert X\rvert \leq 3.37\times10^{38}$
Long Real*	64	15-16	$4.19\times10^{-307} \leq \lvert X\rvert \leq 1.67\times10^{308}$
Temporary Real	80	19	$3.4\times10^{-4932} \leq \lvert X\rvert \leq 1.2\times10^{4932}$

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

Data Types

Hardware Interface

The coprocessor uses the same clock generator and system bus interface components as the microprocessor. The coprocessor is wired directly into the microprocessor. The microprocessor's queue status lines (QS0 and QS1) enable the coprocessor to obtain and decode instructions simultaneously with the microprocessor. The coprocessor's 'busy' signal informs the microprocessor that it is executing; the microprocessor's WAIT instruction forces the microprocessor 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 0 or load a full register), the coprocessor can

signal the microprocessor with an interrupt. There are three conditions that will disable the coprocessor interrupt to the microprocessor:

1. Exception and interrupt-enable bits of the control word are set to 1's.

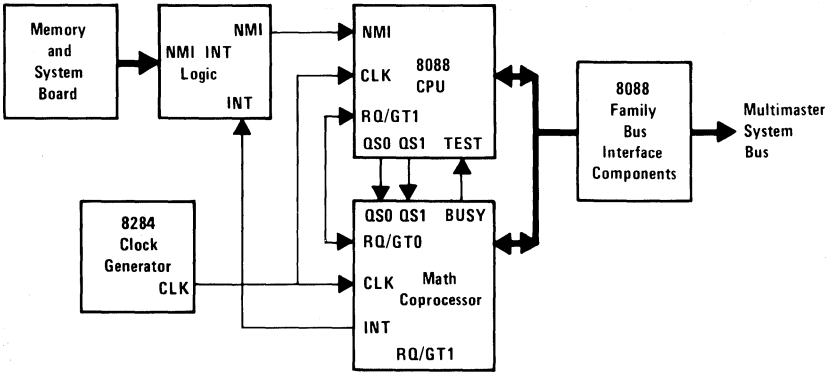
2. System-board switch-block 1, switch 2, set in the On position.

3. Non-maskable interrupt (NMI) register (REG) is set to zero.

At power-on time, the NMI REG is cleared to disable the NMI. Any program 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 microprocessor waiting for the 'not busy' signal from the coprocessor while the coprocessor is waiting for the microprocessor to interrupt.

Because a memory parity error may also cause an interrupt to the microprocessor NMI line, the program should check the coprocessor status for an exception condition. If a coprocessor exception condition is not found, control should be passed to the normal NMI handler. If an 8087 exception condition is found, the program may clear the exception by executing the FNSAVE or the FNCLEX instruction, and the exception can be identified and acted upon.

The NMI REG and the coprocessor's interrupt are tied to the NMI line through the NMI interrupt logic. Minor modifications to programs designed for use with a coprocessor must be made before the programs will be compatible with the IBM Personal Computer Math Coprocessor.



Coprocessor Interconnection

Detailed information for the internal functions of the Intel 8087 Coprocessor can be found in the books listed in the Bibliography.

SECTION 3. POWER SUPPLY

Contents

IBM Personal Computer XT Power Supply

Description

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 overvoltage and overcurrent protection. There are two power supplies, 120 Vac and 220/240 Vac. Both are fused. If dc overcurrent or overvoltage 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 10M fixed disk drive. The -5 Vdc level is used for analog circuits in the diskette adapter's phase-lock loop. The +12 Vdc and -12 Vdc are used for powering the Electronic Industries Association (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's 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.

Input Requirements

The nominal power requirements and output voltages are listed in the following tables.

Voltage @ 50/60 Hz		
Nominal Vac	Minimum Vac	Maximum Vac
110 220/240	90 180	137 259

Input Requirements

Frequency: 50/60 Hz ± 3 Hz

Current: 4.1 A max at 90 Vac

Outputs

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

Vdc Output

Voltage (Vac)	Current (Amps)		Voltage Limits (Vac)	
Nominal	Minimum	Maximum	Minimum	Maximum
120	0.0	1.0	88	137
220/240	0.0	1.0	180	259

Vac Output

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

Overvoltage/Overcurrent Protection

Nominal Voltage (Vac)	Type Protection	Rating (A)
100-125	Fused	5 A
200-240	Fused	2.5 A

Voltage and Current Protection

Power Good

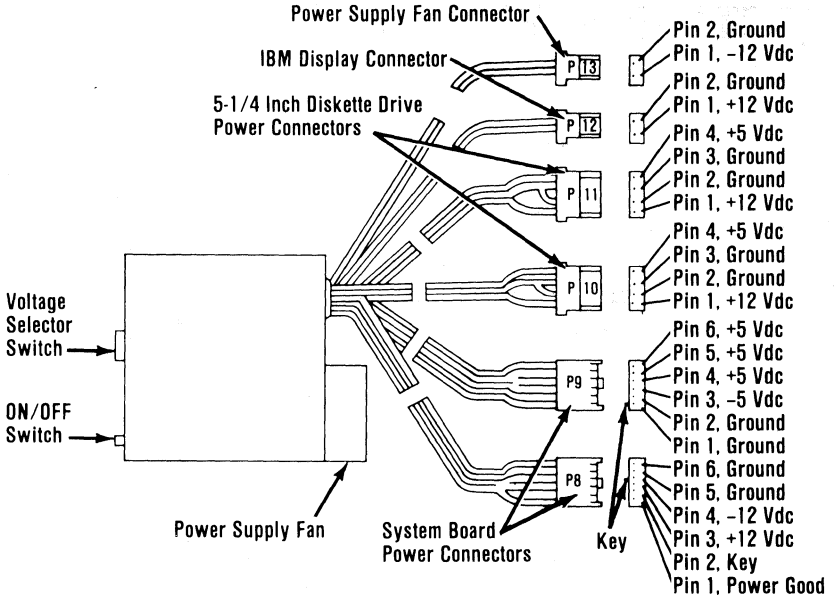
When the power supply is switched off for a minimum of 1 second and then switched on, the 'power good' signal is regenerated.

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 sense voltage limits.

When power is switched on, the dc output-voltage sense signal holds the 'power good' signal at a down level until all output voltages reach their minimum sense levels. The 'power good' signal has a turn-on delay of 100 to 500 milliseconds.

Connector Specifications and Pin Assignments

The power connector on the system board is a 12-pin connector that plugs into the power supply connectors, P8 and P9. The Input Voltage Selector switch and the pin assignment locations follow.



Power Supply and Connectors

Overvoltage/Overcurrent Protection

Voltage Nominal Vac	Type Protection	Rating Amps
110	Fuse	5.0
220/240	Fuse	3.5

Power-Good

When the supply is switched off for a minimum of 1.0 second, and then switched 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.

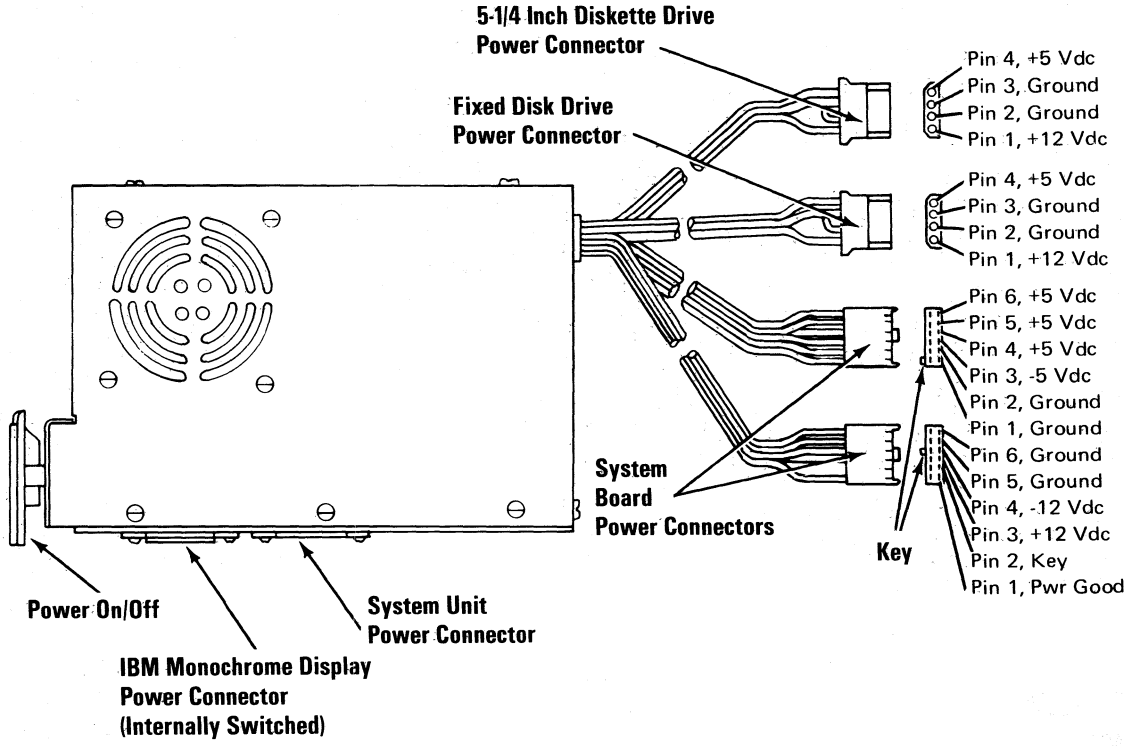
This signal is the logical AND of the dc output-voltage 'sense' signal and the ac input-voltage 'fail' signal. This signal is TTL-compatible up-level for normal operation or down-level for fault conditions. The ac 'fail' signal causes 'power good' to go to a down level when any output voltage falls below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

Connector Specifications 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 assignments and locations are shown on the following page.

Power Supply and Connectors



IBM Portable Personal Computer Power Supply

Description

The system unit's power supply is a 114-watt, switching regulator that provides five outputs. It supplies power for the system unit and its options, the power supply fan, the diskette drive, the composite display, and the keyboard. All power levels are protected against overvoltage and overcurrent conditions. The input voltage selector switch has 115 Vac and 230 Vac positions. If a dc overload or overvoltage condition exists, the power supply automatically shuts down until the condition is corrected, and the power supply is switched off and then on.

The internal 5-1/4 inch diskette drive uses the +5 Vdc and the +12 Vdc power levels. Both the +12 Vdc and -12 Vdc power levels are used in the drivers and receivers of the optional communications adapters. The display uses a separate +12 Vdc power level. The +5 Vdc, -5 Vdc, +12 Vdc, and -12 Vdc power levels are bussed across the system expansion slots.

Input Requirements

Nominal Voltage (Vac)	Minimum Voltage (Vac)	Maximum Voltage (Vac)
100-125	90	137
200-240	180	259

Note: Input voltage to be 50 or 60 hertz, ± 3 hertz.

Amperage Output

Nominal Voltage (Vdc)	Minimum Current (A)	Maximum Current (A)
+ 5	2.3	11.2
- 5	0.0	0.3
- 12	0.0	0.25
+ 12	0.04	2.9
+ 12 (display)	0.5	1.5

Amperage Output

Note: Maximum current is 3.5 amperes at 90 Vac.

Vdc Sense Voltage Output

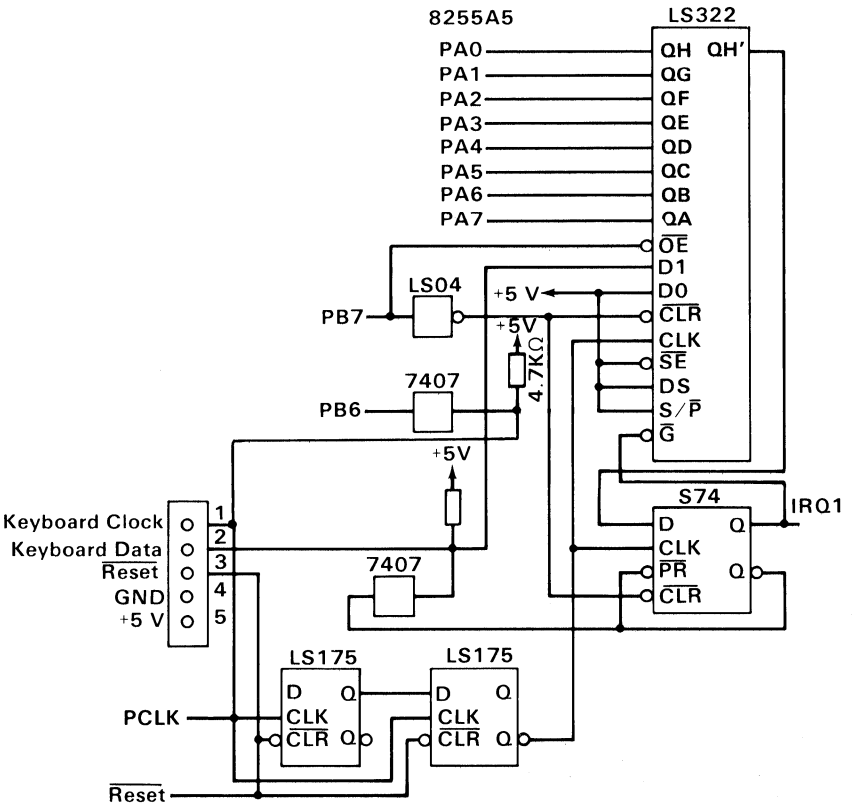
Nominal Voltage (Vdc)	Minimum Sense Voltage (Vdc)	Maximum Sense Voltage (Vdc)
+ 5	+ 4.5	+ 6.5
- 5	- 4.3	- 6.5
- 12	- 10.2	- 15.6
+ 12	+ 10.8	+ 15.6
+ 12 (display)	+ 10.8	+ 15.6

Vdc Sense Voltage Tolerance

SECTION 4. KEYBOARD

Contents

Block Diagram



Keyboard Interface Block Diagram

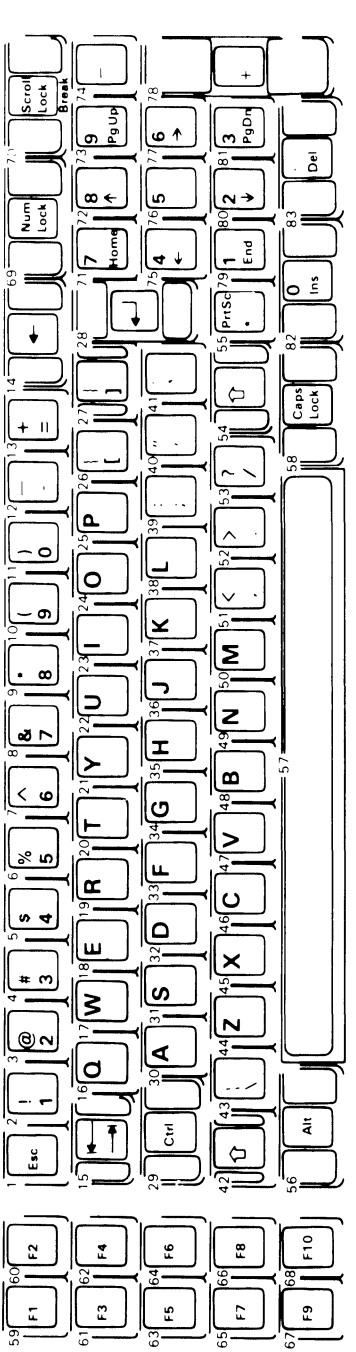
Keyboard Diagrams

The IBM Personal Computer keyboard is available in six different layouts:

- U.S. English
- U.K. English
- French
- German
- Italian
- Spanish

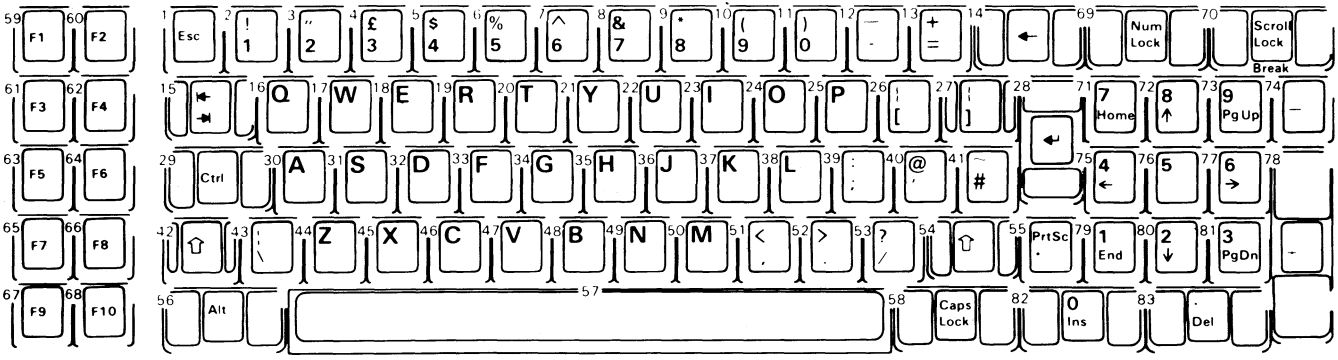
The following pages show all six keyboard layouts.

U.S. English Keyboard Diagram



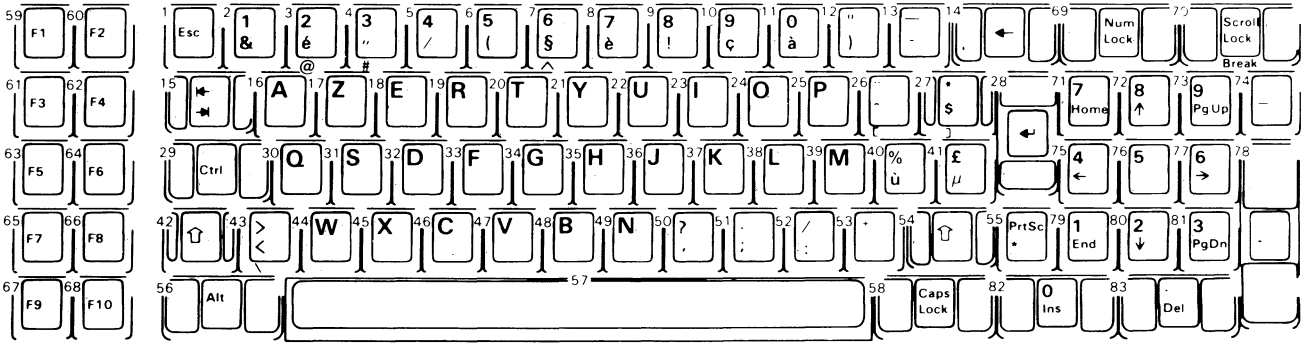
Note: Nomenclature is on both the top and front face of keybuttons as shown. The number to the upper left designates the button position.

U.K. English Keyboard Diagram



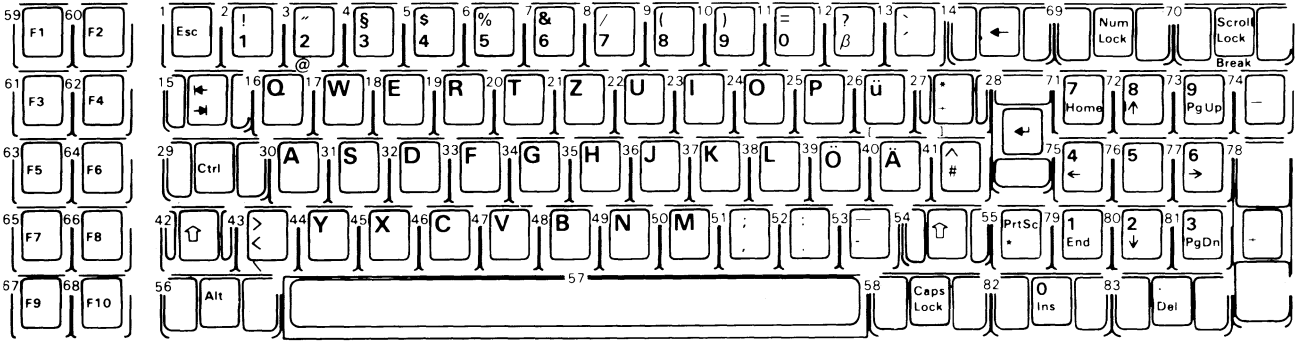
Note: Nomenclature is on both the top and front face of keybuttons as shown. The number to the upper left designates the button position.

French Keyboard Diagram



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

German Keyboard Diagram



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

IBM Personal Computer XT Keyboard and IBM Portable Personal Computer Keyboard

Description

The Personal Computer XT keyboard has a permanently attached cable that connects to a DIN connector at the rear of the system unit. This shielded 5-wire cable has power (+5 Vdc), ground, and two bidirectional signal lines. The cable is approximately 182.88 cm (6 ft) long and is coiled, like that of a telephone handset.

The IBM Portable Personal Computer keyboard cable is detachable, 4-wire, shielded cable that connects to a modular connector in the front panel of the system unit. The cable has power, (+5 Vdc), ground, and two bidirectional signal lines in it. It is 762 mm (30 in.) long and is coiled.

Both keyboards use a capacitive technology with a microprocessor (Intel 8048) performing the keyboard scan function. The keyboard has two tilt positions for operator comfort (5- or 15-degree tilt orientations for the Personal Computer XT and 5- or 12-degree tilt orientations for the IBM Portable Personal Computer).

> **Note:** The following descriptions are common to both the Personal Computer XT and IBM Portable Personal Computer.

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 (if held down, they will repeat) 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.

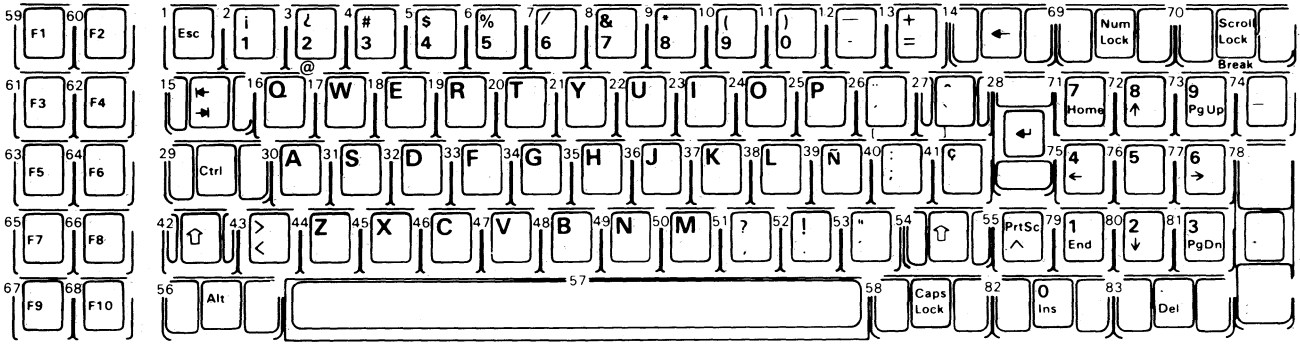
The microprocessor (Intel 8048) in the keyboard performs several functions, including a power-on self test when requested by the system unit. This test checks the microprocessor's ROM, tests memory, and checks for stuck keys. Additional functions are keyboard scanning, buffering of up to 16 key scan codes, maintaining bidirectional serial communications with the system unit, and executing the handshake protocol required by each scan-code transfer.

Several different keyboard arrangements are available. These are illustrated on the following pages. For information about the keyboard routines required to implement non-U.S. keyboards, refer to the *Guide to Operations* and *DOS* manuals.

Italian Keyboard Diagram

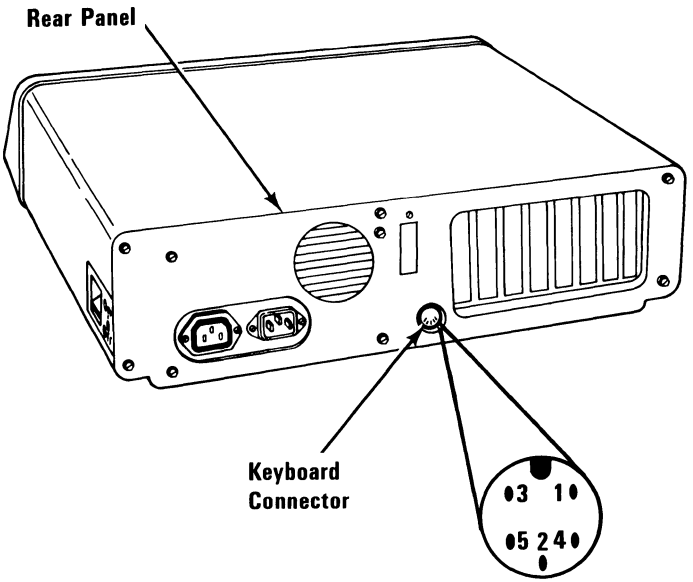
Note: Nomenclature is on both the top and front face of keybuttons as shown. The number to the upper left designates the button position.

Spanish Keyboard Diagram



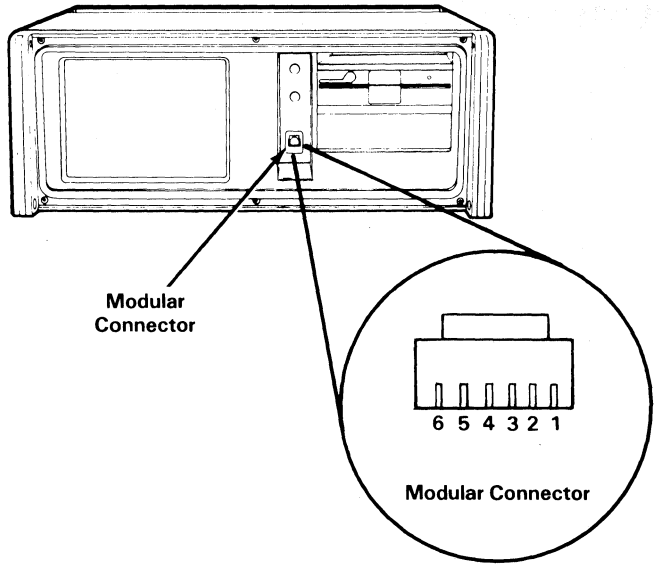
Note: Nomenclature is on both the top and front face of keybuttons as shown. The number to the upper left designates the button position.

Connector Specifications

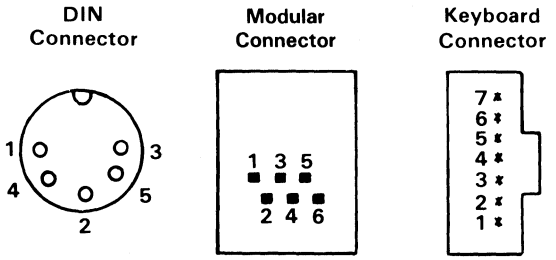


Pin	TTL Signal	Signal Level
1	+ Keyboard Clock	+ 5 Vdc
2	+ Keyboard Data	+ 5 Vdc
3	– Keyboard Reset (Not used by keyboard)	
	Power Supply Voltages	**Voltage**
4	Ground	0
5	+ 5 Volts	+ 5 Vdc

Keyboard Interface Connector Specifications



Keyboard Cable Connections

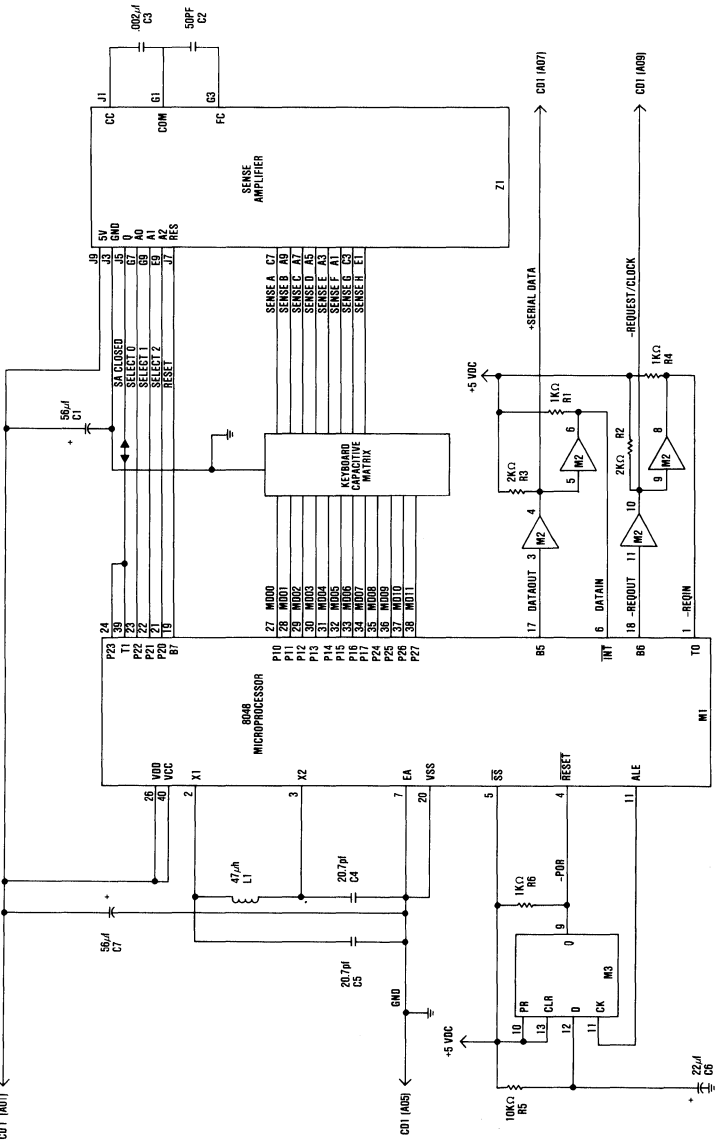


	Pin Side	Wire Side	Wire Side
Clock	1	4	6
Data	2	5	5
Ground	4	3	4
+5 Volts	5	2	2

Modular connector pins 1 and 6 are connected to the ground wire going to the chassis.

The ground wire at the keyboard connector is attached to the ground screw on the keyboard logic board.

Keyboard Logic Diagram



IBM Keyboard (Sheet 1 of 1)

Section 4

SECTION 5. SYSTEM BIOS

Contents

System BIOS Usage

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 assembler 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 this section.

Access to the BIOS is through the 8088 software interrupts. Each BIOS entry point is available through its own interrupt.

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

invokes the BIOS routine for determining memory size and returns the value to the caller.

Parameter Passing

All parameters passed to and from the BIOS routines go through the 8088 registers. The prologue 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 is in the prologue of each BIOS function.

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 system. 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 forms 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 additional code points are required.

Interrupt Hex 40 - Reserved

When an IBM Fixed Disk Adapter is installed, the BIOS routines use interrupt hex 30 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 system. 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 from 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)	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 Characters	0
100-103	40	Diskette pointer save area for Fixed Disk	
104-107	41	Fixed Disk Parameters	FD_TBL
168-16B	5A	Cluster	D000:XXXX
16C-16F	5B	Used by Cluster Program	
180-19F	60-67	Reserved for User Programs	

8088 Software Interrupt Listing

Interrupt Hex 1B - Keyboard Break Address

This vector points to the code to be used 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 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.

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		Reserved for DOS and BASIC
500	DOS	Print Screen Status Flag Store 0-Print Screen Operation Not Active or Successful Print Screen Operation 1-Print Screen In Progress 255-Error Encountered during Print Screen Operation
504	DOS	Single Drive Mode Status Byte
510-511	BASIC	BASIC's Segment Address Store
512-515	BASIC	Clock Interrupt Vector Segment: Offset Store
516-519	BASIC	Break Key Interrupt Vector Segment: Offset Store
51A-51D	BASIC	Disk Error Interrupt Vector Segment: Offset Store

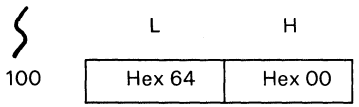
Reserved Memory Locations

If you do DEF SEG (Default workspace segment):

	Offset (Hex Value)	Length
Line number of current line being executed	2E	2
Line number of last error	347	2
Offset into segment of start of program text	30	2
Offset into segment of start of variables (end of program text 1-1)	358	2
Keyboard buffer contents if 0-no characters in buffer if 1-characters in buffer	6A	1
Character color in graphics mode Set to 1, 2, or 3 to get text in colors 1 to 3. Do not set to 0. (Default = 3)	4E	1

Example

```
100 Print  PEEK (&H2E) + 256*PEEK (&H2F)
```



BASIC Workspace Variables

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 application programs. 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 that 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.

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

Character Codes (Part 1 of 3)

Key Number	Base Case	Upper Case	Ctrl	Alt
18	e	E	ENQ (005)	Note 1
19	r	R	DC2 (018)	Note 1
20	t	T	DC4 (020)	Note 1
21	y	Y	EM (025)	Note 1
22	u	U	NAK (021)	Note 1
23	i	I	HT (009)	Note 1
24	o	O	SI (015)	Note 1
25	p	P	DLE (016)	Note 1
26	[	{	Esc (027)	– 1
27	]	}	GS (029)	– 1
28	CR	CR	LF (010)	– 1
29 Ctrl	– 1	– 1	– 1	– 1
30	a	A	SOH (001)	Note 1
31	s	S	DC3 (019)	Note 1
32	d	D	EOT (004)	Note 1
33	f	F	ACK (006)	Note 1
34	g	G	BEL (007)	Note 1
35	h	H	BS (008)	Note 1
36	j	J	LF (010)	Note 1
37	k	K	VT (011)	Note 1
38	l	L	FF (012)	Note 1
39	;	:	– 1	– 1
40	'	"	– 1	– 1
41	`	~	– 1	– 1
42 Shift	– 1	– 1	– 1	– 1
43	\	|	FS (028)	– 1
44	z	Z	SUB (026)	Note 1
45	x	X	CAN (024)	Note 1
46	c	C	ETX (003)	Note 1
47	v	V	SYN (022)	Note 1
48	b	B	STX (002)	Note 1
49	n	N	SO (014)	Note 1
50	m	M	CR (013)	Note 1
51	,	<	– 1	– 1
52	.	>	– 1	– 1
53	/	?	– 1	– 1
54 Shift	– 1	– 1	– 1	– 1
55	*	(Note 2)	(Note 1)	– 1
56 Alt	– 1	– 1	– 1	– 1
57	SP	SP	SP	SP
58 Caps Lock	– 1	– 1	– 1	– 1
59	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
60	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
61	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
62	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
63	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
64	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)

Character Codes (Part 2 of 3)

Key Number	Base Case	Upper Case	Ctrl	Alt
65	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
66	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
67	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
68	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)	Nul (Note 1)
69 Num Lock	– 1	– 1	Pause (Note 2)	– 1
70 Scroll Lock	– 1	– 1	Break (Note 2)	– 1

Notes: 1. Refer to "Extended Codes" in this section.
2. Refer to "Special Handling" in this section.

Character Codes (Part 3 of 3)

Keys 71 through 83 have meaning only in base case, in Num Lock (or shifted) states, or in Ctrl state. Note that the Shift key temporarily reverses the current Num Lock state.

Key Number	Num Lock	Base Case	Alt	Ctrl
71	7	Home (Note 1)	– 1	Clear Screen
72	8	↑ (Note 1)	– 1	– 1
73	9	Page Up (Note 1)	– 1	Top of Text and Home
74	-	----------	– 1	– 1
75	4	← (Note 1)	– 1	Reverse Word (Note 1)
76	5	– 1	– 1	– 1
77	6	→ (Note 1)	– 1	Advance Word (Note 1)
78	+	+	– 1	– 1
79	1	End (Note 1)	– 1	Erase to EOL (Note 1)
80	2	↓ (Note 1)	– 1	– 1
81	3	Page Down (Note 1)	– 1	Erase to EOS (Note 1)
82	0	Ins	– 1	– 1
83		Del (Notes 1,2)	Note 2	Note 2

Notes: 1. Refer to "Extended Codes" in this section.
2. Refer to "Special Handling" in this section.

Extended Codes

Extended Functions

For certain functions that cannot be represented in the standard ASCII code, an extended code is used. A character code of 000 (Nul) is returned in AL. This indicates that the system or application program should examine a second code that will indicate the actual function. Usually, but not always, this second code is the scan code of the primary key that was pressed. This code is returned in AH.

Second Code	Function
3	Nul Character
15	←
16-25	Alt Q, W, E, R, T, Y, U, I, O, P
30-38	Alt A, S, D, F, G, H, J, K, L
44-50	Alt Z, X, C, V, B, N, M
59-68	F1 to F10 Function Keys Base Case
71	Home
72	↑
73	Page Up and Home Cursor
75	←
77	→
79	End
80	↓
81	Page Down and Home Cursor
82	Ins (Insert)
83	Del (Delete)
84-93	F11 to F20 (Uppercase F1 to F10)
94-103	F21 to F30 (Ctrl F1 to F10)
104-113	F31 to F40 (Alt F1 to F10)
114	Ctrl PrtSc (Start/Stop Echo to Printer)
115	Ctrl← (Reverse Word)
116	Ctrl→ (Advance Word)
117	Ctrl End [Erase to End of Line (EOL)]
118	Ctrl PgDn [Erase to End of Screen (EOS)]
119	Ctrl Home (Clear Screen and Home)
120-131	Alt 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, -, = (Keys 2-13)
132	Ctrl PgUp (Top 25 Lines of Text and Home Cursor)

Keyboard Extended Functions

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 is available by calling an entry point in the ROM keyboard routine. The key numbers are shown on the keyboard diagram in Section 4. The following keys result in altered shift states:

Shift

This key temporarily shifts keys 2–13, 15–27, 30–41, 43–53, 55, 59–68 to uppercase (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 ASCII 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 within the keyboard routine.

Caps Lock

This key shifts keys 16–25, 30–38, and 44–50 to uppercase. Pressing the Caps Lock key a second time reverses the action. Caps Lock is handled within the keyboard routine.

Scroll Lock

This key is interpreted by appropriate application programs as indicating that use of the cursor-control keys should cause windowing over the text rather than cursor movement. Pressing the Scroll Lock key a second time 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. System reset is handled within the keyboard routine.

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 within 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.

Extended Functions

The keyboard routine does its own buffering. The keyboard buffer is large enough that few typists will ever fill it. However, if a key is pressed when the buffer is full, the key will be ignored and the "bell" will sound.

Also, the keyboard routine suppresses the typematic action of the following keys: Ctrl, Shift, Alt, Num Lock, Scroll Lock, Caps Lock, and Ins.

Keyboard Usage

This section is intended to outline a set of guidelines of key usage when performing commonly used functions.

Function	Key(s)	Comment
Home Cursor	Home	Editors; word processors
Return to outermost menu	Home	Menu driven applications
Move cursor up	↑	Full screen editor, word processor
Page up, scroll backward 25 lines and home	PgUp	Editors; word processors
Move cursor left	←Key 75	Text, command entry
Move cursor right	→	Text, command entry
Scroll to end of text Place cursor at end of line	End	Editors; word processors
Move cursor down	↓	Full screen editor, word processor
Page down, scroll forward 25 lines and home	Pg Dn	Editors; word processors
Start/Stop insert text at cursor, shift text right in buffer	Ins	Text, command entry
Delete character at cursor	Del	Text, command entry
Destructive backspace	←Key 14	Text, command entry
Tab forward	→|	Text entry
Tab reverse	|←	Text entry
Clear screen and home	Ctrl Home	Command entry
Scroll up	↑	In scroll lock mode
Scroll down	↓	In scroll lock mode
Scroll left	←	In scroll lock mode
Scroll right	→	In scroll lock mode
Delete from cursor to EOL	Ctrl End	Text, command entry
Exit/Escape	Esc	Editor, 1 level of menu, and so on
Start/Stop Echo screen to printer	Ctrl Prt Sc (Key 55)	Any time
Delete from cursor to EOS	Ctrl PgDn	Text, command entry
Advance word	Ctrl →	Text entry
Reverse word	Ctrl ←	Text entry
Window Right	Ctrl →	When text is too wide to fit screen
Window Left	Ctrl ←	When text is too wide to fit screen
Enter insert mode	Ins	Line editor

Keyboard - Commonly Used Functions (Part 1 of 2)

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

BASIC Screen Editor Special Functions

Function	Key
Suspend	Ctrl Num Lock
Echo to printer	Ctrl PrtSc (Key 55 any case)
Stop echo to printer	Ctrl PrtSc (Key 55 any case)
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

DOS Special Functions

System BIOS Listing

Quick Reference

```
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/CNTER 0 PORT ADDR
  0001                  25      THINT           EQU     01              ; TIMER 0 INTR RECVD 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_CTL          EQU     61H             ; CONTROL BITS FOR KEYBOARD SENSE DATA
                        34
                        35      ;----------------------------------------
                        36      ;       8088 INTERRUPT LOCATIONS        :
                        37      ;----------------------------------------
                        38
  ----                  39      ABS0    SEGMENT AT 0
  0000                  40      STG_LOC0        LABEL   BYTE
  0008                  41              ORG     2*4
  0008                  42      NMI_PTR         LABEL    WORD
  0014                  43              ORG     5*4
  0014                  44      INT5_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     1DH*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_LOCN       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 DUP(?)        ; 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_NEC         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 = 1800B0H
                        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

----                      219    XXDATA  SEGMENT AT 50H
0000 ??                   220    STATUS_BYTE     DB      ?
----                      221    XXDATA  ENDS
                          222    ;--------------------------------------
                          223    ;       VIDEO DISPLAY BUFFER            :
                          224    ;--------------------------------------
----                      225    VIDEO_RAM       SEGMENT AT 0B800H
0000                      226    REGEN   LABEL   BYTE
0000                      227    REGENW  LABEL   WORD
0000 (16384               228            DB      16384 DUP(?)
     ??
     )
----                      229    VIDEO_RAM       ENDS
                          230    ;--------------------------------------
                          231    ;       ROM RESIDENT CODE               :
                          232    ;--------------------------------------
----                      233    CODE    SEGMENT AT 0F000H
0000 (57344               234            DB      57344 DUP(?)            ; FILL LOWEST 56K
     ??
     )
                          235
E000 31353031353132       236            DB      '1501512 COPR. IBM 1981'        ; COPYRIGHT NOTICE
     20434F50522E20
     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 204B42204F4B         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 MANUFACTUING 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 TRANSFERED             :
                          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 INCRIMENT
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 READ 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,4CCCH                ; 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
                        298
E054 EA00050000         299             JMP     MFG_TEST_RTN            ; FAR JUMP TO CODE THAT WAS JUST
                        300                                             ; LOADED
                        301
                        302     ;---------------------------------------
                        303     ;       8088 PROCESSOR TEST             :
                        304     ; DESCRIPTION                           :
                        305     ;       VERIFY 8088 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 8088 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     ;---------------------------------------
E0AE                    369     C10:
                        370                                             ; ZERO IN AL ALREADY
E0AE E6A0               371             OUT     0A0H,AL                 ; DISABLE NMI INTERRUPTS
E0B0 E683               372             OUT     83H,AL                  ; INITIALZE DMA PAGE REG
E0B2 BAD803             373             MOV     DX,3D8H
E0B5 EE                 374             OUT     DX,AL                   ; DISABLE COLOR VIDEO
E0B6 FEC0               375             INC     AL
E0B8 B2B8               376             MOV     DL,0B8H
E0BA EE                 377             OUT     DX,AL                   ; DISABLE B/W VIDEO,EN HIGH RES
E0BB B089               378             MOV     AL,89H                  ; SET 8255 FOR B,A=OUT, C=IN
E0BD E663               379             OUT     CMD_PORT,AL
E0BF B0A5               380             MOV     AL,10100101B
                        381                                             ; ENABLE PARITY CHECKERS AND
E0C1 E661               382             OUT     PORT_B,AL               ; PULL KB CLOCK HI, TRI-STATE
                        383                                             ; KEYBOARD INPUTS,ENABLE HIGH
                        384                                             ; BANK OF SWITCHES->PORT C(0-3)
E0C3 B001               385             MOV     AL,01H                  ; <><><><><><><><><><><><>
E0C5 E660               386             OUT     PORT_A,AL               ; <><><>CHECKPOINT 1<><><>
E0C7 8CC8               387             MOV     AX,CS                   ; SETUP SS SEG REG
E0C9 8ED0               388             MOV     SS,AX
E0CB 8ED8               389             MOV     DS,AX                   ; SET UP DATA SEG TO POINT TO
                        390                                             ; ROM ADDRESS
E0CD FC                 391             CLD                             ; SET DIRECTION FLAG TO INC.
                        392             ASSUME  SS:CODE
E0CE BB00E0             393             MOV     BX,0E000H               ; SETUP STARTING ROS ADDR
E0D1 BC16E0             394             MOV     SP,OFFSET C1            ; SETUP RETURN ADDRESS
E0D4 E91B18             395             JMP     ROS_CHECKSUM
E0D7 7504               396     C11:    JNE     ERR01                   ; 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
E0D9 B002               409             MOV     AL,02H                  ; <><><><><><><><><><><><>
E0DB E660               410             OUT     PORT_A,AL               ; <><><>CHECKPOINT 2<><><>
E0DD B004               411             MOV     AL,04                   ; DISABLE DMA CONTROLLER
E0DF E608               412             OUT     DMA08,AL
                        413
                        414     ;----- VERIFY THAT TIMER 1 FUNCTIONS OK
                        415
E0E1 B054               416             MOV     AL,54H                  ; SEL TIMER 1,LSB,MODE 2
E0E3 E643               417             OUT     TIMER+3,AL
E0E5 8AC1               418             MOV     AL,CL                   ; SET INITIAL TIMER CNT TO 0
E0E7 E641               419             OUT     TIMER+1,AL
E0E9                    420     C12:                                    ; TIMER1_BITS_ON
E0E9 B040               421             MOV     AL,40H                  ; LATCH TIMER 1 COUNT
E0EB E643               422             OUT     TIMER+3,AL
E0ED 80FBFF             423             CMP     BL,0FFH                 ; YES - SEE IF ALL BITS GO OFF
E0F0 7407               424             JE      C13                     ; TIMER1_BITS_OFF
E0F2 E441               425             IN      AL,TIMER+1              ; READ TIMER 1 COUNT
E0F4 0AD8               426             OR      BL,AL                   ; ALL BITS ON IN TIMER
E0F6 E2F1               427             LOOP    C12                     ; TIMER1_BITS_ON
E0F8 F4                 428             HLT                             ; TIMER 1 FAILURE, HALT SYS
E0F9                    429     C13:                                    ; TIMER1_BITS_OFF
E0F9 8AC3               430             MOV     AL,BL                   ; SET TIMER 1 CNT
E0FB 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

E10B E2F2               441             LOOP    C14                     ; TIMER_LOOP
E10D F4                 442             HLT                             ; HALT SYSTEM
                        443
                        444     ;----- INITIALIZE TIMER 1 TO REFRESH MEMORY
                        445
E10E B003               446     C15:    MOV     AL,03H                  ; <><><><><><><><><><><><>
E110 E660               447             OUT     PORT_A,AL               ; <><><>CHECKPOINT 3<><><>
                        448                                             ; WRAP_DMA_REG
E112 E60D               449             OUT     DMA+0DH,AL              ; SEND MASTER CLEAR TO DMA
                        450
                        451     ;----- WRAP DMA CHANNELS ADDRESS AND COUNT REGISTERS
                        452
E114 B0FF               453             MOV     AL,0FFH                 ; WRITE PATTERN FF TO ALL REGS
E116 8AD8               454     C16:    MOV     BL,AL                   ; SAVE PATTERN FOR COMPARE
E118 8AF8               455             MOV     BH,AL
E11A B90800             456             MOV     CX,8                    ; SETUP LOOP CNT
E11D BA0000             457             MOV     DX,DMA                  ; SETUP I/O PORT ADDR OF REG
E120 EE                 458     C17:    OUT     DX,AL                   ; WRITE PATTERN TO REG, LSB
E121 50                 459             PUSH    AX                      ; SATISIFY 8237 I/O TIMINGS
E122 EE                 460             OUT     DX,AL                   ; MSB OF 16 BIT REG
E123 B001               461             MOV     AL,01H                  ; AL TO ANOTHER PAT BEFORE RD
E125 EC                 462             IN      AL,DX                   ; READ 16-BIT DMA CH REG, LSB
E126 8AE0               463             MOV     AH,AL                   ; SAVE LSB OF 16-BIT REG
E128 EC                 464             IN      AL,DX                   ; READ MSB OF DMA CH REG
E129 3BD8               465             CMP     BX,AX                   ; PATTERN READ AS WRITTEN?
E12B 7401               466             JE      C18                     ; YES - CHECK NEXT REG
E12D F4                 467             HLT                             ; NO - HALT THE SYSTEM
E12E                    468     C18:                                    ; NXT_DMA_CH
E12E 42                 469             INC     DX                      ; SET I/O PORT TO NEXT CH REG
E12F E2EF               470             LOOP    C17                     ; WRITE PATTERN TO NEXT REG
E131 FEC0               471             INC     AL                      ; SET PATTERN TO 0
E133 74E1               472             JZ      C16                     ; WRITE TO CHANNEL REGS
                        473
                        474     ;----- INITIALIZE AND START DMA FOR MEMORY REFRESH.
                        475
E135 8EDB               476             MOV     DS,BX                   ; SET UP ABS0 INTO DS AND ES
E137 8EC3               477             MOV     ES,BX
                        478             ASSUME  DS:ABS0,ES:ABS0
E139 B0FF               479             MOV     AL,0FFH                 ; SET CNT OF 64K FOR REFRESH
E13B E601               480             OUT     DMA+1,AL
E13D 50                 481             PUSH    AX
E13E E601               482             OUT     DMA+1,AL
E140 B058               483             MOV     AL,058H                 ; SET DMA MODE,CH 0,RD.,AUOTINT
E142 E60B               484             OUT     DMA+0BH,AL              ; WRITE DMA MODE REG
E144 B000               485             MOV     AL,0                    ; ENABLE DMA CONTROLLER
E146 8AE8               486             MOV     CH,AL                   ; SET COUNT HIGH=00
E148 E608               487             OUT     DMA+8,AL                ; SETUP DMA COMMAND REG
E14A 50                 488             PUSH    AX
E14B E60A               489             OUT     DMA+10,AL               ; ENABLE DMA CH 0
E14D B012               490             MOV     AL,18                   ; START TIMER 1
E14F E641               491             OUT     TIMER+1,AL
E151 B041               492             MOV     AL,41H                  ; SET MODE FOR CHANNEL 1
E153 E60B               493             OUT     DMA+0BH,AL
E155 50                 494             PUSH    AX
E156 E408               495             IN      AL,DMA+08               ; GET DMA STATUS
E158 2410               496             AND     AL,00010000B            ; IS TIMER REQUEST THERE?
E15A 7401               497             JZ      C18C                    ; (IT SHOULD'T BE)
E15C F4                 498             HLT                             ; HALT SYS.(NOT TIMER 1 OUTPUT)
E15D B042               499     C18C:   MOV     AL,42H                  ; SET MODE FOR CHANNEL 2
E15F E60B               500             OUT     DMA+0BH,AL
E161 B043               501             MOV     AL,43H                  ; SET MODE FOR CHANNEL 3
E163 E60B               502             OUT     DMA+0BH,AL
                        503     ;-----------------------------------------------
                        504     ;       BASE 16K READ/WRITE STORAGE TEST        :
                        505     ; DESCRIPTION                                   :
                        506     ;       WRITE/READ/VERIFY DATA PATTERNS         :
                        507     ;       AA,55,FF,01, AND 00 TO 1ST 32K OF       :
                        508     ;       STORAGE. VERIFY STORAGE ADDRESSABILITY. :
                        509     ;-----------------------------------------------
                        510
                        511     ;----- DETERMINE MEMORY SIZE AND FILL MEMORY WITH DATA
                        512
E165 BA1302             513             MOV     DX,0213H                ; ENABLE I/O EXPANSION BOX
E168 B001               514             MOV     AL,01H
E16A EE                 515             OUT     DX,AL
                        516
E16B 8B1E7204           517             MOV     BX,DATA_WORD[OFFSET RESET_FLAG] ; SAVE 'RESET_FLAG' IN BX
```

```
LOC OBJ                 LINE    SOURCE

E16F B90020             518             MOV     CX,2000H                ; SET FOR 16K WORDS
E172 81FB3412           519             CMP     BX,1234H                ; WARM START?
E176 7416               520             JE      CLR_STG
E178 BC18E0             521             MOV     SP,OFFSET C2
E17B E9F104             522             JMP     STGTST_CNT
E17E 7412               523     C24:    JE      HOW_BIG                 ; STORAGE OK, DETERMINE SIZE
E180 8AD8               524             MOV     BL,AL                   ; SAVE FAILING BIT PATTERN
E182 B004               525             MOV     AL,04H                  ; <><><><><><><><><><><><>
E184 E660               526     C24A:   OUT     PORT_A,AL               ; <><><>CHECKPOINT 4<><><>
E186 2BC9               527             SUB     CX,CX                   ; BASE RAM FAILURE - HANG
E188 E2FE               528     C24B:   LOOP    C24B                    ; FLIPPING BETWEEN 04 AND
E18A 86D8               529             XCHG    BL,AL                   ; FAILING BIT PATTERN
E18C EBF6               530             JMP     C24A
E18E                    531     CLR_STG:
E18E 2BC0               532             SUB     AX,AX                   ; MAKE AX=0000
E190 F3                 533             REP     STOSW                   ; STORE 8K WORDS OF 0000
E191 AB
E192                    534     HOW_BIG:
E192 891E7204           535             MOV     DATA_WORD[OFFSET RESET_FLAG],BX ; RESTORE RESET FLAG
E196 BA0004             536             MOV     DX,0400H                ; SET POINTER TO JUST>16KB
E199 BB1000             537             MOV     BX,16                   ; BASIC COUNT OF 16K
E19C                    538     FILL_LOOP:
E19C 8EC2               539             MOV     ES,DX                   ; SET SEG. REG.
E19E 2BFF               540             SUB     DI,DI
E1A0 B855AA             541             MOV     AX,0AA55H               ; TEST PATTERN
E1A3 8BC8               542             MOV     CX,AX                   ; SAVE PATTERN
E1A5 268905             543             MOV     ES:[DI],AX              ; SEND PATTERN TO MEM.
E1A8 B00F               544             MOV     AL,0FH                  ; PUT SOMETHING IN AL
E1AA 268B05             545             MOV     AX,ES:[DI]              ; GET PATTERN
E1AD 33C1               546             XOR     AX,CX                   ; COMPARE PATTERNS
E1AF 7511               547             JNZ     HOW_BIG_END             ; GO END IF NO COMPARE
E1B1 B90020             548             MOV     CX,2000H                ; SET COUNT FOR 8K WORDS
E1B4 F3                 549             REP     STOSW                   ; FILL 8K WORDS
E1B5 AB
E1B6 81C20004           550             ADD     DX,400H                 ; POINT TO NEXT 16KB BLOCK
E1BA 83C310             551             ADD     BX,16                   ; BUMP COUNT BY 16KB
E1BD 80FEA0             552             CMP     DH,0A0H                 ; TOP OF RAM AREA YET? (A0000)
E1C0 75DA               553             JNZ     FILL_LOOP
E1C2                    554     HOW_BIG_END:
E1C2 891E1304           555             MOV     DATA_WORD[OFFSET MEMORY_SIZE],BX        ; SAVE MEMORY SIZE
                        556
                        557     ;----- SETUP STACK SEG AND SP
                        558
E1C6 B83000             559             MOV     AX,STACK                ; GET STACK VALUE
E1C9 8ED0               560             MOV     SS,AX                   ; SET THE STACK UP
E1CB BC0001             561             MOV     SP,OFFSET TOS           ; STACK IS READY TO GO
                        562     ;--------------------------------------------------------
                        563     ;       INITIALIZE THE 8259 INTERRUPT CONTROLLER CHIP   :
                        564     ;--------------------------------------------------------
E1CE B013               565     C25:    MOV     AL,13H                  ; ICW1 - EDGE, SNGL, ICW4
E1D0 E620               566             OUT     INTA00,AL
E1D2 B008               567             MOV     AL,8                    ; SETUP ICW2 - INT TYPE 8 (8-F)
E1D4 E621               568             OUT     INTA01,AL
E1D6 B009               569             MOV     AL,9                    ; SETUP ICW4 - BUFFRD,8086 MODE
E1D8 E621               570             OUT     INTA01,AL
E1DA B0FF               571             MOV     AL,0FFH                 ; MASK ALL INTS. OFF
E1DC E621               572             OUT     INTA01,AL               ; (VIDEO ROUTINE ENABLES INTS.)
                        573
                        574     ;----- SET UP THE INTERRUPT VECTORS TO TEMP INTERRUPT
                        575
E1DE 1E                 576             PUSH    DS
E1DF B92000             577             MOV     CX,32                   ; FILL ALL 32 INTERRUPTS
E1E2 2BFF               578             SUB     DI,DI                   ; FIRST INTERRUPT LOCATION
E1E4 8EC7               579             MOV     ES,DI                   ; SET ES=0000 ALSO
E1E6 B823FF             580     D3:     MOV     AX,OFFSET D11           ; MOVE ADDR OF INTR PROC TO TBL
E1E9 AB                 581             STOSW
E1EA 8CC8               582             MOV     AX,CS                   ; GET ADDR OF INTR PROC SEG
E1EC AB                 583             STOSW
E1ED E2F7               584             LOOP    D3                      ; VECTBL0
                        585
                        586     ;----- ESTABLISH BIOS SUBROUTINE CALL INTERRUPT VECTORS
                        587
E1EF BF4000             588             MOV     DI,OFFSET VIDEO_INT     ; SETUP ADDR TO INTR AREA
E1F2 0E                 589             PUSH    CS
E1F3 1F                 590             POP     DS                      ; SETUP ADDR OF VECTOR TABLE
E1F4 8CD8               591             MOV     AX,DS                   ; SET AX=SEGMENT
E1F6 BE03FF90           592             MOV     SI,OFFSET VECTOR_TABLE+16       ; START WITH VIDEO ENTRY
```

```
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 E9EFFD             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:                                     ; TEST_VIDEO:
E26A 3C30               664             CMP     AL,30H                  ; B/W CARD ATTACHED?
E26C 7408               665             JE      E8                      ; YES - SET MODE FOR B/W CARD
E26E FEC4               666             INC     AH                      ; SET COLOR MODE FOR COLOR CD
E270 3C20               667             CMP     AL,20H                  ; 80X25 MODE SELECTED?
E272 7502               668             JNE     E8                      ; NO - SET MODE FOR 40X25
E274 B403               669             MOV     AH,3                    ; 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 SWS 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,0B8H                 ; 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
                        692             ASSUME  DS:NOTHING,ES:NOTHING
E2A5 E8C703             693             CALL    STGTST_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 SWS (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     ;----- UNNATURAL ACT FOR ADDRESS COMPATIBILITY
                        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
E2C8 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

E2EA 22C4               746             AND     AL,AH                   ; CHECK VIDEO/HORZ LINE
E2EC 7411               747             JZ      E16                     ; ITS ON - CHECK NEXT LINE
E2EE E2F9               748             LOOP    E15                     ; LOOP IF OFF TILL IT GOES ON
E2F0                    749     E17:                                    ; CRT_ERR:
E2F0 1F                 750             POP     DS
E2F1 1E                 751             PUSH    DS
E2F2 C606150006         752             MOV     DS:MFG_ERR_FLAG,06H     ; <><><>CRT ERR CHKPT. 06<><><>
E2F7 BA0201             753             MOV     DX,102H
E2FA E8DB16             754             CALL    ERR_BEEP                ; GO BEEP SPEAKER
E2FD EB06               755             JMP     SHORT E18
E2FF                    756     E16:                                    ; NXT_LINE:
E2FF B103               757             MOV     CL,3                    ; GET NEXT BIT TO CHECK
E301 D2EC               758             SHR     AH,CL
E303 75D7               759             JNZ     E12                     ; GO CHECK HORIZONTAL LINE
E305                    760     E18:                                    ; DISPLAY_CURSOR:
E305 58                 761             POP     AX                      ; GET VIDEO SENSE SWS (AH)
E306 B400               762             MOV     AH,0                    ; SET MODE AND DISPLAY CURSOR
E308 CD10               763             INT     10H                     ; CALL VIDEO I/O PROCEDURE
E30A                    764     E18_1:
E30A BA00C0             765             MOV     DX,0C000H               ; SEE IF ADVANCED VIDEO CARD
E30D                    766     E18A:
E30D 8EDA               767             MOV     DS,DX                   ; IS PRESENT
E30F 2BDB               768             SUB     BX,BX
E311 8B07               769             MOV     AX,[BX]                 ; GET FIRST 2 LOCATIONS
E313 53                 770             PUSH    BX
E314 5B                 771             POP     BX                      ; LET BUS SETTLE
E315 3D55AA             772             CMP     AX,0AA55H               ; PRESENT?
E318 7505               773             JNZ     E18B                    ; NO? GO LOOK FOR OTHER MODULES
E31A E83616             774             CALL    ROM_CHECK               ; GO SCAN MODULE
E31D EB04               775             JMP     SHORT E18C
E31F                    776     E18B:
E31F 81C28000           777             ADD     DX,0080H                ; POINT TO NEXT 2K BLOCK
E323                    778     E18C:
E323 81FA00C8           779             CMP     DX,0C800H               ; TOP OF VIDEO ROM AREA YET?
E327 7CE4               780             JL      E18A                    ; GO SCAN FOR ANOTHER MODULE
                        781     ;--------------------------------------------------------
                        782     ;       8259 INTERRUPT CONTROLLER TEST                  :
                        783     ; DESCRIPTION                                           :
                        784     ;       READ/WRITE THE INTERRUPT MASK REGISTER (IMR)    :
                        785     ;       WITH ALL ONES AND ZEROES. ENABLE SYSTEM         :
                        786     ;       INTERRUPTS.  MASK DEVICE INTERRUPTS OFF. CHECK  :
                        787     ;       FOR HOT INTERRUPTS (UNEXPECTED).                :
                        788     ;--------------------------------------------------------
                        789             ASSUME  DS:ABS0
E329 1F                 790     C21:    POP     DS
                        791
                        792     ;----- TEST THE IMR REGISTER
                        793
E32A C606150405         794     C21A:   MOV     DATA_AREA[OFFSET MFR_ERR_FLAG],05H
                        795                                             ; <><><><><><><><><><><><>
                        796                                             ; <><><>CHECKPOINT 5<><><>
E32F B000               797             MOV     AL,0                    ; SET IMR TO ZERO
E331 E621               798             OUT     INTA01,AL
E333 E421               799             IN      AL,INTA01               ; READ IMR
E335 0AC0               800             OR      AL,AL                   ; IMR = 0?
E337 751B               801             JNZ     D6                      ; GO TO ERR ROUTINE IF NOT 0
E339 B0FF               802             MOV     AL,0FFH                 ; DISABLE DEVICE INTERRUPTS
E33B E621               803             OUT     INTA01,AL               ; WRITE TO IMR
E33D E421               804             IN      AL,INTA01               ; READ IMR
E33F 0401               805             ADD     AL,1                    ; ALL IMR BIT ON?
E341 7511               806             JNZ     D6                      ; NO - GO TO ERR ROUTINE
                        807
                        808     ;----- CHECK FOR HOT INTERRUPTS
                        809
                        810     ;----- INTERRUPTS ARE MASKED OFF.  CHECK THAT NO INTERRUPTS OCCUR.
                        811
E343 A26B04             812             MOV     DATA_AREA[OFFSET INTR_FLAG],AL  ; CLEAR INTERRUPT FLAG
E346 FB                 813             STI                             ; ENABLE EXTERNAL INTERRUPTS
E347 2BC9               814             SUB     CX,CX                   ; WAIT 1 SEC FOR ANY INTRS THAT
E349                    815     D4:
E349 E2FE               816             LOOP    D4                      ; MIGHT OCCUR
E34B                    817     D5:
E34B E2FE               818             LOOP    D5
E34D 803E6B0400         819             CMP     DATA_AREA[OFFSET INTR_FLAG],00H ; DID ANY INTERRUPTS OCCUR?
E352 7409               820             JZ      D7                      ; NO - GO TO NEXT TEST
E354                    821     D6:
E354 BEFFF890           822             MOV     SI,OFFSET E0            ; DISPLAY 101 ERROR
```

```
LOC OBJ                 LINE    SOURCE

E358 E84E16             823             CALL    E_MSG
E35B FA                 824             CLI
E35C F4                 825             HLT                             ; HALT THE SYSTEM
                        826     ;-------------------------------------------------------
                        827     ;       8253 TIMER CHECKOUT                             :
                        828     ; DESCRIPTION                                           :
                        829     ;       VERIFY THAT THE SYSTEM TIMER (0) DOESN'T COUNT  :
                        830     ;       TOO FAST OR TOO SLOW.                           :
                        831     ;-------------------------------------------------------
E35D                    832     D7:
E35D C606150402         833             MOV     DATA_AREA[OFFSET MFR_ERR_FLAG],02H
                        834                                             ; <><><><><><><><><><><><><><><><>
                        835                                             ; <><><>TIMER CHECKPOINT (2)<><><>
E362 B0FE               836             MOV     AL,0FEH                 ; MASK ALL INTRS EXCEPT LVL 0
E364 E621               837             OUT     INTA01,AL               ; WRITE THE 8259 IMR
E366 B010               838             MOV     AL,00010000B            ; SEL TIM 0, LSB, MODE 0, BINARY
E368 E643               839             OUT     TIM_CTL,AL              ; WRITE TIMER CONTROL MODE REG
E36A B91600             840             MOV     CX,16H                  ; SET PGM LOOP CNT
E36D 8AC1               841             MOV     AL,CL                   ; SET TIMER 0 CNT REG
E36F E640               842             OUT     TIMER0,AL               ; WRITE TIMER 0 CNT REG
E371                    843     D8:
E371 F6066B0401         844             TEST    DATA_AREA[OFFSET INTR_FLAG],01H
                        845                                             ; DID TIMER 0 INTERRUPT OCCUR?
E376 7504               846             JNZ     D9                      ; YES - CHECK TIMER OP FOR SLOW TIME
E378 E2F7               847             LOOP    D8                      ; WAIT FOR INTR FOR SPECIFIED TIME
E37A EBD8               848             JMP     D6                      ; TIMER 0 INTR DIDN T OCCUR - ERR
E37C                    849     D9:
E37C B10C               850             MOV     CL,12                   ; SET PGM LOOP CNT
E37E B0FF               851             MOV     AL,0FFH                 ; WRITE TIMER 0 CNT REG
E380 E640               852             OUT     TIMER0,AL
E382 C6066B0400         853             MOV     DATA_AREA[OFFSET INTR_FLAG],0   ; RESET INTR RECEIVED FLAG
E387 B0FE               854             MOV     AL,0FEH                 ; REENABLE TIMER 0 INTERRUTS
E389 E621               855             OUT     INTA01,AL
E38B                    856     D10:
E38B F6066B0401         857             TEST    DATA_AREA[OFFSET INTR_FLAG],01H ; DID TIMER 0 INTERRUPT OCCUR?
E390 75C2               858             JNZ     D6                      ; YES - TIMER CNTING TOO FAST, ERR
E392 E2F7               859             LOOP    D10                     ; WAIT FOR INTR FOR SPECIFIED TIME
                        860
                        861     ;----- SETUP TIMER 0 TO MODE 3
                        862
E394 B0FF               863             MOV     AL,0FFH                 ; DISABLE ALL DEVICE INTERRUPTS
E396 E621               864             OUT     INTA01,AL
E398 B036               865             MOV     AL,36H                  ; SEL TIM 0,LSB,MSB,MODE 3
E39A E643               866             OUT     TIMER+3,AL              ; WRITE TIMER MODE REG
E39C B000               867             MOV     AL,0
E39E E640               868             OUT     TIMER,AL                ; WRITE LSB TO TIMER 0 REG
E3A0 E640               869             OUT     TIMER,AL                ; WRITE MSB TO TIMER 0 REG
                        870     ;-------------------------------------------------
                        871     ;       KEYBOARD TEST                           :
                        872     ; DESCRIPTION                                   :
                        873     ;       RESET THE KEYBOARD AND CHECK THAT SCAN  :
                        874     ;       CODE  AA' IS RETURNED TO THE CPU.       :
                        875     ;       CHECK FOR STUCK KEYS.                   :
                        876     ;-------------------------------------------------
E3A2                    877     TST12:
E3A2 B099               878             MOV     AL,99H                  ; SET 8255 MODE A,C=IN B=OUT
E3A4 E663               879             OUT     CMD_PORT,AL
E3A6 A01004             880             MOV     AL,DATA_AREA[OFFSET EQUIP_FLAG]
E3A9 2401               881             AND     AL,01                   ; TEST CHAMBER?
E3AB 7431               882             JZ      F7                      ; BYPASS IF SO
E3AD 803E120401         883             CMP     DATA_AREA[OFFSET MFG_TST],1     ; MANUFACTURING TEST MODE?
E3B2 742A               884             JE      F7                      ; YES - SKIP KEYBOARD TEST
E3B4 E87316             885             CALL    KBD_RESET               ; ISSUE RESET TO KEYBRD
E3B7 E31E               886             JCXZ    F6                      ; PRINT ERR MSG IF NO INTERRUPT
E3B9 B049               887             MOV     AL,49H                  ; ENABLE KEYBOARD
E3BB E661               888             OUT     PORT_B,AL
E3BD 80FBAA             889             CMP     BL,0AAH                 ; SCAN CODE AS EXPECTED?
E3C0 7515               890             JNE     F6                      ; NO - DISPLAY ERROR MSG
                        891
                        892     ;----- CHECK FOR STUCK KEYS
                        893
E3C2 B0C8               894             MOV     AL,0C8H                 ; CLR KBD, SET CLK LINE HIGH
E3C4 E661               895             OUT     PORT_B,AL
E3C6 B048               896             MOV     AL,48H                  ; ENABLE KBD,CLK IN NEXT BYTE
E3C8 E661               897             OUT     PORT_B,AL
E3CA 2BC9               898             SUB     CX,CX
E3CC                    899     F5:                                     ; KBD_WAIT:
```

```
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             CMP     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 C706140054FF       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             CMP     DATA_AREA[OFFSET MFG_TST],01H   ; MFG. TEST MODE?
E40C 750A               936             JNZ     EXP_IO
E40E C70670003CF9       937             MOV     WORD PRT[1CH*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='1801'                                             :
                        945     ;-------------------------------------------------------------
                        946
                        947     ;----- DETERMINE IF BOX IS PRESENT
                        948
E418                    949     EXP_IO:                                 ; (CARD WAS ENABLED EARLIER)
E418 BA1002             950             MOV     DX,0210H                ; CONTROL PORT ADDRESS
E41B B85555             951             MOV     AX,5555H                ; SET DATA PATTERN
E41E EE                 952             OUT     DX,AL
E41F B001               953             MOV     AL,01H                  ; MAKE AL DIFFERENT
E421 EC                 954             IN      AL,DX                   ; RECOVER DATA
E422 3AC4               955             CMP     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             CMP     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             CMP     AL,BH
E440 7521               975             JNE     EXP_ERR                 ; GO ERROR IF MISCOMPARE
E442 42                 976             INC     DX                      ; DX=216H (ADDR. LOW REG)
```

```
LOC OBJ                 LINE    SOURCE

E443 EC                  977            IN      AL,DX
E444 3AC3                978            CMP     AL,BL                   ; COMPARE TO LOW ADDRESS
E446 751B                979            JNE     EXP_ERR
E448 4A                  980            DEC     DX                      ; DX BACK TO 215H
E449 D1E3                981            SHL     BX,1
E44B E2EC                982            LOOP    EXP3                    ; LOOP TILL '1' WALKS ACROSS BX
                         983
                         984    ;----- CHECK DATA BUS
                         985
E44D 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                   ; RETRIVE VALUE FROM REG
E459 3AC4                994            CMP     AL,AH                   ; = TO SAVED VALUE
E45B 7506                995            JNE     SHORT EXP_ERR
E45D 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 E83F15             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 E8EC15             1011            CALL    DDS
E46D 1E                 1012            PUSH    DS
E46E                    1013    E20:
E46E 813E72003412       1014            CMP     RESET_FLAG,1234H        ; WARM 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 83EB10             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 81C30004           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 E8CF01             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 E8DE14             1053            CALL    PRT_HEX
```

```
LOC OBJ                 LINE    SOURCE

E4BE E2FA               1054            LOOP    PRT_DEC_LOOP
E4C0 B90700             1055            MOV     CX,7
E4C3 BE1AE0             1056            MOV     SI,OFFSET F3B           ; PRINT ' KB OK'
E4C6                    1057    KB_LOOP:
E4C6 2E8A04             1058            MOV     AL,CS:[SI]
E4C9 46                 1059            INC     SI
E4CA E8CF14             1060            CALL    PRT_HEX
E4CD E2F7               1061            LOOP    KB_LOOP
E4CF 58                 1062            POP     AX                      ; RECOVER WORK REGS
E4D0 3D1000             1063            CMP     AX,16                   ; FIRST PASS?
E4D3 74A9               1064            JE      E20B
E4D5 5B                 1065            POP     BX
E4D6 59                 1066            POP     CX
E4D7 5A                 1067            POP     DX
E4D8 E2B4               1068            LOOP    E21                     ; LOOP TILL ALL MEM. CHECKED
E4DA B00A               1069            MOV     AL,10
E4DC E8BD14             1070            CALL    PRT_HEX                 ; LINE FEED
                        1071
                        1072    ;----- DMA TC0 SHOULD BE ON BY NOW - SEE IF IT IS
                        1073
E4DF E408               1074            IN      AL,DMA+08H
E4E1 2401               1075            AND     AL,00000001B            ; TC0 STATUS BIT ON?
E4E3 7533               1076            JNZ     ROM_SCAN                ; GO ON WITH NEXT TEST IF OK
E4E5 1F                 1077            POP     DS
E4E6 C606150003         1078            MOV     MFG_ERR_FLAG,03H        ; <><><><><><><><><><><><><><>
E4EB E966FE             1079            JMP     D6                      ; POST 101 ERROR MSG AND HALT
                        1080
                        1081    ;----- PRINT FAILING ADDRESS AND XOR'ED PATTERN IF DATA COMPARE ERROR
                        1082
E4EE 8AE8               1083    E21A:   MOV     CH,AL                   ; SAVE FAILING BIT PATTERN
E4F0 B00D               1084            MOV     AL,13                   ; CARRAGE RETURN
E4F2 E8A714             1085            CALL    PRT_HEX
E4F5 B00A               1086            MOV     AL,10
E4F7 E8A214             1087            CALL    PRT_HEX
E4FA 58                 1088            POP     AX                      ; RECOVER AMT. OF GOOD MEM.
E4FB 83C406             1089            ADD     SP,6                    ; BALANCE STACK
E4FE 8CDA               1090            MOV     DX,DS                   ; GET FAILING SEGMENT
E500 1F                 1091            POP     DS
E501 1E                 1092            PUSH    DS
E502 A31300             1093            MOV     MEMORY_SIZE,AX          ; LOAD MEM. SIZE WORD TO SHOW
                        1094                                            ; HOW MUCH MEM. WORKING
E505 88361500           1095            MOV     MFG_ERR_FLAG,DH         ; <><><><><><><><><><><><><><>
                        1096                                            ; <><>CHECKPOINTS 08->A0<><>
E509 E8CE1A             1097            CALL    PRT_SEG                 ; PRINT IT
E50C 8AC5               1098            MOV     AL,CH                   ; GET FAILING BIT PATTERN
E50E E87A14             1099            CALL    XPC_BYTE                ; CONVERT AND PRINT CODE
E511 BE04F990           1100            MOV     SI,OFFSET E1            ; SETUP ADDRESS OF ERROR MSG
E515 E89114             1101            CALL    E_MSG                   ; PRINT ERROR MSG
                        1102    ;---------------------------------------------------------------
                        1103    ; CHECK FOR OPTIONAL ROM FROM C8000->F4000 IN 2K BLOCKS         :
                        1104    ;       (A VALID MODULE HAS '55AA' IN THE FIRST 2 LOCATIONS,    :
                        1105    ;       LENGTH INDICATOR (LENGTH/512) IN THE 3D LOCATION AND    :
                        1106    ;       TEST/INIT. CODE STARTING IN THE 4TH LOCATION.)          :
                        1107    ;---------------------------------------------------------------
E518                    1108    ROM_SCAN:
E518 BA00C8             1109            MOV     DX,0C800H               ; SET BEGINNING ADDRESS
E51B                    1110    ROM_SCAN_1:
E51B 8EDA               1111            MOV     DS,DX
E51D 2BDB               1112            SUB     BX,BX                   ; SET BX=0000
E51F 8B07               1113            MOV     AX,[BX]                 ; GET 1ST WORD FROM MODULE
E521 53                 1114            PUSH    BX
E522 5B                 1115            POP     BX                      ; BUS SETTLING
E523 3D55AA             1116            CMP     AX,0AA55H               ; = TO ID WORD?
E526 7506               1117            JNZ     NEXT_ROM                ; PROCEED TO NEXT ROM IF NOT
E528 E82814             1118            CALL    ROM_CHECK               ; GO CHECK OUT MODULE
E52B EB0590             1119            JMP     ARE_WE_DONE             ; CHECK FOR END OF ROM SPACE
E52E                    1120    NEXT_ROM:
E52E 81C28000           1121            ADD     DX,0080H                ; POINT TO NEXT 2K ADDRESS
E532                    1122    ARE_WE_DONE:
E532 81FA00F6           1123            CMP     DX,0F600H               ; AT F6000 YET?
E536 7CE3               1124            JL      ROM_SCAN_1              ; GO CHECK ANOTHER ADD. IF NOT
E538 EB0190             1125            JMP     BASE_ROM_CHK            ; GO CHECK BASIC ROM
                        1126    ;---------------------------------------------------------------------------
                        1127    ; A CHECKSUM IS DONE FOR THE 4 ROS MODULES CONTAINING BASIC CODE        :
                        1128    ;---------------------------------------------------------------------------
E53B                    1129    BASE_ROM_CHK:
E53B B404               1130            MOV     AH,4                    ; NO. OF ROS MODULES TO CHECK
```

```
LOC OBJ                 LINE    SOURCE

E53D                    1131    E4:
E53D 2BDB               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    E5:
E549 81C20002           1139            ADD     DX,0200H                ; POINT TO NEXT 8K MODULE
E54D FECC               1140            DEC     AH                      ; ANY MORE TO DO?
E54F 75EC               1141            JNZ     E4                      ; YES - CONTINUE
                        1142    ;--------------------------------------------------------------
                        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 1F                 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    ;----- 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 33D2               1176            XOR     DX,DX                   ; SELECT DRIVE 0
E578 B501               1177            MOV     CH,1                    ; SELECT TRACK 1
E57A 88163E00           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 E81614             1186            CALL    E_MSG                   ; GO PRINT ERROR MSG
                        1187
                        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    ;----- SETUP PRINTER AND RS232 BASE ADDRESSES IF DEVICE ATTACHED
                        1196
E597                    1197    F15:
E597 C6066B0000         1198            MOV     INTR_FLAG,00H           ; SET STRAY INTERRUPT FLAG = 00
E59C BE1E00             1199            MOV     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

E5B7 B81414             1208            MOV     AX,1414H                ; DEFAULT=20
E5BA AB                 1209            STOSW
E5BB AB                 1210            STOSW
E5BC B80101             1211            MOV     AX,0101H                ;RS232 DEFAULT=01
E5BF AB                 1212            STOSW
E5C0 AB                 1213            STOSW
E5C1 E421               1214            IN      AL,INTA01
E5C3 24FC               1215            AND     AL,0FCH                 ; ENABLE TIMER AND KB INTS
E5C5 E621               1216            OUT     INTA01,AL
E5C7 83FD00             1217            CMP     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            CMP     AH,3BH
E5E0 75F7               1228            JNE     ERR_WAIT
E5E2 EB0E90             1229            JMP     F15A                    ; BYPASS ERROR
E5E5                    1230    F15A_0:
E5E5 803E120001         1231            CMP     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 B0AA               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            CMP     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            CMP     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 BAFA03             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 BAFA02             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 8805               1329            MOV     [DI],AL                 ; ON FIRST BYTE
E676 8A05               1330            MOV     AL,[DI]
E678 32C4               1331            XOR     AL,AH                   ; O.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 STG ?
E6AB 7416               1362            JZ      C6X                     ; YES - RETURN TO CALLER WITH AL=0
```

```
LOC OBJ                 LINE    SOURCE

E6AD 8AE0               1363            MOV     AH,AL                   ; SETUP NEW VALUE FOR COMPARE
E6AF 86F2               1364            XCHG    DH,DL                   ; MOVE NEXT DATA PATTERN TO DL
E6B1 22E4               1365            AND     AH,AH                   ; READING ZERO PATTERN THIS PASS ?
E6B3 7504               1366            JNZ     C6                      ; CONTINUE TEST SEQUENCE TILL ZERO DATA
E6B5 8AD4               1367            MOV     DL,AH                   ; ELSE SET ZERO FOR END READ PATTERN
E6B7 EBE0               1368            JMP     C3                      ; AND MAKE FINAL BACKWARDS PASS
E6B9                    1369    C6:
E6B9 FC                 1370            CLD                             ; SET DIR FLAG TO GO FORWARD
E6BA 47                 1371            INC     DI                      ; SET POINTER TO BEG LOCATION
E6BB 74DE               1372            JZ      C4                      ; READ/WRITE FORWARD IN STG
E6BD 4F                 1373            DEC     DI                      ; ADJUST POINTER
E6BE BA0100             1374            MOV     DX,00001H               ; SETUP 01 FOR PARITY BIT AND 00 FOR END
E6C1 EBD6               1375            JMP     C3                      ; READ/WRITE BACKWARD IN STG
E6C3                    1376    C6X:
E6C3 E462               1377            IN      AL,PORT_C               ; DID A PARITY ERROR OCCUR ?
E6C5 24C0               1378            AND     AL,0C0H                 ; ZERO FLAG WILL BE OFF PARITY ERROR
E6C7 B000               1379            MOV     AL,000H                 ; AL=0 DATA COMPARE OK
E6C9                    1380    C7:
E6C9 FC                 1381            CLD                             ; SET DIRECTION FLAG TO INC
E6CA C3                 1382            RET
                        1383    STGTST_CNT      ENDP
                        1384    ;--------------------------------------------------------------------------
                        1385    ;  PRINT ADDRESS AND ERROR MESSAGE FOR ROM CHECKSUM ERRORS               :
                        1386    ;--------------------------------------------------------------------------
E6CB                    1387    ROM_ERR PROC    NEAR
E6CB 52                 1388            PUSH    DX                      ; SAVE POINTER
E6CC 50                 1389            PUSH    AX
E6CD 8CDA               1390            MOV     DX,DS                   ; GET ADDRESS POINTER
E6CF 2688361500         1391            MOV     ES:MFG_ERR_FLAG,DH      ; <><><><><><><><><><><><><><>
                        1392                                            ; <><><>CHECKPOINTS C0->F4<><><>
E6D4 81FA00C8           1393            CMP     DX,0C800H               ; CRT CARD IN ERROR?
E6D8 7C0D               1394            JL      ROM_ERR_BEEP            ; GIVE CRT CARD FAIL BEEP
E6DA E8FD18             1395            CALL    PRT_SEG                 ; PRINT SEGEMENT IN ERROR
E6DD BE0AF990           1396            MOV     SI,OFFSET F3A           ; DISPLAY ERROR MSG
E6E1 E8C512             1397            CALL    E_MSG
E6E4                    1398    ROM_ERR_END:
E6E4 58                 1399            POP     AX
E6E5 5A                 1400            POP     DX
E6E6 C3                 1401            RET
E6E7                    1402    ROM_ERR_BEEP:
E6E7 BA0201             1403            MOV     DX,0102H                ; BEEP 1 LONG, 2 SHORT
E6EA E8EB12             1404            CALL    ERR_BEEP
E6ED EBF5               1405            JMP     SHORT ROM_ERR_END
                        1406    ROM_ERR ENDP
                        1407
                        1408    ;--- INT 19 ----------------------------------------------
                        1409    ; BOOT STRAP LOADER                                     :
                        1410    ;       TRACK 0, SECTOR 1 IS READ INTO THE              :
                        1411    ;       BOOT LOCATION (SEGMENT 0, OFFSET 7C00)          :
                        1412    ;       AND CONTROL IS TRANSFERRED THERE.               :
                        1413    ;                                                       :
                        1414    ;       IF THERE IS A HARDWARE ERROR CONTROL IS         :
                        1415    ;       TRANSFERRED TO THE ROM BASIC ENTRY POINT.       :
                        1416    ;-------------------------------------------------------
                        1417            ASSUME  CS:CODE,DS:ABS0
E6F2                    1418            ORG     0E6F2H
                        1419
E6F2                    1420    BOOT_STRAP      PROC    NEAR
E6F2 FB                 1421            STI                             ; ENABLE INTERRUPTS
E6F3 2BC0               1422            SUB     AX,AX                   ; ESTABLISH ADDRESSING
E6F5 8ED8               1423            MOV     DS,AX
                        1424
                        1425    ;----- RESET THE DISK PARAMETER TABLE VECTOR
                        1426
E6F7 C7067800C7EF       1427            MOV     WORD PTR DISK_POINTER, OFFSET DISK_BASE
E6FD 8C0E7A00           1428            MOV     WORD PTR DISK_POINTER+2,CS
                        1429
                        1430    ;-----  LOAD SYSTEM FROM DISKETTE -- CX HAS RETRY COUNT
                        1431
E701 B90400             1432            MOV     CX,4                    ; SET RETRY COUNT
E704                    1433    H1:                                     ; IPL_SYSTEM
E704 51                 1434            PUSH    CX                      ; SAVE RETRY COUNT
E705 B400               1435            MOV     AH,0                    ; RESET THE DISKETTE SYSTEM
E707 CD13               1436            INT     13H                     ; DISKETTE_IO
E709 720F               1437            JC      H2                      ; IF ERROR, TRY AGAIN
E70B B80102             1438            MOV     AX,201H                 ; READ IN THE SINGLE SECTOR
E70E 2BD2               1439            SUB     DX,DX                   ; TO THE BOOT LOCATION
```

```
LOC OBJ                 LINE    SOURCE

E710 8EC2               1440            MOV     ES,DX
E712 BB007C             1441            MOV     BX,OFFSET BOOT_LOCN
                        1442                                    ; 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    ;                       THE 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:                                     ; RETURN FROM RS232
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 75DB               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 75D3               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 8A5D7C             1685            MOV     BL,RS232_TIM_OUT[DI]    ; 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
     28524553554D45
     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                             ; INTERRUPTS BACK ON
E82F 1E                 1729            PUSH    DS                      ; SAVE CURRENT DS
E830 53                 1730            PUSH    BX                      ; SAVE BX TEMPORARILY
E831 E82512             1731            CALL    DDS
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 E81D00             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     $-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
                         1813    ;----- CTL TABLE SCAN
E8C8                     1814    K9      LABEL   BYTE
E8C8 5E                  1815                    DB      94,95,96,97,98,99,100,101
E8C9 5F
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 73                  1817                    DB      115,-1,116,-1,117,-1,118,-1
E8D9 FF
E8DA 74
E8DB FF
E8DC 75
E8DD FF
```

```
LOC OBJ                  LINE    SOURCE

E8DE 76
E8DF FF
E8E0 FF                  1818                    DB      -1
                         1819    ;----- LC TABLE
E8E1                     1820    K10     LABEL   BYTE
E8E1 1B                  1821                    DB      01BH,'1234567890-=',08H,09H
E8E2 31323334353637
     3839302D3D
E8EE 08
E8EF 09
E8F0 71776572747975      1822                    DB      'qwertyuiop[]',0DH,-1,'asdfghjkl;',027H
     696F705B5D
E8FC 0D
E8FD FF
E8FE 61736466676 86A
     6B6C3B
E908 27
E909 60                  1823                    DB      60H,-1,5CH,'zxcvbnm,./',-1,'*',-1,' '
E90A FF
E90B 5C
E90C 7A7863766268 6D
     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 51575452545955      1828                    DB      'QWERTYUIOP{}',0DH,-1,'ASDFGHJKL:"'
     494F507B7D
E936 0D
E937 FF
E938 41534446474848A
     4B4C3A22
E943 7E                  1829                    DB      07EH,-1,'|ZXCVBNM<>?',-1,0,-1,' ',-1
E944 FF
E945 7C5A5843564248E
     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     1840                    DB      '789-456+1230.'
     2B3132333302E
                        1841    ;----- BASE CASE TABLE
E976                    1842    K15     LABEL   BYTE
E976 47                 1843                    DB      71,72,73,-1,75,-1,77
E977 48
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
                        1846    ;----- KEYBOARD INTERRUPT ROUTINE
                        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 0C80               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 B90800             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    ;----- SHIFT KEY FOUND
                        1892
E9C0 81EF7FE8           1893    K17:    SUB     DI,OFFSET K6+1          ; ADJUST PTR TO SCAN CODE MTCH
E9C4 2E8AA586E8         1894            MOV     AH,CS:K7[DI]            ; 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
E9D0 7307               1901            JAE     K18                     ; IF SCROLL SHIFT OR ABOVE, TOGGLE KEY
                        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    ;----- TOGGLED SHIFT KEY, TEST FOR 1ST MAKE OR NOT
                        1909
E9D9                    1910    K18:                                    ; SHIFT-TOGGLE
E9D9 F606170004         1911            TEST    KB_FLAG, CTL_SHIFT      ; CHECK CTL SHIFT STATE
E9DE 7565               1912            JNZ     K25                     ; JUMP IF CTL STATE
E9E0 3C52               1913            CMP     AL, INS_KEY             ; CHECK FOR INSERT KEY
E9E2 7522               1914            JNZ     K22                     ; JUMP IF NOT INSERT KEY
E9E4 F606170008         1915            TEST    KB_FLAG, ALT_SHIFT      ; CHECK FOR ALTERNATE SHIFT
E9E9 755A               1916            JNZ     K25                     ; JUMP IF ALTERNATE SHIFT
E9EB F606170020         1917    K19:    TEST    KB_FLAG, NUM_STATE      ; CHECK FOR BASE STATE
E9F0 750D               1918            JNZ     K21                     ; JUMP IF NUM LOCK IS ON
E9F2 F606170003         1919            TEST    KB_FLAG, LEFT_SHIFT+ RIGHT_SHIFT
E9F7 740D               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 F606170003         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 754D               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            CMP     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            CMP     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            CMP     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            CMP     AL,0                    ; WAS THE INPUT=0
EA38 741F               1955            JE      K26                     ; INTERRUPT_RETURN
EA3A E9A101             1956            JMP     K58                     ; IT WASN'T, SO PUT IN BUFFER
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            CMP     AL,80H                  ; TEST FOR BREAK KEY
EA47 7310               1966            JAE     K26                     ; NOTHING FOR BREAK CHARS FROM HERE ON
EA49 F606180008         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            CMP     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 C706720003412      2005            MOV     RESET_FLAG, 1234H       ; SET FLAG FOR RESET FUNCTION
EA82 EA5BE000F0         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
EAA0 23
EAA1 24                 2015            DB      36,37,38,44,45,46,47,48
EAA2 25
EAA3 26
EAA4 2C
EAA5 2D
EAA6 2E
EAA7 2F
EAA8 30
EAA9 31                 2016            DB      49,50
EAAA 32
                        2017
                        2018    ;----- IN ALTERNATE SHIFT, RESET NOT FOUND
                        2019
EAAB                    2020    K31:                                    ; NO-RESET
EAAB 3C39               2021            CMP     AL,57                   ; TEST FOR SPACE KEY
EAAD 7505               2022            JNE     K32                     ; NOT THERE
EAAF B020               2023            MOV     AL,' '                  ; SET SPACE CHAR
EAB1 E92101             2024            JMP     K57                     ; BUFFER_FILL
                        2025
                        2026    ;----- LOOK FOR KEY PAD ENTRY
                        2027
```

```
LOC OBJ                 LINE    SOURCE

EAB4                    2028    K32:                                    ; ALT-KEY-PAD
EAB4 BF87EA             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
EAC5 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 C606190000         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 PSUEDO SCAN CODE TO RANGE
EAEC B000               2059            MOV     AL,0                    ; INDICATE AS SUCH
EAEE 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
EAFA 73F9               2071            JAE     K36                     ; IF SO, IGNORE
EAFC BB5FE9             2072            MOV     BX,OFFSET K13           ; ALT SHIFT PSEUDO SCAN TABLE
EAFF 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,EOI                  ; 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            CMP     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 BB8EE8             2123            MOV     BX,OFFSET K8            ; SET UP TO TRANSLATE CTL
EB57 3C3B               2124            CMP     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 BBC8E8             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            CMP     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            CMP     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            CMP     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 E9DCFE             2154            JMP     K27                     ; GO BACK WITHOUT EOI OCCURRING
EB82                    2155    K46:                                    ; NOT-PRINT-SCREEN
EB82 3C3B               2156            CMP     AL,59                   ; FUNCTION KEYS
EB84 7206               2157            JB      K47                     ; NOT-UPPER-FUNCTION
EB86 BB55E9             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            CMP     AL,74                   ; SPECIAL CASE FOR A COUPLE OF KEYS
EBA1 740B               2176            JE      K50                     ; MINUS
EBA3 3C4E               2177            CMP     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
EBBD 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 BB69E9             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
EC1D 2C3B               2266            SUB     AL,59                   ; CONVERT ORIGIN TO FUNCTION KEYS
EC1F                    2267    K64:                                    ; TRANSLATE-SCAN-ORGD
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
EC51 0A
EC52 363031             2301    F3      DB      '601',13,10             ; DISKETTE ERROR
EC55 0D
EC56 0A
                        2302
                        2303    ;-- INT 13 ------------------------------------------------------------
                        2304    ; DISKETTE I/O                                                        :
                        2305    ;       THIS INTERFACE PROVIDES ACCESS TO THE 5 1/4 DISKETTE DRIVES    :
                        2306    ; INPUT                                                               :
                        2307    ;       (AH)=0  RESET DISKETTE SYSTEM                                 :
                        2308    ;               HARD RESET TO NEC, PREPARE COMMAND, RECAL REQUIRED     :
                        2309    ;               ON ALL DRIVES                                         :
                        2310    ;       (AH)=1  READ THE STATUS OF THE SYSTEM INTO (AL)               :
                        2311    ;               DISKETTE_STATUS FROM LAST OPERATION IS USED            :
                        2312    ;                                                                     :
                        2313    ; REGISTERS FOR READ/WRITE/VERIFY/FORMAT                              :
                        2314    ;       (DL) - DRIVE NUMBER (0-3 ALLOWED, VALUE CHECKED)              :
                        2315    ;       (DH) - HEAD NUMBER (0-1 ALLOWED, NOT VALUE CHECKED)           :
                        2316    ;       (CH) - TRACK NUMBER (0-39, NOT VALUE CHECKED)                 :
                        2317    ;       (CL) - SECTOR NUMBER (1-8, NOT VALUE CHECKED,                 :
                        2318    ;                                   NOT USED FOR FORMAT)              :
                        2319    ;       (AL) - NUMBER OF SECTORS ( MAX = 8, NOT VALUE CHECKED, NOT USED :
                        2320    ;                                          FOR FORMAT)                :
                        2321    ;       (ES:BX) - ADDRESS OF BUFFER ( NOT REQUIRED FOR VERIFY)        :
                        2322    ;                                                                     :
                        2323    ;       (AH)=2  READ THE DESIRED SECTORS INTO MEMORY                  :
                        2324    ;       (AH)=3  WRITE THE DESIRED SECTORS FROM MEMORY                 :
                        2325    ;       (AH)=4  VERIFY THE DESIRED SECTORS                            :
                        2326    ;       (AH)=5  FORMAT THE DESIRED TRACK                              :
                        2327    ;               FOR THE FORMAT OPERATION, THE BUFFER POINTER (ES,BX)   :
                        2328    ;               MUST POINT TO THE COLLECTION OF DESIRED ADDRESS FIELDS :
                        2329    ;               FOR THE TRACK.  EACH FIELD IS COMPOSED OF 4 BYTES,     :
```

```
LOC OBJ                 LINE    SOURCE

                        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
EC5A 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 E8F30D             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 88264000           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            CMP     AH,1                    ; SET THE CARRY FLAG TO INDICATE
EC7A F5                 2376            CMC                             ;  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            CMP     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_CMD ; 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,8                    ; 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
ECDC 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 OWN TEST
ECEB 3CC0               2442            CMP     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 E8B801             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 RW_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 EBF5               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 RW_OPN            ; DO THE OPERATION
ED26                    2492    J10:                                    ; CONTINUATION OF RW_OPN FOR FMT
ED26 BB0700             2493            MOV     BX,7                    ; GET THE
ED29 E84001             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 RW_OPN
                        2513
                        2514    ;--------------------------------------------------------------
                        2515    ; RW_OPN                                                      :
                        2516    ;       THIS ROUTINE PERFORMS THE READ/WRITE/VERIFY OPERATION   :
                        2517    ;--------------------------------------------------------------
ED4A                    2518    RW_OPN  PROC    NEAR
ED4A 7308               2519            JNC     J11                     ; TEST FOR DMA ERROR
ED4C C606410009         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_RW_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 8ACA               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 E80F00             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 BEF0ED90           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 80FF4D             2590            CMP     BH,04DH                 ; 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 BB0700             2600            MOV     BX,7                    ; BYTES/SECTOR PARM FROM BLOCK
EDD7 E89200             2601            CALL    GET_PARM                ;  TO THE NEC
EDDA BB0900             2602            MOV     BX,9                    ; EOT PARM FROM BLOCK
EDDD E88C00             2603            CALL    GET_PARM                ;  TO THE NEC
EDE0 BB0B00             2604            MOV     BX,11                   ; GAP LENGTH PARM FROM BLOCK
EDE3 E88600             2605            CALL    GET_PARM                ;  TO THE NEC
EDE6 BB0D00             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            CMP     AL,040H                 ; TEST FOR ABNORMAL TERMINATION
EE02 7529               2628            JNZ     J18                     ; 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                     ; RW_FAIL
EE13 D0E0               2640            SAL     AL,1                    ; TEST FOR DMA OVERRUN
EE15 B408               2641            MOV     AH,BAD_DMA
EE17 7216               2642            JC      J19                     ; RW_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                     ; RW_FAIL
EE21 D0E0               2647            SAL     AL,1
EE23 B403               2648            MOV     AH,WRITE_PROTECT        ; TEST FOR WRITE_PROTECT
EE25 7208               2649            JC      J19                     ; RW_FAIL
EE27 D0E0               2650            SAL     AL,1                    ; TEST MISSING ADDRESS MARK
EE29 B402               2651            MOV     AH,BAD_ADDR_MARK
EE2B 7202               2652            JC      J19                     ; RW_FAIL
                        2653
                        2654    ;----- NEC MUST HAVE FAILED
                        2655
EE2D                    2656    J18:                                    ; RW-NEC-FAIL
EE2D B420               2657            MOV     AH,BAD_NEC
EE2F                    2658    J19:                                    ; RW-FAIL
EE2F 08264100           2659            OR      DISKETTE_STATUS,AH
EE33 E87801             2660            CALL    NUM_TRANS               ; HOW MANY WERE REALLY TRANSFERRED
EE36                    2661    J20:                                    ; RW_ERR
EE36 C3                 2662            RET                             ; RETURN TO CALLER
EE37                    2663    J21:                                    ; RW_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    RW_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    ;--------------------------------------------------------------------------
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 WAS 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 840643E00          2770            TEST    AL,SEEK_STATUS          ; TEST FOR RECAL REQUIRED
EE89 7513               2771            JNZ     J28                     ; NO_RECAL
EE8B 800643E00          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 INTERUPT 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
```

```
LOC OBJ                 LINE    SOURCE

                        2792    ;----- WAIT FOR HEAD SETTLE
                        2793
EEB0 9C                 2794            PUSHF                           ; SAVE STATUS FLAGS
EEB1 BB1200             2795            MOV     BX,18                   ; 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
EEBB 0AE4               2800            OR      AH,AH                   ; TEST FOR TIME EXPIRED
EEBD 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                                                             :
                        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,0F0H                 ; ZERO THE LOW NYBBLE FROM SEGMENT
EEDA 03C3               2833            ADD     AX,BX                   ; TEST FOR CARRY FROM ADDITION
EEDC 7302               2834            JNC     J33
EEDE FEC5               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 E681               2843            OUT     081H,AL                 ; OUTPUT THE HIGH 4 BITS TO
                        2844                                            ;  THE PAGE REGISTER
                        2845
                        2846    ;----- DETERMINE COUNT
                        2847
EEED 8AE6               2848            MOV     AH,DH                   ; NUMBER OF SECTORS
EEEF 2AC0               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
```

```
LOC OBJ                 LINE    SOURCE

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 F8                 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
                        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    J38:                                    ; INPUT_LOOP
EF72 33C9               2983            XOR     CX,CX                   ; COUNTER
EF74 BAF403             2984            MOV     DX,03F4H                ; STATUS PORT
EF77                    2985    J39:                                    ; WAIT FOR MASTER
EF77 EC                 2986            IN      AL,DX                   ; GET STATUS
EF78 A880               2987            TEST    AL,080H                 ; MASTER READY
EF7A 750C               2988            JNZ     J40A                    ; TEST_DIR
EF7C E2F9               2989            LOOP    J39                     ; WAIT_MASTER
EF7E 800E410080         2990            OR      DISKETTE_STATUS,TIME_OUT
EF83                    2991    J40:                                    ; RESULTS_ERROR
EF83 F9                 2992            STC                             ; SET ERROR RETURN
EF84 5B                 2993            POP     BX
EF85 5A                 2994            POP     DX
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                 ; TEST DIRECTION BIT
EF8B 7507               3003            JNZ     J42                     ; OK TO READ STATUS
EF8D                    3004    J41:                                    ; NEC_FAIL
EF8D 800E410020         3005            OR      DISKETTE_STATUS,BAD_NEC
EF92 EBEF               3006            JMP     J40                     ; RESULTS_ERROR
                        3007
                        3008    ;----- READ IN THE STATUS
                        3009
EF94                    3010    J42:                                    ; INPUT_STAT
EF94 42                 3011            INC     DX                      ; POINT AT DATA PORT
EF95 EC                 3012            IN      AL,DX                   ; GET THE DATA
EF96 8805               3013            MOV     [DI],AL                 ; STORE THE BYTE
EF98 47                 3014            INC     DI                      ; INCREMENT THE POINTER
EF99 B90A00             3015            MOV     CX,10                   ; LOOP TO KILL TIME FOR NEC
EF9C E2FE               3016    J43:    LOOP    J43
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

EFA8 EBE3               3023            JMP     J41                     ; CHIP HAS FAILED
                        3024
                        3025    ;----- RESULT OPERATION IS DONE
                        3026
EFAA                    3027    J44:
EFAA 5B                 3028            POP     BX
EFAB 5A                 3029            POP     DX
EFAC 59                 3030            POP     CX                      ; RECOVER REGISTERS
EFAD 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,8
EFBB E8AEFE             3049            CALL    GET_PARM                ; GET EOT VALUE
EFBE 8AC4               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    ;                                                                          :
```

```
LOC OBJ                 LINE    SOURCE

                        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
EFD2                    3111            ORG     0EFD2H
EFD2                    3112    PRINTER_IO      PROC    FAR
EFD2 FB                 3113            STI                             ; INTERRUPTS BACK ON
EFD3 1E                 3114            PUSH    DS                      ; SAVE SEGMENT
EFD4 52                 3115            PUSH    DX
EFD5 56                 3116            PUSH    SI
EFD6 51                 3117            PUSH    CX
EFD7 53                 3118            PUSH    BX
EFD8 E87E0A             3119            CALL    DDS
EFDB 8BF2               3120            MOV     SI,DX                   ; GET PRINTER PARM
EFDD 8A5C78             3121            MOV     BL,PRINT_TIM_OUT[SI]    ; LOAD TIME-OUT PARM
EFE0 D1E6               3122            SHL     SI,1                    ; WORD OFFSET INTO TABLE
EFE2 8B5408             3123            MOV     DX,PRINTER_BASE[SI]     ; GET BASE ADDRESS FOR PRINTER CARD
EFE5 0BD2               3124            OR      DX,DX                   ; TEST DX FOR ZERO,
                        3125                                            ;  INDICATING NO PRINTER
EFE7 740C               3126            JZ      B1                      ; RETURN
EFE9 0AE4               3127            OR      AH,AH                   ; TEST FOR (AH)=0
EFEB 740E               3128            JZ      B2                      ; PRINT_AL
EFED FECC               3129            DEC     AH                      ; TEST FOR (AH)=1
EFEF 743F               3130            JZ      B8                      ; INIT_PRT
EFF1 FECC               3131            DEC     AH                      ; TEST FOR (AH)=2
EFF3 7428               3132            JZ      B5                      ; PRINTER STATUS
EFF5                    3133    B1:                                     ; RETURN
EFF5 5B                 3134            POP     BX
EFF6 59                 3135            POP     CX
EFF7 5E                 3136            POP     SI                      ; RECOVER REGISTERS
EFF8 5A                 3137            POP     DX                      ; RECOVER REGISTERS
EFF9 1F                 3138            POP     DS
EFFA CF                 3139            IRET
                        3140
                        3141    ;------ PRINT THE CHARACTER IN (AL)
                        3142
EFFB                    3143    B2:
EFFB 50                 3144            PUSH    AX                      ; SAVE VALUE TO PRINT
EFFC EE                 3145            OUT     DX,AL                   ; OUTPUT CHAR TO PORT
EFFD 42                 3146            INC     DX                      ; POINT TO STATUS PORT
EFFE                    3147    B3:
EFFE 2BC9               3148            SUB     CX,CX                   ; WAIT_BUSY
F000                    3149    B3_1:
F000 EC                 3150            IN      AL,DX                   ; GET STATUS
F001 8AE0               3151            MOV     AH,AL                   ; STATUS TO AH ALSO
F003 A880               3152            TEST    AL,80H                  ; IS THE PRINTER CURRENTLY BUSY
F005 750E               3153            JNZ     B4                      ; OUT_STROBE
F007 E2F7               3154            LOOP    B3_1                    ; TRY AGAIN
F009 FECB               3155            DEC     BL                      ; DROP LOOP COUNT
F00B 75F1               3156            JNZ     B3                      ; GO TILL TIMEOUT ENDS
F00D 80CC01             3157            OR      AH,1                    ; SET ERROR FLAG
F010 80E4F9             3158            AND     AH,0F9H                 ; TURN OFF THE OTHER BITS
F013 EB13               3159            JMP     SHORT B7                ; RETURN WITH ERROR FLAG SET
F015                    3160    B4:                                     ; OUT_STROBE
F015 B00D               3161            MOV     AL,0DH                  ; SET THE STROBE HIGH
F017 42                 3162            INC     DX                      ; STROBE IS BIT 0 OF PORT C OF 8255
F018 EE                 3163            OUT     DX,AL
F019 B00C               3164            MOV     AL,0CH                  ; SET THE STROBE LOW
F01B EE                 3165            OUT     DX,AL
F01C 58                 3166            POP     AX                      ; RECOVER THE OUTPUT CHAR
                        3167
                        3168    ;------ PRINTER STATUS
                        3169
F01D                    3170    B5:
F01D 50                 3171            PUSH    AX                      ; SAVE AL REG
F01E                    3172    B6:
F01E 8B5408             3173            MOV     DX,PRINTER_BASE[SI]
F021 42                 3174            INC     DX
F022 EC                 3175            IN      AL,DX                   ; GET PRINTER STATUS
F023 8AE0               3176            MOV     AH,AL
```

```
LOC OBJ                 LINE    SOURCE

F025 80E4F8             3177            AND     AH,0F8H                 ; 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 80F448             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 EBDD               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    :
```

```
3254     ;               (BH) = ATTRIBUTE TO BE USED ON BLANK LINE                :
3255     ;                                                                        :
3256     ;       CHARACTER HANDLING ROUTINES                                      :
3257     ;                                                                        :
3258     ;       (AH) = 8 READ ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION     :
3259     ;               (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)         :
3260     ;               ON EXIT:                                                 :
3261     ;               (AL) = CHAR READ                                         :
3262     ;               (AH) = ATTRIBUTE OF CHARACTER READ (ALPHA MODES ONLY)    :
3263     ;       (AH) = 9 WRITE ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION    :
3264     ;               (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)         :
3265     ;               (CX) = COUNT OF CHARACTERS TO WRITE                      :
3266     ;               (AL) = CHAR TO WRITE                                     :
3267     ;               (BL) = ATTRIBUTE OF CHARACTER (ALPHA)/COLOR OF CHAR      :
3268     ;                      (GRAPHICS)                                        :
3269     ;                       SEE NOTE ON WRITE DOT FOR BIT 7 OF BL = 1.       :
3270     ;       (AH) = 10 WRITE CHARACTER ONLY AT CURRENT CURSOR POSITION        :
3271     ;               (BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)         :
3272     ;               (CX) = COUNT OF CHARACTERS TO WRITE                      :
3273     ;               (AL) = CHAR TO WRITE                                     :
3274     ;       FOR READ/WRITE CHARACTER INTERFACE WHILE IN GRAPHICS MODE, THE    :
3275     ;               CHARACTERS ARE FORMED FROM A CHARACTER GENERATOR IMAGE    :
3276     ;               MAINTAINED IN THE SYSTEM ROM.  ONLY THE 1ST 128 CHARS    :
3277     ;               ARE CONTAINED THERE.  TO READ/WRITE THE SECOND 128       :
3278     ;               CHARS, THE USER MUST INITIALIZE THE POINTER AT           :
3279     ;               INTERRUPT 1FH (LOCATION 0007CH) TO POINT TO THE 1K BYTE  :
3280     ;               TABLE CONTAINING THE CODE POINTS FOR THE SECOND           :
3281     ;               128 CHARS (128-255).                                     :
3282     ;       FOR WRITE CHARACTER INTERFACE IN GRAPHICS MODE, THE REPLICATION   :
3283     ;               FACTOR CONTAINED IN (CX) ON ENTRY WILL PRODUCE VALID     :
3284     ;               RESULTS ONLY FOR CHARACTERS CONTAINED ON THE SAME ROW.    :
3285     ;               CONTINUATION TO SUCCEEDING LINES WILL NOT PRODUCE         :
3286     ;               CORRECTLY.                                               :
3287     ;                                                                        :
3288     ;       GRAPHICS INTERFACE                                               :
3289     ;       (AH) = 11 SET COLOR PALETTE                                      :
3290     ;               (BH) = PALETTE COLOR ID BEING SET (0-127)                :
3291     ;               (BL) = COLOR VALUE TO BE USED WITH THAT COLOR ID         :
3292     ;                  NOTE: FOR THE CURRENT COLOR CARD, THIS ENTRY POINT    :
3293     ;                        HAS MEANING ONLY FOR 320X200 GRAPHICS.          :
3294     ;                       COLOR ID = 0 SELECTS THE BACKGROUND COLOR (0-15): :
3295     ;                       COLOR ID = 1 SELECTS THE PALETTE TO BE USED:     :
3296     ;                               0 = GREEN(1)/RED(2)/YELLOW(3)            :
3297     ;                               1 = CYAN(1)/MAGENTA(2)/WHITE(3)          :
3298     ;                       IN 40X25 OR 80X25 ALPHA MODES, THE VALUE SET     :
3299     ;                               FOR PALETTE COLOR 0 INDICATES THE        :
3300     ;                               BORDER COLOR TO BE USED (VALUES 0-31,    :
3301     ;                               WHERE 16-31 SELECT THE HIGH INTENSITY    :
3302     ;                               BACKGROUND SET.                          :
3303     ;       (AH) = 12 WRITE DOT                                              :
3304     ;               (DX) = ROW NUMBER                                        :
3305     ;               (CX) = COLUMN NUMBER                                     :
3306     ;               (AL) = COLOR VALUE                                       :
3307     ;                       IF BIT 7 OF AL = 1, THEN THE COLOR VALUE IS      :
3308     ;                       EXCLUSIVE OR'D WITH THE CURRENT CONTENTS OF      :
3309     ;                       THE DOT                                          :
3310     ;       (AH) = 13 READ DOT                                               :
3311     ;               (DX) = ROW NUMBER                                        :
3312     ;               (CX) = COLUMN NUMBER                                     :
3313     ;               (AL) RETURNS THE DOT READ                                :
3314     ;                                                                        :
3315     ; ASCII TELETYPE ROUTINE FOR OUTPUT                                      :
3316     ;                                                                        :
3317     ;       (AH) = 14 WRITE TELETYPE TO ACTIVE PAGE                          :
3318     ;               (AL) = CHAR TO WRITE                                     :
3319     ;               (BL) = FOREGROUND COLOR IN GRAPHICS MODE                 :
3320     ;               NOTE -- SCREEN WIDTH IS CONTROLLED BY PREVIOUS MODE SET  :
3321     ;                                                                        :
3322     ;       (AH) = 15 CURRENT VIDEO STATE                                    :
3323     ;               RETURNS THE CURRENT VIDEO STATE                          :
3324     ;               (AL) = MODE CURRENTLY SET ( SEE AH=0 FOR EXPLANATION)    :
3325     ;               (AH) = NUMBER OF CHARACTER COLUMNS ON SCREEN              :
3326     ;               (BH) = CURRENT ACTIVE DISPLAY PAGE                       :
3327     ;                                                                        :
3328     ;       CS,SS,DS,ES,BX,CX,DX PRESERVED DURING CALL                       :
3329     ;       ALL OTHERS DESTROYED                                             :
3330     ;-------------------------------------------------------------------------
```

```
LOC OBJ                 LINE    SOURCE

                        3331            ASSUME  CS:CODE,DS:DATA,ES:VIDEO_RAM
F045                    3332            ORG     0F045H
F045                    3333    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    M1L     EQU     $-M1
                        3351
F065                    3352            ORG     0F065H
F065                    3353    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
F098 8A264900           3383            MOV     AH,CRT_MODE             ; GET CURRENT MODE INTO AH
F09C 2EFFA445F0         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                    3398            ORG     0F0A4H
F0A4                    3399    VIDEO_PARMS     LABEL   BYTE
                        3400    ;----- INIT_TABLE
F0A4 38                 3401            DB      38H,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 0008               3419            DW      2048                    ; 40X25
F0E6 0010               3420            DW      4096                    ; 80X25
F0E8 0040               3421            DW      16384                   ; GRAPHICS
F0EA 0040               3422            DW      16384
                        3423
                        3424    ;----- COLUMNS
                        3425
F0EC                    3426    M6      LABEL   BYTE
F0EC 28                 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 B007               3439            MOV     AL,7                    ; INDICATE BW CARD MODE
F108 B2B4               3440            MOV     DL,0B4H                 ; ADDRESS OF BW CARD (3B4)
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, OUTPUTTTING 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 C606620000         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 BW CARD
F168 7404               3502            JE      M11                     ; BW_CARD_INIT
F16A 33C0               3503            XOR     AX,AX                   ; FILL FOR GRAPHICS MODE
F16C EB05               3504            JMP     SHORT M13               ; CLEAR_BUFFER
F16E                    3505    M11:                                    ; BW_CARD_INIT
F16E B508               3506            MOV     CH,08H                  ; BUFFER SIZE ON BW CARD
F170                    3507    M12:                                    ; NO_GRAPHICS_INIT
F170 B82007             3508            MOV     AX,' '+7*256            ; FILL CHAR FOR ALPHA
F173                    3509    M13:                                    ; CLEAR_BUFFER
F173 F3                 3510            REP     STOSW                   ; FILL THE REGEN BUFFER WITH BLANKS
F174 AB
                        3511
                        3512    ;----- ENABLE VIDEO AND CORRECT PORT SETTING
                        3513
F175 C706660000706      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]
F18E EE                 3522            OUT     DX,AL                   ; SET VIDEO ENABLE PORT
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 2E8B8CE4F0         3535            MOV     CX,CS:[SI+OFFSET M5]    ; LENGTH TO CLEAR
F1A5 890E4C00           3536            MOV     CRT_LEN,CX              ; SAVE LENGTH OF CRT -- NOT USED FOR BW
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     STOSW                   ; FILL WITH ZEROES
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 BW
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:                                    ; VIDEO_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                                                   :
```

```
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                    3577    SET_CTYPE       PROC    NEAR
F1CD B40A               3578            MOV     AH,10                   ; 6845 REGISTER FOR CURSOR SET
F1CF 890E6000           3579            MOV     CURSOR_MODE,CX          ; SAVE IN DATA AREA
F1D3 E80200             3580            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                    3585    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                    3612    SET_CPOS        PROC    NEAR
F1EE 8ACF               3613            MOV     CL,BH
F1F0 32ED               3614            XOR     CH,CH                   ; ESTABLISH LOOP COUNT
F1F2 D1E1               3615            SAL     CX,1                    ; WORD OFFSET
F1F4 8BF1               3616            MOV     SI,CX                   ; USE INDEX REGISTER
F1F6 895450             3617            MOV     [SI+OFFSET CURSOR_POSN],DX      ; SAVE THE POINTER
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:                                    ; SET_CPOS_RETURN
F204 EBBF               3623            JMP     VIDEO_RETURN
                        3624    SET_CPOS        ENDP
                        3625
                        3626    ;----- SET CURSOR POSITION, AX HAS ROW/COLUMN FOR CURSOR
                        3627
F206                    3628    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                    3646    ACT_DISP_PAGE   PROC    NEAR
F217 A26200             3647            MOV     ACTIVE_PAGE,AL          ; SAVE ACTIVE PAGE VALUE
```

```
LOC OBJ                 LINE    SOURCE

F21A 8B0E4C00           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 07                 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    ;----- 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    M19:                                    ; OUTPUT THE PALETTE
F263 EE                 3719            OUT     DX,AL                   ; OUTPUT COLOR SELECTION TO 3D9 PORT
F264 A26600             3720            MOV     CRT_PALETTE,AL          ; SAVE THE COLOR VALUE
F267 E95BFF             3721            JMP     VIDEO_RETURN
                        3722
                        3723    ;----- 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 8BD8               3759            MOV     BX,AX
F288 8AC4               3760            MOV     AL,AH                   ; ROWS TO AL
F28A F6264A00           3761            MUL     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            CMP     AH,4                    ; TEST FOR GRAPHICS MODE
F29B 7208               3787            JC      N1                      ; HANDLE SEPARATELY
F29D 80FC07             3788            CMP     AH,7                    ; TEST FOR BW 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

F2B8 03FD               3802            ADD     DI,BP                   ; POINT TO NEXT LINE IN BLOCK
F2BA FECC               3803            DEC     AH                      ; COUNT OF LINES TO MOVE
F2BC 75F5               3804            JNZ     N2                      ; ROW_LOOP
F2BE                    3805    N3:                                     ; CLEAR_ENTRY
F2BE 58                 3806            POP     AX                      ; RECOVER ATTRIBUTE IN AH
F2BF B020               3807            MOV     AL,' '                  ; FILL WITH BLANKS
F2C1                    3808    N4:                                     ; CLEAR_LOOP
F2C1 E86D00             3809            CALL    N11                     ; CLEAR THE ROW
F2C4 03FD               3810            ADD     DI,BP                   ; POINT TO NEXT LINE
F2C6 FECB               3811            DEC     BL                      ; COUNTER OF LINES TO SCROLL
F2C8 75F7               3812            JNZ     N4                      ; CLEAR_LOOP
F2CA                    3813    N5:                                     ; SCROLL_END
F2CA E88C07             3814            CALL    DDS
F2CD 803E490007         3815            CMP     CRT_MODE,7              ; IS THIS THE BLACK AND WHITE CARD
F2D2 7407               3816            JE      N6                      ; IF SO, SKIP THE MODE RESET
F2D4 A06500             3817            MOV     AL,CRT_MODE_SET         ; GET THE VALUE OF THE MODE SET
F2D7 BAD803             3818            MOV     DX,03D8H                ; ALWAYS SET COLOR CARD PORT
F2DA EE                 3819            OUT     DX,AL
F2DB                    3820    N6:                                     ; VIDEO_RET_HERE
F2DB E9E7FE             3821            JMP     VIDEO_RETURN
F2DE                    3822    N7:                                     ; BLANK_FIELD
F2DE 8ADE               3823            MOV     BL,DH                   ; GET ROW COUNT
F2E0 EBDC               3824            JMP     N3                      ; GO CLEAR THAT AREA
                        3825    SCROLL_UP       ENDP
                        3826
                        3827    ;----- HANDLE COMMON SCROLL SET UP HERE
                        3828
F2E2                    3829    SCROLL_POSITION PROC    NEAR
F2E2 803E490002         3830            CMP     CRT_MODE,2              ; TEST FOR SPECIAL CASE HERE
F2E7 7218               3831            JB      N9                      ; HAVE TO HANDLE 80X25 SEPARATELY
F2E9 803E490003         3832            CMP     CRT_MODE,3
F2EE 7711               3833            JA      N9
                        3834
                        3835    ;----- 80X25 COLOR CARD SCROLL
                        3836
F2F0 52                 3837            PUSH    DX
F2F1 BADA03             3838            MOV     DX,3DAH                 ; GUARANTEED TO BE COLOR CARD HERE
F2F4 50                 3839            PUSH    AX
F2F5                    3840    N8:                                     ; WAIT_DISP_ENABLE
F2F5 EC                 3841            IN      AL,DX                   ; GET PORT
F2F6 A808               3842            TEST    AL,8                    ; WAIT FOR VERTICAL RETRACE
F2F8 74FB               3843            JZ      N8                      ; WAIT_DISP_ENABLE
F2FA B025               3844            MOV     AL,25H
F2FC B2D8               3845            MOV     DL,0D8H                 ; DX=3D8
F2FE EE                 3846            OUT     DX,AL                   ; TURN OFF VIDEO
F2FF 58                 3847            POP     AX                      ; DURING VERTICAL RETRACE
F300 5A                 3848            POP     DX
F301                    3849    N9:
F301 E881FF             3850            CALL    POSITION                ; CONVERT TO REGEN POINTER
F304 03064E00           3851            ADD     AX,CRT_START            ; OFFSET OF ACTIVE PAGE
F308 8BF8               3852            MOV     DI,AX                   ; TO ADDRESS FOR SCROLL
F30A 8BF0               3853            MOV     SI,AX                   ; FROM ADDRESS FOR SCROLL
F30C 2BD1               3854            SUB     DX,CX                   ; DX = # ROWS, #COLS IN BLOCK
F30E FEC6               3855            INC     DH
F310 FEC2               3856            INC     DL                      ; INCREMENT FOR 0 ORIGIN
F312 32ED               3857            XOR     CH,CH                   ; SET HIGH BYTE OF COUNT TO ZERO
F314 8B2E4A00           3858            MOV     BP,CRT_COLS             ; GET NUMBER OF COLUMNS IN DISPLAY
F318 03ED               3859            ADD     BP,BP                   ; TIMES 2 FOR ATTRIBUTE BYTE
F31A 8AC3               3860            MOV     AL,BL                   ; GET LINE COUNT
F31C F66264A00          3861            MUL     BYTE PTR CRT_COLS       ; DETERMINE OFFSET TO FROM ADDRESS
F320 03C0               3862            ADD     AX,AX                   ; *2 FOR ATTRIBUTE BYTE
F322 06                 3863            PUSH    ES                      ; ESTABLISH ADDRESSING TO REGEN BUFFER
F323 1F                 3864            POP     DS                      ;  FOR BOTH POINTERS
F324 80FB00             3865            CMP     BL,0                    ; 0 SCROLL MEANS BLANK FIELD
F327 C3                 3866            RET                             ; RETURN WITH FLAGS SET
                        3867    SCROLL_POSITION ENDP
                        3868
                        3869    ;----- MOVE_ROW
                        3870
F328                    3871    N10     PROC    NEAR
F328 8ACA               3872            MOV     CL,DL                   ; GET # OF COLS TO MOVE
F32A 56                 3873            PUSH    SI
F32B 57                 3874            PUSH    DI                      ; SAVE START ADDRESS
F32C F3                 3875            REP     MOVSW                   ; MOVE THAT LINE ON SCREEN
F32D A5
F32E 5F                 3876            POP     DI
F32F 5E                 3877            POP     SI                      ; RECOVER ADDRESSES
```

```
LOC OBJ                 LINE    SOURCE

F330 C3                 3878            RET
                        3879    N10     ENDP
                        3880
                        3881    ;----- CLEAR_ROW
                        3882
F331                    3883    N11     PROC    NEAR
F331 8ACA               3884            MOV     CL,DL                   ; GET # COLUMNS TO CLEAR
F333 57                 3885            PUSH    DI
F334 F3                 3886            REP     STOSW                   ; STORE THE FILL CHARACTER
F335 AB
F336 5F                 3887            POP     DI
F337 C3                 3888            RET
                        3889    N11     ENDP
                        3890    ;--------------------------------------------------------
                        3891    ; SCROLL_DOWN                                          :
                        3892    ;       THIS ROUTINE MOVES THE CHARACTERS WITHIN A      :
                        3893    ;       DEFINED BLOCK DOWN ON THE SCREEN, FILLING THE   :
                        3894    ;       TOP LINES WITH A DEFINED CHARACTER              :
                        3895    ; INPUT                                                :
                        3896    ;       (AH) = CURRENT CRT MODE                         :
                        3897    ;       (AL) = NUMBER OF LINES TO SCROLL                :
                        3898    ;       (CX) = UPPER LEFT CORNER OF REGION              :
                        3899    ;       (DX) = LOWER RIGHT CORNER OF REGION             :
                        3900    ;       (BH) = FILL CHARACTER                           :
                        3901    ;       (DS) = DATA SEGMENT                             :
                        3902    ;       (ES) = REGEN SEGMENT                            :
                        3903    ; OUTPUT                                               :
                        3904    ;       NONE -- SCREEN IS SCROLLED                      :
                        3905    ;--------------------------------------------------------
F338                    3906    SCROLL_DOWN     PROC    NEAR
F338 FD                 3907            STD                             ; DIRECTION FOR SCROLL DOWN
F339 8AD8               3908            MOV     BL,AL                   ; LINE COUNT TO BL
F33B 80FC04             3909            CMP     AH,4                    ; TEST FOR GRAPHICS
F33E 7208               3910            JC      N12
F340 80FC07             3911            CMP     AH,7                    ; TEST FOR BW CARD
F343 7403               3912            JE      N12
F345 E9A601             3913            JMP     GRAPHICS_DOWN
F348                    3914    N12:                                    ; CONTINUE_DOWN
F348 53                 3915            PUSH    BX                      ; SAVE ATTRIBUTE IN BH
F349 8BC2               3916            MOV     AX,DX                   ; LOWER RIGHT CORNER
F34B E894FF             3917            CALL    SCROLL_POSITION         ; GET REGEN LOCATION
F34E 7420               3918            JZ      N16
F350 2BF0               3919            SUB     SI,AX                   ; SI IS FROM ADDRESS
F352 8AE6               3920            MOV     AH,DH                   ; GET TOTAL # ROWS
F354 2AE3               3921            SUB     AH,BL                   ; COUNT TO MOVE IN SCROLL
F356                    3922    N13:
F356 E8CFFF             3923            CALL    N10                     ; MOVE ONE ROW
F359 2BF5               3924            SUB     SI,BP
F35B 2BFD               3925            SUB     DI,BP
F35D FECC               3926            DEC     AH
F35F 75F5               3927            JNZ     N13
F361                    3928    N14:
F361 58                 3929            POP     AX                      ; RECOVER ATTRIBUTE IN AH
F362 B020               3930            MOV     AL,' '
F364                    3931    N15:
F364 E8CAFF             3932            CALL    N11                     ; CLEAR ONE ROW
F367 2BFD               3933            SUB     DI,BP                   ; GO TO NEXT ROW
F369 FECB               3934            DEC     BL
F36B 75F7               3935            JNZ     N15
F36D E95AFF             3936            JMP     N5                      ; SCROLL_END
F370                    3937    N16:
F370 8ADE               3938            MOV     BL,DH
F372 EBED               3939            JMP     N14
                        3940    SCROLL_DOWN     ENDP
                        3941    ;--------------------------------------------------------
                        3942    ; READ_AC_CURRENT                                      :
                        3943    ;       THIS ROUTINE READS THE ATTRIBUTE AND CHARACTER   :
                        3944    ;       AT THE CURRENT CURSOR POSITION AND RETURNS THEM  :
                        3945    ;       TO THE CALLER                                   :
                        3946    ;INPUT                                                 :
                        3947    ;       (AH) = CURRENT CRT MODE                         :
                        3948    ;       (BH) = DISPLAY PAGE ( ALPHA MODES ONLY )        :
                        3949    ;       (DS) = DATA SEGMENT                             :
                        3950    ;       (ES) = REGEN SEGMENT                            :
                        3951    ;OUTPUT                                                :
                        3952    ;       (AL) = CHAR READ                                :
                        3953    ;       (AH) = ATTRIBUTE READ                           :
                        3954    ;--------------------------------------------------------
```

```
LOC OBJ                 LINE    SOURCE

                        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
F41E                    4119    READ_DOT        PROC    NEAR
F41E E83100             4120            CALL    R3                      ; DETERMINE BYTE POSITION OF DOT
F421 268A04             4121            MOV     AL,ES:[SI]              ; GET THE BYTE
F424 22C4               4122            AND     AL,AH                   ; MASK OFF THE OTHER BITS IN THE BYTE
F426 D2E0               4123            SHL     AL,CL                   ; LEFT JUSTIFY THE VALUE
F428 8ACE               4124            MOV     CL,DH                   ; GET NUMBER OF BITS IN RESULT
F42A D2C0               4125            ROL     AL,CL                   ; RIGHT JUSTIFY THE RESULT
F42C E996FD             4126            JMP     VIDEO_RETURN            ; RETURN FROM VIDEO IO
                        4127    READ_DOT        ENDP
                        4128
F42F                    4129    WRITE_DOT       PROC    NEAR
F42F 50                 4130            PUSH    AX                      ; SAVE DOT VALUE
F430 50                 4131            PUSH    AX                      ;  TWICE
F431 E81E00             4132            CALL    R3                      ; DETERMINE BYTE POSITION OF THE DOT
F434 D2E8               4133            SHR     AL,CL                   ; SHIFT TO SET UP THE BITS FOR OUTPUT
F436 22C4               4134            AND     AL,AH                   ; STRIP OFF THE OTHER BITS
F438 268A0C             4135            MOV     CL,ES:[SI]              ; GET THE CURRENT BYTE
F43B 5B                 4136            POP     BX                      ; RECOVER XOR FLAG
F43C F6C380             4137            TEST    BL,80H                  ; IS IT ON
F43F 750D               4138            JNZ     R2                      ; YES, XOR THE DOT
F441 F6D4               4139            NOT     AH                      ; SET THE MASK TO REMOVE THE
F443 22CC               4140            AND     CL,AH                   ;  INDICATED BITS
F445 0AC1               4141            OR      AL,CL                   ; OR IN THE NEW VALUE OF THOSE BITS
F447                    4142    R1:                                     ; FINISH_DOT
F447 268804             4143            MOV     ES:[SI],AL              ; RESTORE THE BYTE IN MEMORY
F44A 58                 4144            POP     AX
F44B E977FD             4145            JMP     VIDEO_RETURN            ; RETURN FROM VIDEO IO
F44E                    4146    R2:                                     ; XOR_DOT
F44E 32C1               4147            XOR     AL,CL                   ; EXCLUSIVE OR THE DOTS
F450 EBF5               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    ;--------------------------------------------------------
F452                    4162    R3      PROC    NEAR
F452 53                 4163            PUSH    BX                      ; SAVE BX DURING OPERATION
F453 50                 4164            PUSH    AX                      ; WILL SAVE AL DURING OPERATION
                        4165
                        4166    ;----- DETERMINE 1ST BYTE IN IDICATED ROW BY MULTIPLYING ROW VALUE BY 40
                        4167    ;----- ( LOW BIT OF ROW DETERMINES EVEN/ODD, 80 BYTES/ROW
                        4168
F454 B028               4169            MOV     AL,40
F456 52                 4170            PUSH    DX                      ; SAVE ROW VALUE
F457 80E2FE             4171            AND     DL,0FEH                 ; STRIP OFF ODD/EVEN BIT
F45A F6E2               4172            MUL     DL                      ; AX HAS ADDRESS OF 1ST BYTE
                        4173                                            ;  OF INDICATED ROW
F45C 5A                 4174            POP     DX                      ; RECOVER IT
F45D F6C201             4175            TEST    DL,1                    ; TEST FOR EVEN/ODD
F460 7403               4176            JZ      R4                      ; JUMP IF EVEN ROW
F462 050020             4177            ADD     AX,2000H                ; OFFSET TO LOCATION OF ODD ROWS
F465                    4178    R4:                                     ; EVEN_ROW
F465 8BF0               4179            MOV     SI,AX                   ; MOVE POINTER TO SI
F467 58                 4180            POP     AX                      ; RECOVER AL VALUE
F468 8BD1               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/C0H 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 8AF7               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 E88800             4300            CALL    R18                     ; CLEAR THAT ROW
F4DF 81EFB01F           4301            SUB     DI,2000H-80             ; POINT TO NEXT LINE
F4E3 FECB               4302            DEC     BL                      ; NUMBER OF LINES TO FILL
F4E5 75F5               4303            JNZ     R10                     ; CLEAR_LOOP
F4E7 E9DBFC             4304            JMP     VIDEO_RETURN            ; EVERYTHING DONE
F4EA                    4305    R11:                                    ; BLANK_FIELD
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                             ; SET DIRECTION
F4EF 8AD8               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    R18                     ; CLEAR A ROW
F53E 81EF5020           4389            SUB     DI,2000H+80             ; POINT TO NEXT LINE
F542 FECB               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 ROW
                        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                   ; STORE 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 FILELD
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            CMP     AL,80H                  ; IS IT IN SECOND HALF
F583 7306               4480            JAE     S1                      ; 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 EB0F               4486            JMP     SHORT S2                ; DETERMINE_MODE
                        4487
                        4488    ;----- IMAGE IS IN SECOND HALF, IN USER RAM
```

```
LOC OBJ                 LINE    SOURCE

                        4489
F58B                    4490    S1:                                     ; EXTEND_CHAR
F58B 2C80               4491            SUB     AL,80H                  ; ZERO ORIGIN FOR SECOND HALF
F58D 1E                 4492            PUSH    DS                      ; SAVE DATA POINTER
F58E 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
F5BB 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            LOOP    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 E8DE00             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 268825             4562            MOV     ES:[DI],AH              ; STORE FIRST BYTE
F5F6 26884501           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 PONTER
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 23D9               4719            AND     BX,CX                   ; USE MASK TO EXTRACT A BIT
F6CF 0BD3               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 8BD8               4723            MOV     BX,AX                   ; BASE TO TEMP
F6D7 23D9               4724            AND     BX,CX                   ; EXTRACT THE SAME BIT
F6D9 0BD3               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 PARM 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 8BD8               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
```

```
LOC OBJ                 LINE    SOURCE

                        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 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 80FE18             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 FEC6               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            CMP     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            CMP     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 POD 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
```

```
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:V1[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 EQUPMENT                    :
                        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 E80702             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,0C0H                 ; 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 8EDA               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 E84601             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,MEMORY_SIZE          ; GET MEMORY SIZE WORD
F896 FC                 5181            CLD                             ; SET DIR FLAG TO INCRIMENT
F897 2BD2               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 REALTIVE 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,'('
F8C9 E8D000             5207            CALL    PRT_HEX
F8CC B85AA5             5208            MOV     AX,0A55AH
F8CF 8BC8               5209            MOV     CX,AX
F8D1 2BDB               5210            SUB     BX,BX
F8D3 8907               5211            MOV     [BX],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            CMP     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
     434845434B2032
F923 0D
F924 0A
F925 50415249545920      5255    D2      DB      'PARITY CHECK 1',13,10
     434845434B2031
F933 0D
F934 0A
F935 3F3F3F3F3F          5256    D2A     DB      '?????',13,10
F93A 0D
F93B 0A
                         5257
                         5258    ;--------------------------------------------------------------------------
                         5259    ;       BLINK LED PROCEDURE FOR MFG RUN-IN TESTS
                         5260    ;        IF LED IS ON, TURN IT OFF. IF OFF, TURN ON.
                         5261    ;--------------------------------------------------------------------------
                         5262            ASSUME  DS:DATA
F93C                     5263    BLINK_INT       PROC    NEAR
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,EOI
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    ; THIS ROUTINE CHECKSUMS OPTIONAL ROM MODULES AND       :
                         5281    ; IF CHECKSUM IS OK, CALLS INIT/TEST CODE IN MODULE     :
                         5282    ;--------------------------------------------------------
F953                     5283    ROM_CHECK       PROC    NEAR
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 03D0                5294            ADD     DX,AX                   ; SET POINTER TO NEXT MODULE
F96B 59                  5295            POP     CX                      ; RETRIVE COUNT
F96C E886FF              5296            CALL    ROS_CHECKSUM_CNT        ; DO CHECKSUM
F96F 7406                5297            JZ      ROM_CHECK_1
F971 E857ED              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
```

```
LOC OBJ                 LINE    SOURCE

                        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 ASCHI 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     CMD_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:
F9C8 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                   ; WAS 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    ;-------------------------------------------------------------------------
F9D8                    5399    ERR_BEEP  PROC  NEAR
F9D8 9C                 5400            PUSHF                           ; SAVE FLAGS
F9D9 FA                 5401            CLI                             ; DISABLE*SYSTEM INTERRUPTS
F9DA 1E                 5402            PUSH    DS                      ; SAVE DS REG CONTENTS
F9DB E87B00             5403            CALL    DDS
F9DE 0AF6               5404            OR      DH,DH                   ; ANY LONG ONES TO BEEP
F9E0 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     MFG_TST,1               ; MFG TEST MODE?
F9F2 7502               5414            JNE     G3                      ; YES - CONTINUE BEEPING SPEAKER
F9F4 EBC3               5415            JMP     MFG_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 9D                 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 SETTINGH
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 FECB               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
                        5460            ASSUME  DS:ABS0
FA2A B008               5461            MOV     AL,08H                  ; SET KBD CLK LINE LOW
FA2C E661               5462            OUT     PORT_B,AL               ; WRITE 8255 PORT B
FA2E B95629             5463            MOV     CX,10582                ; HOLD KBD CLK LOW FOR 20 MS
FA31                    5464    G8:
FA31 E2FE               5465            LOOP    G8                      ; LOOP FOR 20 MS
```

```
LOC OBJ                  LINE    SOURCE

FA33 B0C8                5466            MOV     AL,0C8H                 ; 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 IMR
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            TEST    DATA_AREA[OFFSET INT R_FLAG],02H ; DID A KEYBOARD INTR OCCUR?
FA4C 7502                5478            JNZ     G10                     ; 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            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    ;       CHARACTER GENERATOR GRAPHICS FOR 320X200 AND 640X200 GRAPHICS   :
                         5498    ;--------------------------------------------------------------------------
FA6E                     5499            ORG     0FA6EH
FA6E                     5500    CRT_CHAR_GEN    LABEL   BYTE
FA6E 0000000000000000    5501            DB      000H,000H,000H,000H,000H,000H,000H,000H ; D_00
FA76 7E81A581BD99817E    5502            DB      07EH,081H,0A5H,081H,0BDH,099H,081H,07EH ; D_01
FA7E 7EFFDBFFC3E7FF7E    5503            DB      07EH,0FFH,0DBH,0FFH,0C3H,0E7H,0FFH,07EH ; D_02
FA86 6CFEFEFE7C381000    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 387C38FEFE7C387C    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 0000183C3C180000    5508            DB      000H,000H,018H,03CH,03CH,018H,000H,000H ; D_07
FAAE FFFFE7C3C3E7FFFF    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 FFC399BDBD99C3FF    5511            DB      0FFH,0C3H,099H,0BDH,0BDH,099H,0C3H,0FFH ; D_0A
FAC6 0F070F7DCCCCCC78    5512            DB      00FH,007H,00FH,07DH,0CCH,0CCH,0CCH,078H ; D_0B
FACE 3C6666663C187E18    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 7F637F636367E6C0    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 020E3EFE3E0E0200    5518            DB      002H,00EH,03EH,0FEH,03EH,00EH,002H,000H ; D_11
FAFE 183C7E18187E3C18    5519            DB      018H,03CH,07EH,018H,018H,07EH,03CH,018H ; D_12
FB06 6666666666006600    5520            DB      066H,066H,066H,066H,066H,000H,066H,000H ; D_13
FB0E 7FDBDB7B1B1B1B00    5521            DB      07FH,0DBH,0DBH,07BH,01BH,01BH,01BH,000H ; D_14
FB16 3E63386C6C38CC78    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 003060FE60300000    5528            DB      000H,030H,060H,0FEH,060H,030H,000H,000H ; D_1B
FB4E 0000C0C0C0FE0000    5529            DB      000H,000H,0C0H,0C0H,0C0H,0FEH,000H,000H ; D_1C
FB56 002466FF66240000    5530            DB      000H,024H,066H,0FFH,066H,024H,000H,000H ; D_1D
FB5E 00183C7EFFFF0000    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,000H ; SP D_20
FB76 3078783030003000    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 307CC0780CF83000    5537            DB      030H,07CH,0C0H,078H,00CH,0F8H,030H,000H ; $ D_24
FB96 00C6CC183066C600    5538            DB      000H,0C6H,0CCH,018H,030H,066H,0C6H,000H ; PER CENT D_25
FB9E 386C3876DCCC7600    5539            DB      038H,06CH,038H,076H,0DCH,0CCH,076H,000H ; & D_26
FBA6 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
FBB6 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
FBC6 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
FBDE 0000000000303000    5547        DB    000H,000H,000H,000H,000H,030H,030H,000H ; . D_2E
FBE6 060C183060C08000    5548        DB    006H,00CH,018H,030H,060H,0C0H,080H,000H ; / D_2F
FBEE 7CC6CEDEF6E67C00    5549        DB    07CH,0C6H,0CEH,0DEH,0F6H,0E6H,07CH,000H ; 0 D_30
FBF6 3070303030303FC00   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 78CC0C380CCC7800    5552        DB    078H,0CCH,00CH,038H,00CH,0CCH,078H,000H ; 3 D_33
FC0E 1C3C6CCCFE0C1E00    5553        DB    01CH,03CH,06CH,0CCH,0FEH,00CH,01EH,000H ; 4 D_34
FC16 FCC0F80C0CCC7800    5554        DB    0FCH,0C0H,0F8H,00CH,00CH,0CCH,078H,000H ; 5 D_35
FC1E 3860C0F8CCCC7800    5555        DB    038H,060H,0C0H,0F8H,0CCH,0CCH,078H,000H ; 6 D_36
FC26 FCCC0C1830303000    5556        DB    0FCH,0CCH,00CH,018H,030H,030H,030H,000H ; 7 D_37
FC2E 78CCCC78CCCC7800    5557        DB    078H,0CCH,0CCH,078H,0CCH,0CCH,078H,000H ; 8 D_38
FC36 78CCCC7C0C187000    5558        DB    078H,0CCH,0CCH,07CH,00CH,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,030H,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,060H,000H ; > D_3E
FC66 78CC0C1830003000    5564        DB    078H,0CCH,00CH,018H,030H,000H,030H,000H ; ? D_3F
FC6E 7CC6DEDEDEC07800    5565        DB    07CH,0C6H,0DEH,0DEH,0DEH,0C0H,078H,000H ; @ D_40
FC76 3078CCCCFCCCCC00    5566        DB    030H,078H,0CCH,0CCH,0FCH,0CCH,0CCH,000H ; A D_41
FC7E FC66667C6666FC00    5567        DB    0FCH,066H,066H,07CH,066H,066H,0FCH,000H ; B D_42
FC86 3C66C0C0C0663C00    5568        DB    03CH,066H,0C0H,0C0H,0C0H,066H,03CH,000H ; C D_43
FC8E F86C6666666CF800    5569        DB    0F8H,06CH,066H,066H,066H,06CH,0F8H,000H ; D D_44
FC96 FE6268786862FE00    5570        DB    0FEH,062H,068H,078H,068H,062H,0FEH,000H ; E D_45
FC9E FE6268786860F000    5571        DB    0FEH,062H,068H,078H,068H,060H,0F0H,000H ; F D_46
FCA6 3C66C0C0CE663E00    5572        DB    03CH,066H,0C0H,0C0H,0CEH,066H,03EH,000H ; G D_47
FCAE CCCCCCFCCCCCCC00    5573        DB    0CCH,0CCH,0CCH,0FCH,0CCH,0CCH,0CCH,000H ; H D_48
FCB6 7830303030307800    5574        DB    078H,030H,030H,030H,030H,030H,078H,000H ; I D_49
FCBE 1E0C0C0CCCCC7800    5575        DB    01EH,00CH,00CH,00CH,0CCH,0CCH,078H,000H ; J D_4A
FCC6 E6666C786C66E600    5576        DB    0E6H,066H,06CH,078H,06CH,066H,0E6H,000H ; K D_4B
FCCE F06060606266FE00    5577        DB    0F0H,060H,060H,060H,062H,066H,0FEH,000H ; L D_4C
FCD6 C6EEFEFED6C6C600    5578        DB    0C6H,0EEH,0FEH,0FEH,0D6H,0C6H,0C6H,000H ; M D_4D
FCDE C6E6F6DECEC6C600    5579        DB    0C6H,0E6H,0F6H,0DEH,0CEH,0C6H,0C6H,000H ; N D_4E
FCE6 386CC6C6C66C3800    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 78CCCCCDC781C00     5582        DB    078H,0CCH,0CCH,0CCH,0DCH,078H,01CH,000H ; Q D_51
FCFE FC66667C6C66E600    5583        DB    0FCH,066H,066H,07CH,06CH,066H,0E6H,000H ; R D_52
FD06 78CCE0701CCC7800    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 CCCCCCCCCCCCFC00    5586        DB    0CCH,0CCH,0CCH,0CCH,0CCH,0CCH,0FCH,000H ; U D_55
FD1E CCCCCCCCCC783000    5587        DB    0CCH,0CCH,0CCH,0CCH,0CCH,078H,030H,000H ; V D_56
FD26 C6C6C6D6FEEEC600    5588        DB    0C6H,0C6H,0C6H,0D6H,0FEH,0EEH,0C6H,000H ; W D_57
FD2E C6C66C38386CC600    5589        DB    0C6H,0C6H,06CH,038H,038H,06CH,0C6H,000H ; X D_58
FD36 CCCCCC7830307800    5590        DB    0CCH,0CCH,0CCH,078H,030H,030H,078H,000H ; Y D_59
FD3E FEC68C183266FE00    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,006H,002H,000H ; BACKSLASH D_5C
FD56 7818181818187800    5594        DB    078H,018H,018H,018H,018H,018H,078H,000H ; ] D_5D
FD5E 10386CC600000000    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 0000780C7CCC7600    5598        DB    000H,000H,078H,00CH,07CH,0CCH,076H,000H ; LOWER CASE A D_61
FD7E E060607C6666DC00    5599        DB    0E0H,060H,060H,07CH,066H,066H,0DCH,000H ; L.C. B D_62
FD86 000078CCC0CC7800    5600        DB    000H,000H,078H,0CCH,0C0H,0CCH,078H,000H ; L.C. C D_63
FD8E 1C0C0C7CCCCC7600    5601        DB    01CH,00CH,00CH,07CH,0CCH,0CCH,076H,000H ; L.C. D D_64
FD96 000078CCFCC07800    5602        DB    000H,000H,078H,0CCH,0FCH,0C0H,078H,000H ; L.C. E D_65
FD9E 386C60F06060F000    5603        DB    038H,06CH,060H,0F0H,060H,060H,0F0H,000H ; L.C. F D_66
FDA6 000076CCCC7C0CF8    5604        DB    000H,000H,076H,0CCH,0CCH,07CH,00CH,0F8H ; L.C. G D_67
FDAE E0606C766666E600    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 0C000C0C0CCCCC78    5607        DB    00CH,000H,00CH,00CH,00CH,0CCH,0CCH,078H ; L.C. J D_6A
FDC6 E0606606C786CE600   5608        DB    0E0H,060H,066H,06CH,078H,06CH,0E6H,000H ; L.C. K D_6B
FDCE 7030303030307800    5609        DB    070H,030H,030H,030H,030H,030H,078H,000H ; L.C. L D_6C
FDD6 0000CCFEFED6C600    5610        DB    000H,000H,0CCH,0FEH,0FEH,0D6H,0C6H,000H ; L.C. M D_6D
FDDE 0000F8CCCCCCCC00    5611        DB    000H,000H,0F8H,0CCH,0CCH,0CCH,0CCH,000H ; L.C. N D_6E
FDE6 000078CCCCCC7800    5612        DB    000H,000H,078H,0CCH,0CCH,0CCH,078H,000H ; L.C. O D_6F
FDEE 0000DC66667C60F0    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,00CH,01EH ; L.C. Q D_71
FDFE 0000DC766660F000    5615        DB    000H,000H,0DCH,076H,066H,060H,0F0H,000H ; L.C. R D_72
FE06 00007CC0780CF800    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 0000CCCCCCCC7600    5618        DB    000H,000H,0CCH,0CCH,0CCH,0CCH,076H,000H ; L.C. U D_75
FE1E 0000CCCCCC783000    5619        DB    000H,000H,0CCH,0CCH,0CCH,078H,030H,000H ; L.C. V D_76
```

```
LOC OBJ                  LINE    SOURCE

FE26 0000C6D6FEFE6C00    5620            DB      000H,000H,0C6H,0D6H,0FEH,0FEH,06CH,000H ; L.C. W D_77
FE2E 0000C66C386CC600    5621            DB      000H,000H,0C6H,06CH,038H,06CH,0C6H,000H ; L.C. X D_78
FE36 0000CCCCCC7C0CF8    5622            DB      000H,000H,0CCH,0CCH,0CCH,07CH,00CH,0F8H ; L.C. Y D_79
FE3E 0000FC983064FC00    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 76DC000000000000    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     0FE6EH
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 C606700000          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 C606700000          5673            MOV     TIMER_OFL,0             ; RESET OVERFLOW
FE9F EBDA                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 DECREMENTS 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 FF066C00           5702            INC     TIMER_LOW               ; INCREMENT TIME
FEB0 7504               5703            JNZ     T4                      ; TEST_DAY
FEB2 FF066E00           5704            INC     TIMER_HIGH              ; INCREMENT HIGH WORD OF TIME
FEB6                    5705    T4:                                     ; TEST_DAY
FEB6 833E6E0018         5706            CMP     TIMER_HIGH,018H         ; 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 C606700001         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,EOI
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
FEF3                    5745            ORG     0FEF3H
FEF3                    5746    VECTOR_TABLE    LABEL   WORD             ; VECTOR TABLE FOR MOVE TO INTERRUPTS
FEF3 A5FE               5747            DW      OFFSET TIMER_INT        ; INTERRUPT 8
FEF5 87E9               5748            DW      OFFSET  KB_INT          ; INTERRUPT 9
FEF7 23FF               5749            DW      OFFSET  D11             ; INTERRUPT A
FEF9 23FF               5750            DW      OFFSET  D11             ; INTERRUPT B
FEFB 23FF               5751            DW      OFFSET  D11             ; INTERRUPT C
FEFD 23FF               5752            DW      OFFSET  D11             ; INTERRUPT D
FEFF 57EF               5753            DW      OFFSET DISK_INT         ; INTERRUPT E
FF01 23FF               5754            DW      OFFSET  D11             ; INTERRUPT F
FF03 65F0               5755            DW      OFFSET VIDEO_IO         ; INTERRUPT 10H
FF05 4DF8               5756            DW      OFFSET EQUIPMENT        ; INTERRUPT 11H
FF07 41F8               5757            DW      OFFSET MEMORY_SIZE_DET  ; INTERRUPT 12H
FF09 59EC               5758            DW      OFFSET DISKETTE_IO      ; INTERRUPT 13H
FF0B 39E7               5759            DW      OFFSET RS232_IO         ; INTERRUPT 14H
FF0D 59F8               5760            DW      CASSETTE_IO             ; INTERRUPT 15H(FORMER CASSETTE IO)
FF0F 2EE8               5761            DW      OFFSET KEYBOARD_IO      ; INTERRUPT 16H
FF11 D2EF               5762            DW      OFFSET PRINTER_IO       ; INTERRUPT 17H
                        5763
FF13 0000               5764            DW      00000H                  ; INTERRUPT 18H
                        5765            DW      0F600H                  ;  MUST BE INSERTED INTO TABLE LATER
                        5766
FF15 F2E6               5767            DW      OFFSET BOOT_STRAP       ; INTERRUPT 19H
FF17 6EFE               5768            DW      TIME_OF_DAY             ; INTERRUPT 1AH -- TIME OF DAY
FF19 4BFF               5769            DW      DUMMY_RETURN            ; INTERRUPT 1BH -- KEYBOARD BREAK ADDR
FF1B 4BFF               5770            DW      DUMMY_RETURN            ; INTERRUPT 1C -- TIMER BREAK ADDR
FF1D A4F0               5771            DW      VIDEO_PARMS             ; INTERRUPT 1D -- VIDEO PARAMETERS
FF1F C7EF               5772            DW      OFFSET DISK_BASE        ; INTERRUPT 1E -- DISK PARMS
FF21 0000               5773            DW      0                       ; INTERRUPT 1F -- POINTER TO VIDEO EXT
```

```
LOC OBJ                 LINE    SOURCE

                        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 INTERUPTS THAT WAS     :
                        5783    ;       EXECUTED ACCIDENTLY.                            :
                        5784    ;--------------------------------------------------------
FF23                    5785    D11     PROC    NEAR
                        5786            ASSUME  DS:DATA
FF23 1E                 5787            PUSH    DS
FF24 52                 5788            PUSH    DX
FF25 50                 5789            PUSH    AX                      ; SAVE REG AX CONTENTS
FF26 E830FB             5790            CALL    DDS                     FA59
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     HW_INT
FF36 B4FF               5798            MOV     AH,0FFH
FF38 EB0A               5799            JMP     SHORT SET_INTR_FLAG     ; SET FLAG TO FF IF NON-HDWARE
FF3A                    5800    HW_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 88266B00           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     0FF53H
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:XXDATA
FF54                    5837            ORG     0FF54H
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,XXDATA               ; 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 33D2               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 0AC0               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 33D2               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            CMP     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            CMP     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 EB0A               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 33D2               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,12Q                  ; LF
```

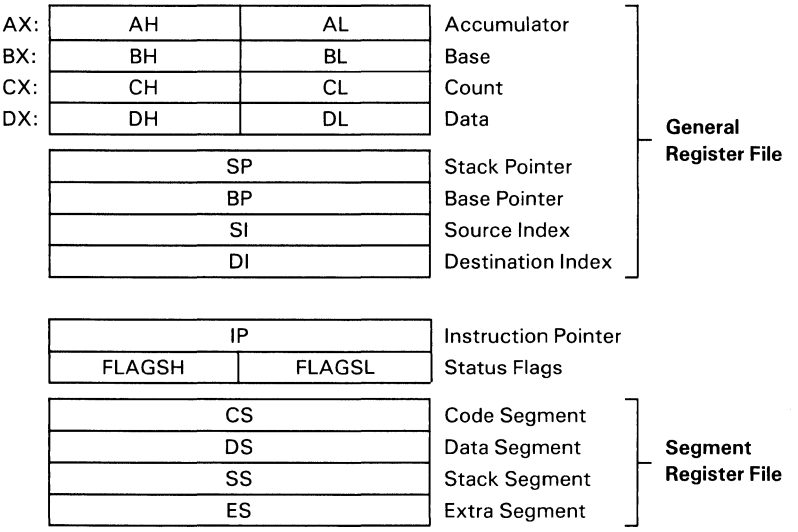
```
LOC OBJ                  LINE    SOURCE

FFD1 CD17                5928            INT     17H                     ; SEND THE LINE FEED
FFD3 32E4                5929            XOR     AH,AH                   ; NOW FOR THE CR
FFD5 B00D                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,DH                   ;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 OFFFFH
                         5957
                         5958    ;----- POWER ON RESET
                         5959
0000 EA5BE000F0          5960            JMP     RESET
                         5961
0005 31312F30382F38      5962            DB      '11/08/82'              ; RELEASE MARKER
     32
----                     5963    VECTOR  ENDS
                         5964            END
```

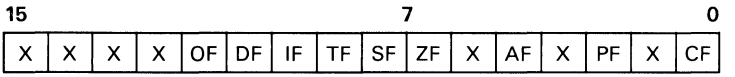
SECTION 6. INSTRUCTION SET

Contents

8088
Register Model



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



x = Don't Care

AF: Auxiliary Carry - BCD
CF: Carry Flag
PF: Parity Flag
SF: Sign Flag
ZF: Zero Flag

8080 Flags

DF: Direction Flag (Strings)
IF: Interrupt Enable Flag
OF: Overflow Flag (CF ⊕ SF)
TF: Trap - Single Step Flag

8088 Flags

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

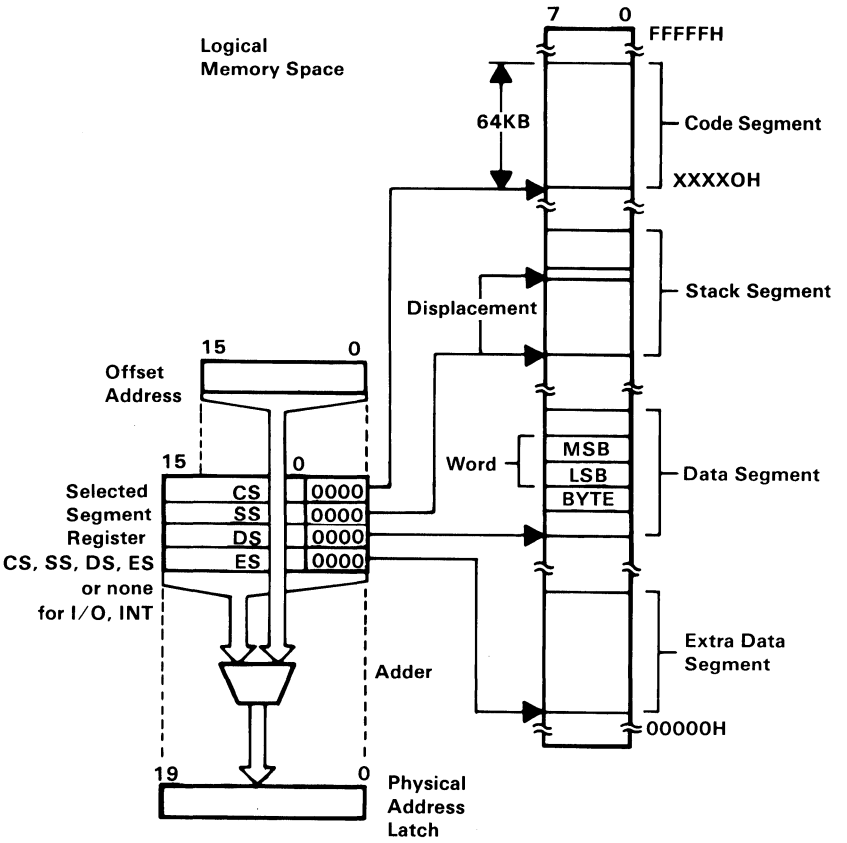
mod	Displacement
00	DISP = 0*, disp-low and disp-high are absent
01	DISP = disp-low sign-extended to 16-bits, disp-high is absent
10	DISP = disp-high: disp-low
11	r/m is treated as a "reg" field

MF = Memory format
00 — 32-bit Real
01 — 32-bit Integer
10 — 64-bit Real
11 — 64-bit Integer

r/m	Operand Address
000	(BX) + (SI) + DISP
001	(BX) + (DI) + DISP
010	(BP) + (SI) + DISP
011	(BP) + (DI) + DISP
100	(SI) + DISP
101	(DI) + DISP
110	(BP) + DISP*
111	(BX) + DISP

DISP follows 2nd byte of instruction (before data if required).
*except if mod = 00 and r/m = 110 then EA = disp-high: disp-low.

Memory Segmentation Model



Segment Override Prefix

0	0	1	reg	1	1	0

Use of Segment Override

Operand Register	Default	With Override Prefix
IP (Code Address)	CS	Never
SP (Stack Address)	SS	Never
BP (Stack Address or Stack Marker)	SS	BP + DS or ES, or CS
SI or DI (not including strings)	DS	ES, SS, or CS
SI (Implicit Source Address for Strings)	DS	ES, SS, or CS
DI (Implicit Destination Address for Strings)	ES	Never

Data Transfer

MOV = Move

Register/memory to/from register

1 0 0 0 1 0 d w	mod reg r/m

Immediate to register/memory

1 1 0 0 0 1 1 w	mod 0 0 0 r/m	data	data if w = 1

Immediate to register

1 0 1 1 w reg	data	data if w = 1

Memory to accumulator

1 0 1 0 0 0 0 w	addr-low	addr-high

Accumulator to memory

1 0 1 0 0 0 1 w	addr-low	addr-high

Register/memory to segment register

1 0 0 0 1 1 1 0	mod 0 reg r/m

Segment register to register/memory

1 0 0 0 1 1 0 0	mod 0 reg r/m

PUSH = Push

Register/memory

1 1 1 1 1 1 1 1	mod 1 1 0 r/m

Register

0 1 0 1 0 reg

Segment register

0 0 0 reg 1 1 0

Pop = Pop

Register/memory

1 0 0 0 1 1 1 1	mod 0 0 0 r/m

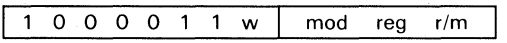
Register

0 1 0 1 1 reg

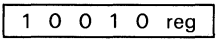
Segment register

0 0 0 reg 1 1 1

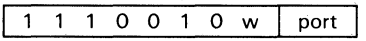
XCHG = Exchange
Register/memory with register



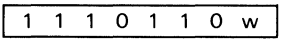
Register with accumulator



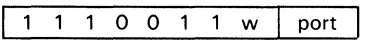
IN = Input to AL/AX from
Fixed port



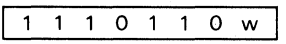
Variable port (DX)



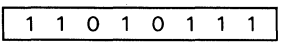
OUT = Output from AL/AX to
Fixed port



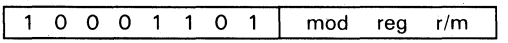
Variable port (DX)



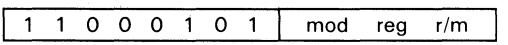
XLAT = Translate byte to AL



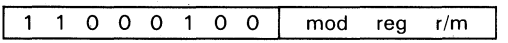
LEA = Load EA to register



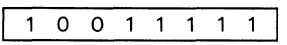
LDS = Load pointer to DS



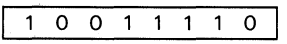
LES = Load pointer to ES



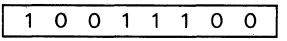
LAHF = Load AH with flags



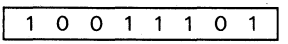
SAHF = Store AH into flags



PUSHF = Push flags

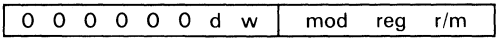


POPF = Pop flags

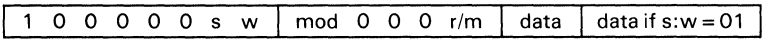


Arithmetic

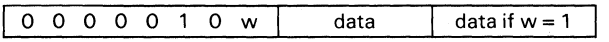
ADD = Add
Register/memory with register to either



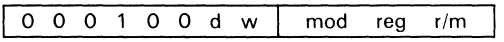
Immediate to register/memory



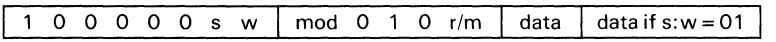
Immediate to accumulator



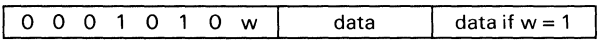
ADC = Add with carry
Register/memory with register to either



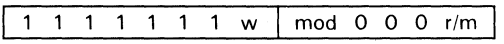
Immediate to register/memory



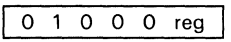
Immediate to accumulator



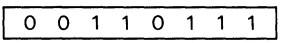
INC = Increment
Register/Memory



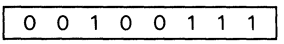
Register



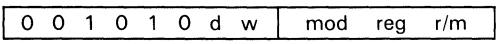
AAA = ASCII adjust for add



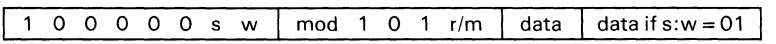
DAA = Decimal adjust for add



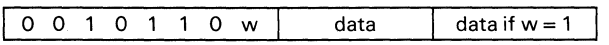
SUB = Subtract
Register/memory and register to either



Immediate from register/memory



Immediate from accumulator



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 subtact

0 0 1 1 1 1 1 1

DAS = Decimal adjust for subtact

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

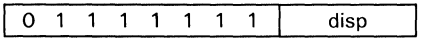
JNE/JNZ = Jump on not equal/not zero

0	1	1	1	0	1	0	1	disp

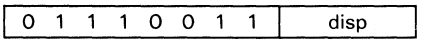
JNL/JGE = Jump on not less/greater or equal

0	1	1	1	1	1	0	1	disp

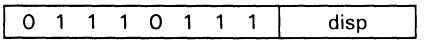
JNLE/JG = Jump on not less or equal/greater



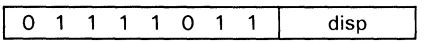
JNB/JAE = Jump on not below/above or equal



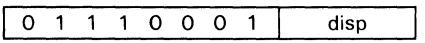
JNBE/JA = Jump on not below or equal/above



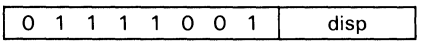
JNP/JPO = Jump on not parity/parity odd



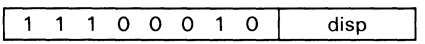
JNO = Jump on not overflow



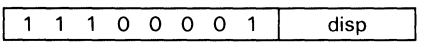
JNS = Jump on not sign



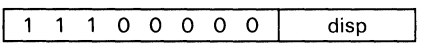
LOOP = Loop CX times



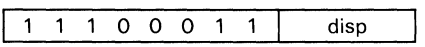
LOOPZ/LOOPE = Loop while zero/equal



LOOPNZ/LOOPNE = Loop while not zero/not equal



JCXZ = Jump on CX zero

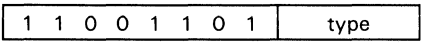


8088 Conditional Transfer Operations

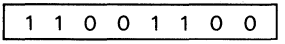
Instruction	Condition	Interpretation
JE or JZ	ZF = 1	"equal" or "zero"
JL or JNGE	(SF xor OF) = 1	"less" or "not greater or equal"
JLE or JNG	((SF xor OF) or ZF) = 1	"less or equal" or "not greater"
JB or JNAE or JC	CF = 1	"below" or "not above or equal"
JBE or JNA	(CF or ZF) = 1	"below or equal" or "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 less 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.

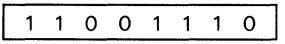
INT = Interrupt
Type specified



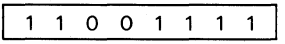
Type 3



INTO = Interrupt on overflow



IRET = Interrupt return



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 signed 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

Data Transfer

FLD = Load
Integer/Real Memory to ST(0)

Escape MF 1	mod 0 0 0 r/m	disp-low	disp-high

Long Integer Memory to ST(0)

Escape 1 1 1	mod 1 0 1 r/m	disp-low	disp-high

Temporary Real Memory to ST(0)

Escape 0 1 1	mod 1 0 1 r/m	disp-low	disp-high

BCD Memory to ST(0)

Escape 1 1 1	mod 1 0 0 r/m	disp-low	disp-high

ST(i) to ST(0)

Escape 0 0 1	1 1 0 0 0 ST(i)

FST = Store
ST(0) to Integer/Real Memory

Escape MF 1	mod 0 1 0 r/m	disp-low	disp-high

ST(0) to ST(i)

Escape 1 0 1	1 1 0 1 0 ST(i)

FSTP = STORE AND POP
ST(0) to Integer/Real Memory

Escape	MF	1	mod	0	1	1	r/m	disp-low	disp-high

ST(0) to Long Integer Memory

Escape	1	1	1	mod	1	1	1	r/m	disp-low	disp-high

ST(0) to Temporary Real Memory

Escape	0	1	1	mod	1	1	1	r/m	disp-low	disp-high

ST(0) to BCD Memory

Escape	1	1	1	mod	1	1	0	r/m	disp-low	disp-high

ST(0) to ST(i)

Escape	1	0	1	1	1	0	1	1	ST(i)

FXCH = Exchange ST(i) and ST(0)

Escape	0	0	1	1	1	0	0	1	ST(i)

Comparison

FCOM = Compare
Integer/Real Memory to ST(0)

Escape MF 0	mod 0 1 0 r/m	disp-low	disp-high

ST(i) to ST(0)

Escape 0 0 0	1 1 0 1 0 ST(i)

FCOMP = Compare and Pop
Integer/Real Memory to ST(0)

Escape MF 0	mod 0 1 1 r/m	disp-low	disp-high

ST(i) to ST(0)

Escape 0 0 0	1 1 0 1 1 ST(i)

FCOMPP = Compare ST(1) to ST(0) and Pop twice

Escape 1 1 0	1 1 0 1 1 0 0 1

FTST = Test ST(0)

Escape 0 0 1	1 1 1 0 0 1 0 0

FXAM = Examine ST(0)

Escape 0 0 1	1 1 1 0 0 1 0 1

Arithmetic

FADD = Addition
Integer/Real Memory with ST(0)

Escape MF 0	mod 0 0 0 r/m	disp-low	disp-high

ST(i) to ST(0)

Escape d P 0	1 1 0 0 0 ST(i)

FSUB = Subtraction
Integer/Real Memory with ST(0)

Escape MF 0	mod 1 0 R r/m	disp-low	disp-high

ST(i) to ST(0)

Escape d P 0	1 1 1 0 R r/m

FMUL = Multiplication
Integer/Real Memory to ST(0)

Escape	MF	0	mod	0	0	1	r/m	disp-low	disp-high

ST(i) and ST(0)

Escape	d	P	0	1	1	0	0	1	r/m

FDIV = Division
Integer/Real Memory with ST(0)

Escape	MF	0	mod	1	1	R	r/m	disp-low	disp-high

ST(i) and ST(0)

Escape	d	P	0	1	1	0	0	1	r/m

FSQRT = Square Root of ST(0)

Escape	0	0	1	1	1	1	1	1	0	1	0

FSCALE = Scale ST(0) by ST(1)

Escape	0	0	1	1	1	1	1	1	1	0	1

FPREM = Partial Remainder of ST(0) ÷ ST(1)

Escape	0	0	1	1	1	1	1	1	0	0	0

FRNDINT = Round ST(0) to Integer

Escape	0	0	1	1	1	1	1	1	1	0	0

FXTRACT = Extract Components of ST(0)

Escape	0	0	1	1	1	1	1	0	1	0	0

FABS = Absolute Value of ST(0)

Escape	0	0	1	1	1	1	0	0	0	0	1

FCHS = Change Sign of ST(0)

Escape	0	0	1	1	1	1	0	0	0	0	0

Transcendental

FPTAN = Partial Tangent of ST(0)

Escape	0	0	1	1	1	1	1	0	0	1	0

FPATAN = Partial Arctangent of ST(0) ÷ ST(1)

Escape	0	0	1	1	1	1	1	0	0	1	1

F2XM1 = $2^{ST(0)}-1$

Escape	0	0	1	1	1	1	1	0	0	0	0

FYL2X = $ST(1) \cdot LOG_2[ST(0)]$

Escape	0	0	1	1	1	1	1	0	0	0	1

FYL2XP1 = $ST(1) \cdot LOG_2[ST(0) + 1]$

Escape	0	0	1	1	1	1	1	1	0	0	1

Constants

FLDZ = Load + 0.0 into ST(0)

Escape	0	0	1	1	1	1	0	1	1	1	0

FLD1 = Load + 1.0 into ST(0)

Escape	0	0	1	1	1	1	0	1	0	0	0

FLDPI = Load π into ST(0)

Escape	0	0	1	1	1	1	0	1	0	1	1

FLDL2T = Load $\log_2 10$ into ST(0)

Escape	0	0	1	1	1	1	0	1	0	0	1

FLDL2E = Load $\log_2 e$ into ST(0)

Escape	0	0	1	1	1	1	0	1	0	1	0

FLDLG2 = Load $\log_{10} 2$ into ST(0)

Escape	0	0	1	1	1	1	0	1	1	0	0

FLDLN2 = Load $\log_e 2$ into ST(0)

Escape	0	0	1	1	1	1	0	1	1	0	1

FINIT = Initialize NDP

Escape	0	1	1	1	1	1	0	0	0	1	1

FENI = Enable Interrupts

Escape	0	1	1	1	1	1	0	0	0	0	0

FDISI = Disable Interrupts

Escape	0	1	1	1	1	1	0	0	0	0	1

FLDCW = Load Control Word

Escape	0	0	1	mod	1	0	1	r/m	disp-low	disp-high

FSTCW = Store Control Word

Escape	0	0	1	mod	1	1	1	r/m	disp-low	disp-high

FSTSW = Store Status Word

Escape	1	0	1	mod	1	1	1	r/m	disp-low	disp-high

FCLEX = Clear Exceptions

Escape	0	1	1	1	1	1	0	0	0	1	0

FSTENV = Store Environment

Escape	0	0	1	mod	1	1	0	r/m	disp-low	disp-high

FLDENV = Load Enviroment

Escape 0 0 1	mod 1 0 0 r/m	disp-low	disp-high

FSAVE = Save State

Escape 1 0 1	mod 1 1 0 r/m	disp-low	disp-high

FRSTOR = Restore State

Escape 1 0 1	mod 1 0 0 r/m	disp-low	disp-high

FINCSTP = Increment Stack Pointer

Escape 0 0 1	1 1 1 1 0 1 1 1

FDECSTP = Decrement Stack Pointer

Escape 0 0 1	1 1 1 1 0 1 1 0

FFREE = Free ST(i)

Escape 0 0 1	1 1 0 0 0 ST(i)

FNOP = No Operation

Escape 0 0 1	1 1 0 1 0 0 0 0

FWAIT = CPU Wait for NDP

1 0 0 1 1 0 1 1

Footnotes:
ST(0) = Current Stack top
ST(i) = i^{th} register below stack top

d = Destination
0 — Destination is ST(0)
1 — Destination is ST(i)

P = POP
0 — No pop
1 — Pop ST(0)

R = Reverse
0 — Destination (op) Source
1 — Source (op) Destination

For **FSQRT:** $-0 \leq ST(0) \leq +\infty$
For **FSCALE:** $-2^{15} \leq ST(1) < +2^{15}$ and ST(1) integer
For **F2XM1:** $0 \leq ST(0) \leq 2^{-1}$
For **FYL2X:** $0 < ST(0) < \infty$
$-\infty < ST(1) < +\infty$
For **FYL2XP1:** $0 < |ST(0)| < (2 - \sqrt{2})/2$
$-\infty < ST(1) < \infty$
For **FPTAN:** $0 \leq ST(0) < \pi/4$
For **FPATAN:** $0 \leq ST(0) < ST(1) < +\infty$

8088 Instruction Set Matrix

HI \ LO	0	1	2	3	4	5	6	7
0	ADD b,f,r/m	ADD w,f,r/m	ADD b,t,r/m	ADD w,t,r/m	ADD b,ia	ADD w,ia	PUSH ES	POP ES
1	ADC b,f,r/m	ADC w,f,r/m	ADC b,t,r/m	ADC w,t,r/m	ADC b,i	ADC w,i	PUSH SS	POP SS
2	AND b,f,r/m	AND w,f,r/m	AND b,t,r/m	AND w,t,r/m	AND b,i	AND w,i	SEG =ES	DAA
3	XOR b,f,r/m	XOR w,f,r/m	XOR b,t,r/m	XOR w,t,r/m	XOR b,i	XOR w,i	SEG =SS	AAA
4	INC AX	INC CX	INC DX	INC BX	INC SP	INC BP	INC SI	INC DI
5	PUSH AX	PUSH CX	PUSH DX	PUSH BX	PUSH SP	PUSH BP	PUSH SI	PUSH D1
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 AL	MOV AL m	MOV AL m	MOVS b	MOVS w	CMPS b	CMPS w
B	MOV i AL	MOV i CL	MOV i DL	MOV i BL	MOV i AH	MOV i CH	MOV i DH	MOV i BH
C			RET (i + SP)	RET	LES	LDS	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

HI \ LO	8	9	A	B	C	D	E	F
0	OR b,f,r/m	w,f,r/m	OR b,t,r/m	OR w,t,r/m	OR b,i	OR w,i	PUSH CS	
1	SBB b,f,r/m	SBB w,f,r/m	SBB b,t,r/m	SBB w,t,r/m	SBB b,i	SBB w,i	PUSH DS	POP DS
2	SUB b,f,r/m	SUB w,f,r/m	SUB b,t,r/m	SUB w,t,r/m	SUB b,i	SUB w,i	SEG = CS	DAS
3	CMP b,f,r/m	CMP w,f,r/m	CMP b,t,r/m	CMP w,t,r/m	CMP b,i	CMP w,i	SEG = CS	AAS
4	DEC AX	DEC CX	DEC DX	DEC BX	DEC SP	DEC BP	DEC SI	DEC DI
5	POP AX	POP CX	POP DX	POP BX	POP SP	POP BP	POP SI	POP DI
6								
7	JS	JNS	JP/ JPE	JNP/ JPO	JL/ JNGE	JNL/ JGE	JLE/ JNG	JNLE/ JG
8	MOV b,f,r/m	MOV w,f,r/m	MOV b,t,r/m	MOV w,t,r/m	MOV sr,t,r/m	LEA	MOV sr,f,r/m	POP r/m
9	CBW	CWD	CALL l,d	WAIT	PUSHF	POPF	SAHF	LAHF
A	TEST b,i	TEST w,i	STOS b	STOS w	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

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	—

Notes:

SECTION 7. CHARACTERS, KEYSTROKES, AND COLORS

Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
00	0	Blank (Null)	Ctrl 2		Black	Black	Non-Display
01	1	☺	Ctrl A		Black	Blue	Underline
02	2	☻	Ctrl B		Black	Green	Normal
03	3	♥	Ctrl C		Black	Cyan	Normal
04	4	♦	Ctrl D		Black	Red	Normal
05	5	♣	Ctrl E		Black	Magenta	Normal
06	6	♠	Ctrl F		Black	Brown	Normal
07	7	•	Ctrl G		Black	Light Grey	Normal
08	8	◘	Ctrl H, Backspace, Shift Backspace		Black	Dark Grey	Non-Display
09	9	○	Ctrl I		Black	Light Blue	High Intensity Underline
0A	10	◙	Ctrl J, 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

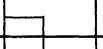
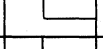
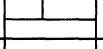
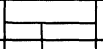
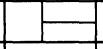
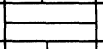
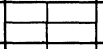
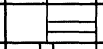
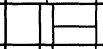
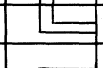
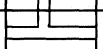
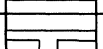
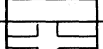
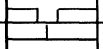
Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
2F	47	/	/		Green	White	High Intensity
30	48	0	0	Note 3	Cyan	Black	Normal
31	49	1	1	Note 3	Cyan	Blue	Underline
32	50	2	2	Note 3	Cyan	Green	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

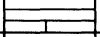
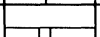
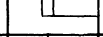
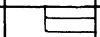
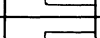
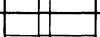
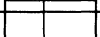
Value		As Characters			As Text Attributes		
					Color/Graphics Monitor Adapter		IBM Monochrome Display Adapter
Hex	Dec	Symbol	Keystrokes	Modes	Background	Foreground	
4B	75	K	K	Note 4	Red	Light Cyan	High Intensity
4C	76	L	L	Note 4	Red	Light Red	High Intensity
4D	77	M	M	Note 4	Red	Light Magenta	High Intensity
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

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	t	t	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	Ä	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	Ö	Alt 153	Note 6	Blue	Light Blue	High Intensity Underline
9A	154	Ü	Alt 154	Note 6	Blue	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	
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	ƒ	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	a	Alt 166	Note 6	Green	Brown	Normal
A7	167	o	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	¡	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

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	ϵ	Alt 238	Note 6	Yellow	Yellow	High Intensity
EF	239	∩	Alt 239	Note 6	Yellow	White	High Intensity
F0	240	≡	Alt 240	Note 6	White	Black	Reverse Video
F1	241	±	Alt 241	Note 6	White	Blue	Underline
F2	242	≥	Alt 242	Note 6	White	Green	Normal
F3	243	≤	Alt 243	Note 6	White	Cyan	Normal
F4	244	⌠	Alt 244	Note 6	White	Red	Normal
F5	245	⌡	Alt 245	Note 6	White	Magenta	Normal
F6	246	÷	Alt 246	Note 6	White	Brown	Normal
F7	247	≈	Alt 247	Note 6	White	Light Grey	Normal
F8	248	○	Alt 248	Note 6	White	Dark Grey	Reverse Video
F9	249	●	Alt 249	Note 6	White	Light Blue	High Intensity Underline
FA	250	·	Alt 250	Note 6	White	Light Green	High Intensity
FB	251	√	Alt 251	Note 6	White	Light Cyan	High Intensity
FC	252	η	Alt 252	Note 6	White	Light Red	High Intensity
FD	253	2	Alt 253	Note 6	White	Light Magenta	High Intensity
FE	254	■	Alt 254	Note 6	White	Yellow	High Intensity
FF	255	BLANK	Alt 255	Note 6	White	White	High Intensity

NOTE 1 Asterisk (*) can easily be keyed using two methods: 1) hit the Prt Sc * key or 2) in shift mode hit the 8 * 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 Del . key.

NOTE 3 Numeric characters (0—9) can easily be keyed using two methods: 1) hit the numeric keys on the top row of the typewriter portion of the keyboard or 2) in shift or Num Lock mode hit the numeric keys in the 10—key pad portion of the keyboard.

NOTE 4 Upper case alphabetic characters (A—Z) can easily be keyed in two modes: 1) in shift mode the appropriate alphabetic key or 2) in Caps Lock mode hit the appropriate alphabetic key.

NOTE 5 Lower case alphabetic characters (a—z) can easily be keyed in two modes: 1) in "normal" mode hit the appropriate key or 2) in Caps Lock combined with shift mode hit the appropriate alphabetic key.

NOTE 6 The 3 digits after the Alt key must be typed from the numeric key pad (keys 71—73, 75—77, 79—82). Character codes 000 through 255 can be entered in this fashion. (With Caps Lock activated, Character codes 97 through 122 will display upper case rather than lower case alphabetic characters.)

Character Set (00-7F) Quick Reference

DECIMAL VALUE ➡		0	16	32	48	64	80	96	112
⬇	HEXA DECIMAL VALUE	0	1	2	3	4	5	6	7
0	0	BLANK (NULL)	►	BLANK (SPACE)	0	@	P	‘	p
1	1	☺	◄	!	1	A	Q	a	q
2	2	☻	↕	"	2	B	R	b	r
3	3	♥	!!	#	3	C	S	c	s
4	4	♦	¶	$	4	D	T	d	t
5	5	♣	§	%	5	E	U	e	u
6	6	♠	▬	&	6	F	V	f	v
7	7	•	↨	’	7	G	W	g	w
8	8	◘	↑	(	8	H	X	h	x
9	9	○	↓	)	9	I	Y	i	y
10	A	◙	→	*	:	J	Z	j	z
11	B	♂	←	+	;	K	[	k	{
12	C	♀	∟	,	<	L	\	l	¦
13	D	♪	↔	—	=	M	]	m	}
14	E	♫	▲	.	>	N	^	n	~
15	F	☼	▼	/	?	O	_	o	△

Character Set (80-FF) Quick Reference

DECIMAL VALUE ➡	128	144	160	176	192	208	224	240
⬇ HEXA DECIMAL VALUE	8	9	A	B	C	D	E	F
0	0	Ç	É	á	▒	⊥	∝	≡
1	1	ü	æ	í	▒	⊥	β	±
2	2	é	Æ	ó	▒	⊥	Γ	≥
3	3	â	ô	ú	▒	⊥	π	≤
4	4	ä	ö	ñ	▒	⊥	Σ	⌠
5	5	à	ò	Ñ	▒	⊥	σ	⌡
6	6	å	û	ª	▒	⊥	μ	÷
7	7	ç	ù	º	▒	⊥	τ	≈
8	8	ê	ÿ	¿	▒	⊥	Φ	°
9	9	ë	Ö	⌐	▒	⊥	Θ	•
10	A	è	Ü	¬	▒	⊥	Ω	•
11	B	ï	¢	½	▒	▒	δ	√
12	C	î	£	¼	▒	▒	∞	n
13	D	ì	¥	¡	▒	▒	φ	2
14	E	Ä	₧	«	▒	▒	∈	■
15	F	Å	ƒ	»	▒	▒	∩	BLANK 'FF'

Notes:

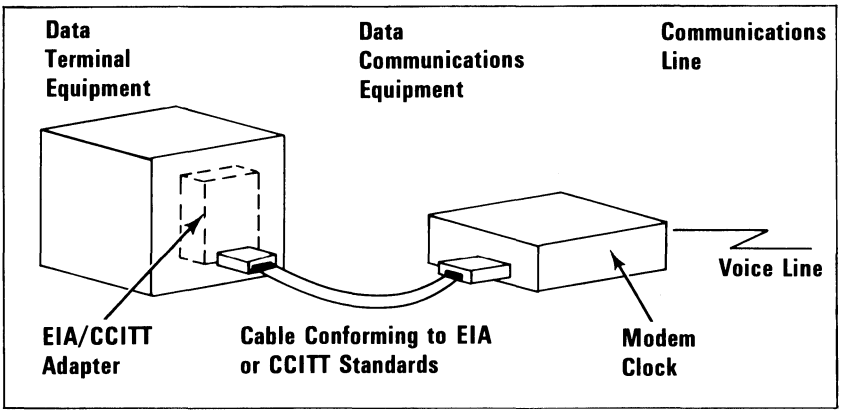
SECTION 8. COMMUNICATIONS

Contents

Description

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:



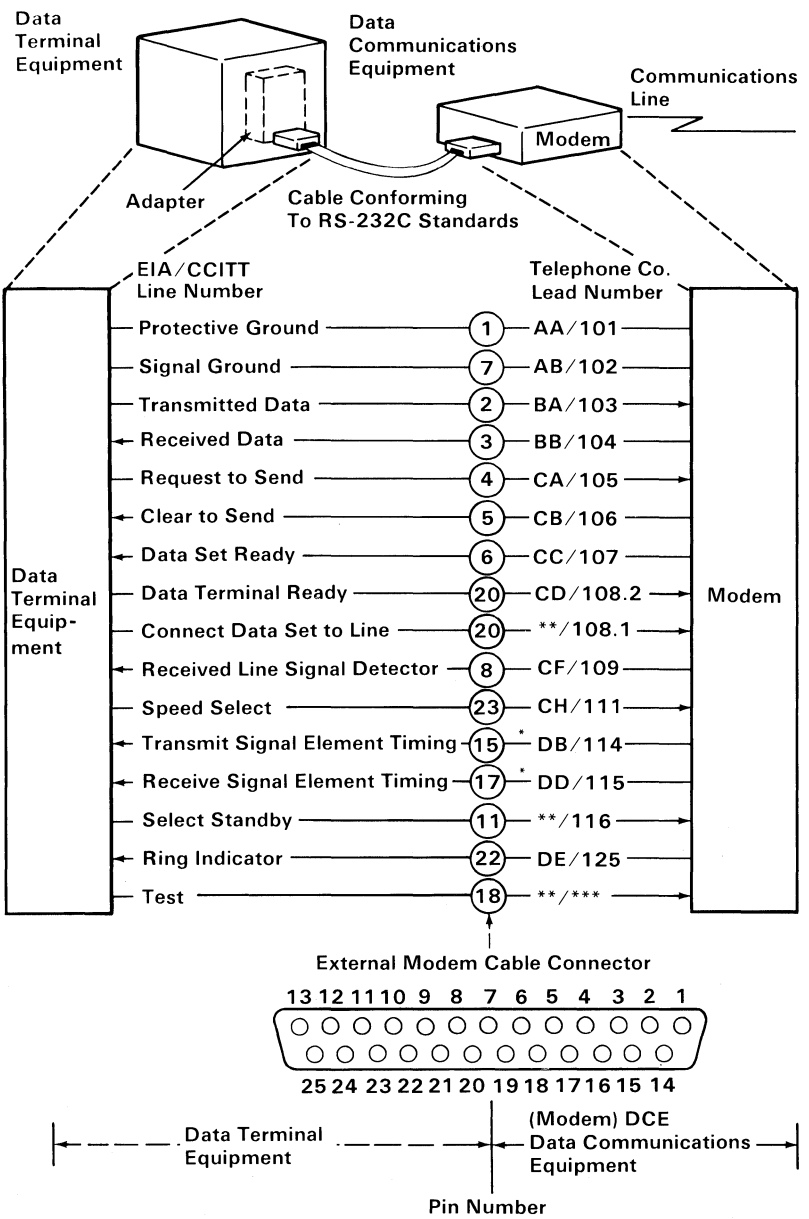
The EIA/CCITT adapter allows data terminal equipment to be connected to data communications equipment using EIA or CCITT standardized connections. An external modem is shown in this example; however, other types of data communications equipment can also be connected to data terminal equipment using EIA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

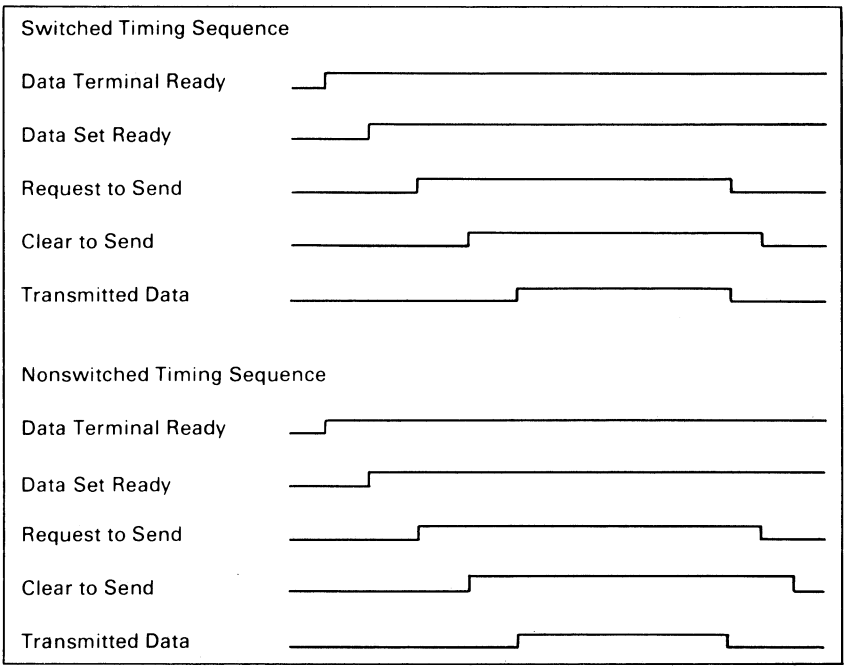
The following is an illustration of data terminal equipment connected to an external modem using connections defined by the RS-232C interface standard:



*Not used when business machine clocking is used.
**Not standardized by EIA (Electronic Industries Association).
***Not standardized by CCITT

Establishing a Communications Link

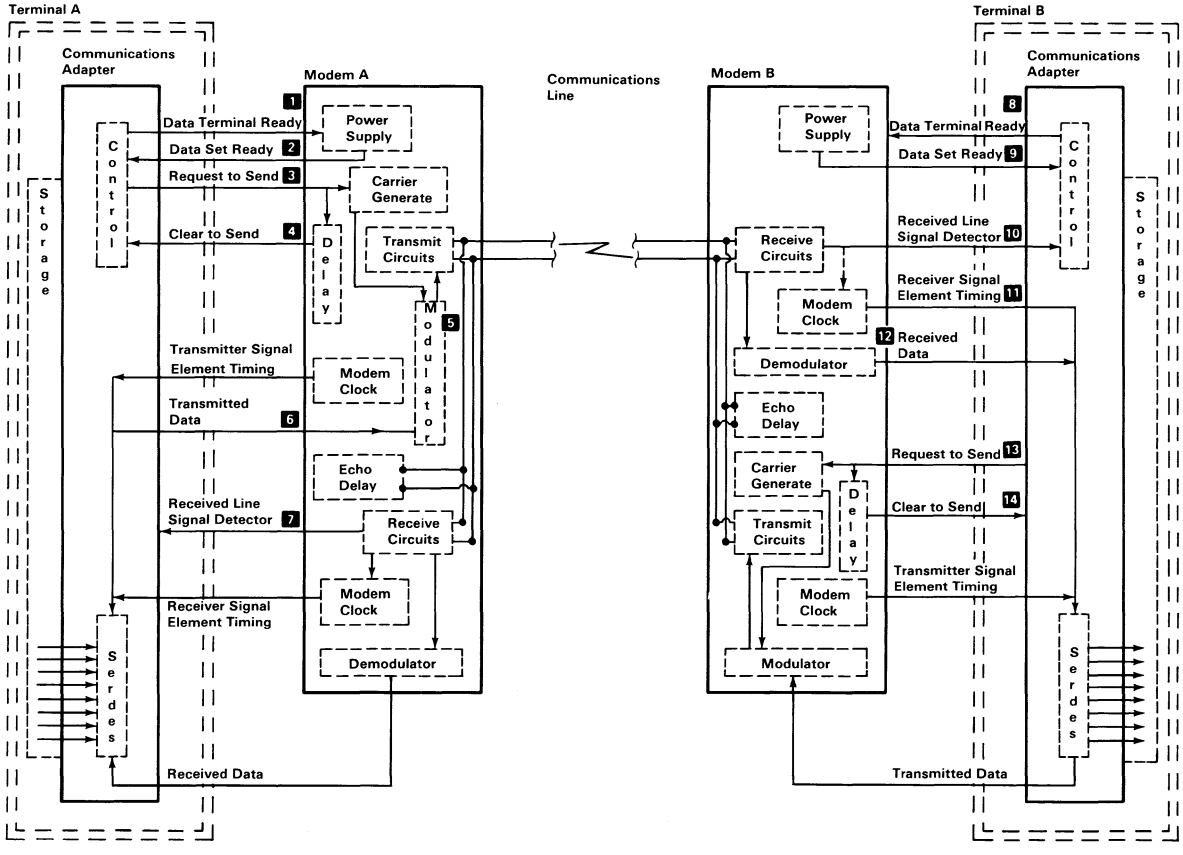
The following bar graphs represent normal timing sequences of operation during the establishment of communications for both switched (dial-up) and nonswitched (direct line) networks.



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.

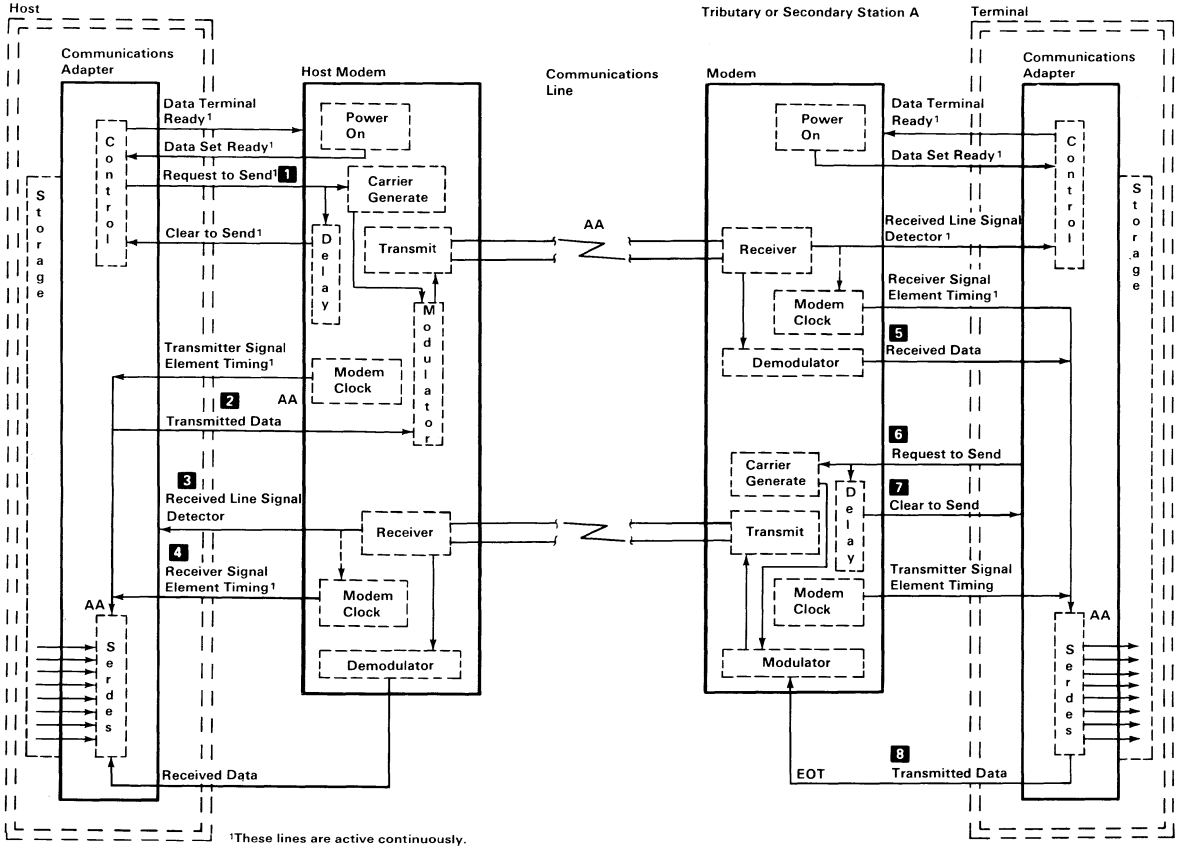
Establishing a Link on a Nonswitched Point-to-Point Line

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, 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 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 ready to receive the response from termianl B. Remember, the response was not transmitted until after the request-to-send to clear-to-send delay at modem B (step 14).


Terminal A
Communications Adapter
Modem A
Communications Line
Modem B
Terminal B
Communications Adapter
Storage
Control
Data Terminal Ready 1
Data Set Ready 2
Request to Send 3
Clear to Send 4
Power Supply
Carrier Generate
Delay
Transmit Circuits
Modulator 5
Transmitter Signal Element Timing
Modem Clock
Transmitted Data 6
Echo Delay
Received Line Signal Detector 7
Receive Circuits
Receiver Signal Element Timing
Modem Clock
Demodulator
Serdes
Received Data
Power Supply
Data Terminal Ready 8
Data Set Ready 9
Control
Storage
Receive Circuits
Received Line Signal Detector 10
Receiver Signal Element Timing 11
Modem Clock
12 Received Data
Demodulator
Echo Delay
Request to Send 13
Carrier Generate
Delay
Clear to Send 14
Transmit Circuits
Transmitter Signal Element Timing
Modem Clock
Modulator
Serdes
Transmitted Data

Establishing a Link on a Nonswitched Multipoint Line

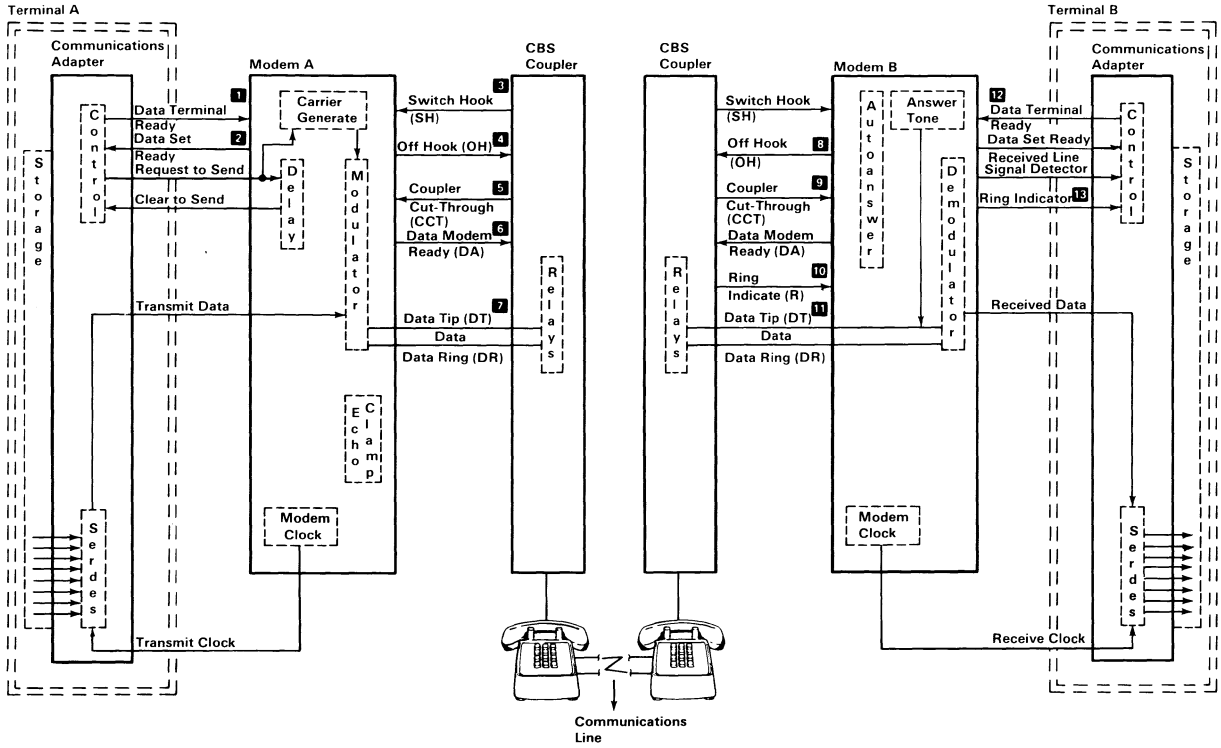
1. The control station serializes the address for the tributary or secondary station (AA) and sends its address to the modem on the 'transmitted data' line 2.
2. Since the 'request to send' line and, therefore, the modem carrier, is active continuously 1, the modem immediately modulates the carrier with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


Host
Communications Adapter
Host Modem
Communications Line
Tributary or Secondary Station A
Modem
Terminal
Communications Adapter
Storage
Control
Data Terminal Ready¹
Data Set Ready¹
Request to Send¹ 1
Clear to Send¹
Power On
Carrier Generate
Delay
Transmit
Modulator
Transmitter Signal Element Timing¹
Modem Clock
2
AA
Transmitted Data
3
Received Line Signal Detector
4
Receiver Signal Element Timing¹
Receiver
Demodulator
Serdes
Received Data
AA
Power On
Receiver
Modem Clock
Demodulator
Data Terminal Ready¹
Data Set Ready¹
Received Line Signal Detector¹
Receiver Signal Element Timing¹
5
Received Data
6
Request to Send
7
Clear to Send
Carrier Generate
Delay
Transmit
Modem Clock
Modulator
Transmitter Signal Element Timing
8
Transmitted Data
EOT
AA
Serdes
Storage
Control
¹These lines are active continuously.

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.


Terminal A
Communications Adapter
Modem A
CBS Coupler
CBS Coupler
Modem B
Terminal B
Communications Adapter
Storage
Control
Data Terminal Ready
Data Set Ready
Request to Send
Clear to Send
Carrier Generate
Delay
Modulator
Switch Hook (SH)
Off Hook (OH)
Coupler Cut-Through (CCT)
Data Modem Ready (DA)
Transmit Data
Data Tip (DT)
Data
Data Ring (DR)
Relays
Echo Clamp
Modem Clock
Serdes
Transmit Clock
Switch Hook (SH)
Off Hook (OH)
Coupler Cut-Through (CCT)
Data Modem Ready (DA)
Ring Indicate (R)
Data Tip (DT)
Data
Data Ring (DR)
Autoanswer
Answer Tone
Demodulator
Modem Clock
Data Terminal Ready
Data Set Ready
Received Line Signal Detector
Ring Indicator
Received Data
Receive Clock
Communications Line

Notes:

```
LOC OBJ                 LINE    SOURCE

                        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                             ; NO MORE INTERRUPTS
F395                    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            LODSW                           ; GET THE CHAR/ATTR
F39B E927FE             3982            JMP     VIDEO_RETURN
                        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 33DB               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 03D8               3998            ADD     BX,AX                   ; ADD TO START OF REGEN
F3B8 C3                 3999            RET
                        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 8BFB               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                             ; 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                             ; 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 E8A0FF             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                             ; 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                             ; 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)  :
```

Glossary

μ. Prefix micro; 0.000 001.

μs. Microsecond; 0.000 001 second.

A. Ampere.

ac. Alternating current.

accumulator. A register in which the result of an operation is formed.

active high. Designates a signal that has to go high to produce an effect. Synonymous with positive true.

active low. Designates a signal that has to go low to produce an effect. Synonymous with negative true.

adapter. An auxiliary device 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 processor throughout the rest of the system.

algorithm. A finite set of well-defined rules for the solution of a problem in a finite number of steps.

all points addressable (APA). A mode in which all points of a displayable image can be controlled by the user.

alphameric. Synonym for alphanumeric.

alphanumeric (A/N). Pertaining to a character set that contains letters, digits, and usually other characters, such as punctuation marks. Synonymous with alphameric.

alternating current (ac). A current that periodically reverses its direction of flow.

American National Standard Code for Information Exchange (ASCII). The standard code, using a coded character set consisting of 7-bit coded characters (8 bits including parity check), used for information exchange between data processing systems, data communication systems, and associated equipment. The ASCII set consists of control characters and graphic characters.

ampere (A). The basic unit of electric current.

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.

AND gate. A logic gate in which the output is 1 only if all inputs are 1.

AND operation. The boolean operation whose result has the boolean value 1, if and only if, each operand has the boolean value 1. Synonymous with conjunction.

APA. All points addressable.

ASCII. American National Standard Code for Information Exchange.

assemble. To translate a program expressed in an assembler language into a computer language.

assembler. A computer program used to assemble.

assembler language. A computer-oriented language whose instructions are usually in one-to-one correspondence with computer instructions.

asynchronous transmission. (1) Transmission in which the time of occurrence of the start of each character, or block of characters, is arbitrary; once started, the time of occurrence of each signal representing a bit within a character, or block, has the same relationship to significant instants of a fixed time frame. (2) Transmission in which each information character is individually transmitted (usually timed by the use of start elements and stop elements).

audio frequencies. Frequencies that can be heard by the human ear (approximately 15 hertz to 20 000 hertz).

auxiliary storage. (1) A storage device that is not main storage. (2) Data storage other than main storage; for example, storage on magnetic disk. (3) Contrast with main storage.

BASIC. Beginner's all-purpose symbolic instruction code.

basic input/output system (BIOS). The feature of the IBM Personal Computer that provides the level control of the major I/O devices, and relieves the programmer from concern about 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 bit per second in a train of binary signals, one-half dot cycle per second in Morse code, 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 numeric applications.

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 2.

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 uniform procedure, using a standardized set of control characters and control character sequences for synchronous transmission of binary–coded data between stations.

BIOS. Basic input/output system.

bit. Synonym for binary digit

bits per second (bps). A unit of measurement representing the number of discrete binary digits transmitted by a device in one second.

block. (1) A string of records, a string of words, or a character string formed for technical or logic reasons to be treated as an entity. (2) A set of things, such as words, characters, or digits, treated as a unit.

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; for example, 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 sequence of eight adjacent binary digits that are operated upon as a unit. (2) A binary character operated upon as a unit. (3) The representation of a character.

C. Celsius.

capacitor. An electronic circuit component that stores an electric charge.

CAS. Column address strobe.

cathode ray tube (CRT). A vacuum tube in which a stream of electrons is projected onto a fluorescent screen producing a luminous spot. The location of the spot can be controlled.

cathode ray tube display (CRT display). (1) A CRT used for displaying data. For example, the electron beam can be controlled to form alphanumeric data by use of a dot matrix. (2) Synonymous with monitor.

CCITT. International Telegraph and Telephone Consultative Committee.

Celsius (C). A temperature scale. Contrast with Fahrenheit (F).

central processing unit (CPU). Term for processing unit.

channel. A path along which signals can be sent; for example, data channel, output channel.

character generator. (1) In computer graphics, a functional unit that converts the coded representation of a graphic character into the shape of the character for display. (2) In word processing, the means within equipment for generating visual characters or symbols from coded data.

character set. (1) A finite set of different characters upon which agreement has been reached and that is considered complete for some purpose. (2) A set of unique representations called characters. (3) A defined collection of characters.

characters per second (cps). A standard unit of measurement for the speed at which a printer prints.

check key. A group of characters, derived from and appended to a data item, that can be used to detect errors in the data item during processing.

closed circuit. A continuous unbroken circuit; that is, one in which current can flow. Contrast with open circuit.

CMOS. Complementary metal oxide semiconductor.

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) To represent data or a computer program in a symbolic form that can be accepted by a data processor. (4) Loosely, one or more computer programs, or part of a computer program.

coding scheme. Synonym for code.

collector. An element in a transistor toward which current flows.

column address strobe (CAS). A signal that latches the column addresses in a memory chip.

compile. (1) To translate a computer program expressed in a problem-oriented language into a computer-oriented language. (2) To prepare a machine-language program from a computer program written in another programming language by making use of the overall logic structure of the program, or generating more than one computer instruction for each symbolic statement, or both, as well as performing the function of an assembler.

complementary metal oxide semiconductor (CMOS). A logic circuit family that uses very little power. It works with a wide range of power supply voltages.

computer. A functional unit that can perform substantial computation, including numerous arithmetic operations or logic operations, without intervention by a human operator during a run.

computer instruction code. A code used to represent the instructions in an instruction set. Synonymous with machine code.

computer program. A sequence of instructions suitable for processing by a computer.

computer word. A word stored in one computer location and capable of being treated as a unit.

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. Synonym for AND operation.

contiguous. Touching or joining at the edge or boundary; adjacent.

control character. A character whose occurrence in a particular context initiates, modifies, or stops a control operation.

control operation. An action that affects the recording, processing, transmission, or interpretation of data; for example, starting or stopping a process, carriage return, font change, rewind, and end of transmission.

control storage. A portion of storage that contains microcode.

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.

cursor. (1) In computer graphics, a movable marker that is used to indicate a position on a display. (2) A displayed symbol that acts as a marker to help the user locate a point in text, in a system command, or in storage. (3) A movable spot of light on the screen of a display device, usually indicating where the next character is to be entered, replaced, or deleted.

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 human or automatic means. (2) Any representations, such as characters or analog quantities, to which meaning is, or might be assigned.

data base. A collection of data that can be immediately accessed and operated upon by a data processing system for a specific purpose.

data processing system. A system that performs input, processing, storage, output, and control functions to accomplish a sequence of operations on data.

data transmission. Synonym for transmission.

dB. Decibel.

dBa. Adjusted decibels.

dc. Direct current.

debounce. An electronic means of overcoming the make/break bounce of switches to obtain one smooth change of signal level.

decibel. (1) A unit that expresses the ratio of two power levels on a logarithmic scale. (2) A unit for measuring relative power.

decoupling capacitor. A capacitor that provides a low impedance path to ground to prevent common coupling between circuits.

Deutsche Industrial 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 0 to 9.

digital. (1) Pertaining to data in the form of digits. (2) Contrast with analog.

DIN. Deutsche Industrial Norm.

DIN connector. One of the connectors specified by the DIN committee.

DIP. Dual in-line package.

DIP switch. One of a set of small switches mounted in a dual in-line package.

direct current (dc). A current that always flows in one direction.

direct memory access (DMA). A method of transferring data between main storage and I/O devices that does not require processor intervention.

disable. To stop the operation of a circuit or device.

disabled. Pertaining to a state of a processing unit that prevents the occurrence of certain types of interruptions. Synonymous with masked.

disk. Loosely, a magnetic disk.

diskette. A thin, flexible magnetic disk and a semirigid protective jacket, in which the disk is permanently enclosed. Synonymous with flexible disk.

diskette drive. A device for storing data on and retrieving data from a diskette.

display. (1) A visual presentation of data. (2) A device for visual presentation of information on any temporary character imaging device. (3) To present data visually. (4) See cathode ray tube display.

display attribute. In computer graphics, a particular property that is assigned to all or part of a display; for example, low intensity, green color, blinking status.

DMA. Direct memory access.

dot matrix. (1) In computer graphics, a two-dimensional pattern of dots used for constructing a display image. This type of matrix can be used to represent characters by dots. (2) In word processing, a pattern of dots used to form characters. This term normally refers to a small section of a set of addressable points; for example, a representation of characters by dots.

dot printer. Synonym for matrix printer.

dot-matrix character generator. In computer graphics, a character generator that generates character images composed of dots.

DSR. Data set ready. Associated with modem control.

DTR. In the IBM Personal Computer, data terminal ready. Associated with modem control.

dual in-line package (DIP). A widely used container for an integrated circuit. DIPs have pins in two parallel rows. The pins are spaced 1/10 inch apart. See also DIP switch.

duplex. (1) In data communication, pertaining to a simultaneous two-way independent transmission in both directions. (2) Contrast with half-duplex.

duty cycle. In the operation of a device, the ratio of on time to idle time. Duty cycle is expressed as a decimal or percentage.

dynamic memory. RAM using transistors and capacitors as the memory elements. This memory requires a refresh (recharge) cycle every few milliseconds. Contrast with static memory.

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.

electromagnet. Any device that exhibits magnetism only while an electric current flows through it.

enable. To initiate the operation of a circuit or device.

end of block (EOB). A code that marks the end of a block of data.

end of file (EOF). An internal label, immediately following the last record of a file, signaling the end of that file. It may include control totals for comparison with counts accumulated during processing.

end-of-text (ETX). A transmission control character used to terminate text.

end-of-transmission (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.

end-of-transmission-block (ETB). A transmission control character used to indicate the end of a transmission block of data when data is divided into such blocks for transmission purposes.

EOB. End of block.

EOF. End of file.

EOT. End-of-transmission.

EPROM. Erasable programmable read-only memory.

erasable programmable read-only memory (EPROM). A PROM in which the user can erase old information and enter new information.

error checking and correction (ECC). The detection and correction of all single-bit errors, plus the detection of double-bit and some multiple-bit errors.

ESC. The escape character.

escape character (ESC). A code extension character used, in some cases, with one or more succeeding characters to indicate by some convention or agreement that the coded representations following the character or the group of characters are to be interpreted according to a different code or according to a different coded character set.

ETB. End-of-transmission-block.

ETX. End-of-text.

extended binary-coded decimal interchange code (EBCDIC). A set of 256 characters, each represented by eight bits.

F. Fahrenheit.

Fahrenheit (F). A temperature scale. Contrast with Celsius (C).

falling edge. Synonym for negative-going edge.

FCC. Federal Communications Commission.

fetch. To locate and load a quantity of data from storage.

FF. The form feed character.

field. (1) In a record, a specified area used for a particular category of data. (2) In a data base, the smallest unit of data that can be referred to.

fixed disk drive. In the IBM Personal Computer, a unit consisting of nonremovable magnetic disks, and a device for storing data on and retrieving data from the disks.

flag. (1) Any of various types of indicators used for identification. (2) A character that signals the occurrence of some condition, such as the end of a word. (3) Deprecated term for mark.

flexible disk. Synonym for diskette.

flip-flop. A circuit or device containing active elements, capable of assuming either one of two stable states at a given time.

font. A family or assortment of characters of a given size and style; for example, 10 point Press Roman medium.

foreground. (1) In multiprogramming, the environment in which high-priority programs are executed. (2) On a color display screen, the characters as opposed to the background.

form feed. (1) Paper movement used to bring an assigned part of a form to the printing position. (2) In word processing, a function that advances the typing position to the same character position on a predetermined line of the next form or page.

form feed character. A control character that causes the print or display position to move to the next predetermined first line on the next form, the next page, or the equivalent.

format. The arrangement or layout of data on a data medium.

frame. (1) In SDLC, the vehicle for every command, every response, and all information that is transmitted using SDLC procedures. Each frame begins and ends with a flag. (2) In data transmission, the sequence of contiguous bits bracketed by and including beginning and ending flag sequences.

g. Gram.

G. (1) Prefix giga; 1 000 000 000. (2) When referring to computer storage capacity, 1 073 741 824. (1 073 741 824 = 2 to the 30th power.)

gate. (1) A combinational logic circuit having one output channel and one or more input channels, such that the output channel state is completely determined by the input channel states. (2) A signal that enables the passage of other signals through a circuit.

Gb. 1 073 741 824 bytes.

general-purpose register. A register, usually explicitly addressable within a set of registers, that can be used for different purposes; for example, as an accumulator, as an index register, or as a special handler of data.

giga (G). Prefix 1 000 000 000.

gram (g). A unit of weight (equivalent to 0.035 ounces).

graphic. A symbol produced by a process such as handwriting, drawing, or printing.

graphic character. A character, other than a control character, that is normally represented by a graphic.

half-duplex. (1) In data communication, pertaining to an alternate, one way at a time, independent transmission. (2) Contrast with duplex.

hardware. (1) Physical equipment used in data processing, as opposed to programs, procedures, rules, and associated documentation. (2) Contrast with software.

head. A device that reads, writes, or erases data on a storage medium; for example, a small electromagnet used to read, write, or erase data on a magnetic disk.

hertz (Hz). A unit of frequency equal to one cycle per second.

hex. Common abbreviation for hexadecimal.

hexadecimal. (1) Pertaining to a selection, choice, or condition that has 16 possible different values or states. These values or states are usually symbolized by the ten digits 0 through 9 and the six letters A through F. (2) Pertaining to a fixed radix numeration system having a radix of 16.

high impedance state. A state in which the output of a device is effectively isolated from the circuit.

highlighting. In computer graphics, emphasizing a given display group by changing its attributes relative to other display groups in the same display field.

high-order position. The leftmost position in a string of characters. See also most-significant digit.

housekeeping. Operations or routines that do not contribute directly to the solution of the problem but do contribute directly to the operation of the computer.

Hz. Hertz

image. A fully processed unit of operational data that is ready to be transmitted to a remote unit; when loaded into control storage in the remote unit, the image determines the operations of the unit.

immediate instruction. An instruction that contains within itself an operand for the operation specified, rather than an address of the operand.

index register. A register whose contents may be used to modify an operand address during the execution of computer instructions.

indicator. (1) A device that may be set into a prescribed state, usually according to the result of a previous process or on the occurrence of a specified condition in the equipment, and that usually gives a visual or other indication of the existence of the prescribed state, and that may in some cases be used to determine the selection among alternative processes; for example, an overflow indicator. (2) An item of data that may be interrogated to determine whether a particular condition has been satisfied in the execution of a computer program; for example, a switch indicator, an overflow indicator.

inhibited. (1) Pertaining to a state of a processing unit in which certain types of interruptions are not allowed to occur. (2) Pertaining to the state in which a transmission control unit or an audio response unit cannot accept incoming calls on a line.

initialize. To set counters, switches, addresses, or contents of storage to 0 or other starting values at the beginning of, or at prescribed points in, the operation of a computer routine.

input/output (I/O). (1) Pertaining to a device or to a channel that may be involved in an input process, and, at a different time, in an output process. In the English language, "input/output" may be used in place of such terms as "input/output data," "input/output signal," and "input/output terminals," when such usage is clear in a given context. (2) Pertaining to a device whose parts can be performing an input process and an output process at the same time. (3) Pertaining to either input or output, or both.

instruction. In a programming language, a meaningful expression that specifies one operation and identifies its operands, if any.

instruction set. The set of instructions of a computer, of a programming language, or of the programming languages in a programming system.

interface. A device that alters or converts actual electrical signals between distinct devices, programs, or systems.

interleave. To arrange parts of one sequence of things or events so that they alternate with parts of one or more other sequences of the same nature and so that each sequence retains its identity.

interrupt. (1) A suspension of a process, such as the execution of a computer program, caused by an event external to that process, and performed in such a way that the process can be resumed. (2) In a data transmission, to take an action at a receiving station that causes the transmitting station to terminate a transmission. (3) Synonymous with interruption.

I/O. Input/output.

I/O area. Synonym for buffer.

irrecoverable error. An error that makes recovery impossible without the use of recovery techniques external to the computer program or run.

joystick. In computer graphics, a lever that can pivot in all directions and that is used as a locator device.

k. Prefix kilo; 1000.

K. When referring to storage capacity, 1024. (1024 = 2 to the 10th power.)

Kb. 1024 bytes.

kg. Kilogram; 1000 grams.

kHz. Kilohertz; 1000 hertz.

kilo (k). Prefix 1000

kilogram (kg). 1000 grams.

kilohertz (kHz). 1000 hertz

latch. (1) A simple logic-circuit storage element. (2) A feedback loop in sequential digital circuits used to maintain a state.

least-significant digit. The rightmost digit. See also low-order position.

LED. Light-emitting diode.

light-emitting diode (LED). A semiconductor device that gives off visible or infrared light when activated.

load. In programming, to enter data into storage or working registers.

low power Schottky TTL. A version (LS series) of TTL giving a good compromise between low power and high speed. See also transistor-transistor logic and Schottky TTL.

low-order position. The rightmost position in a string of characters. See also least-significant digit.

m. (1) Prefix milli; 0.001. (2) Meter.

M. (1) Prefix mega; 1 000 000. (2) When referring to computer storage capacity, 1 048 576. (1 048 576 = 2 to the 20th power.)

mA. Milliampere; 0.001 ampere.

machine code. The machine language used for entering text and program instructions onto the recording medium or into storage and which is subsequently used for processing and printout.

machine language. (1) A language that is used directly by a machine. (2) Deprecated term for computer instruction code.

magnetic disk. (1) A flat circular plate with a magnetizable surface layer on which data can be stored by magnetic recording. (2) See also diskette.

main storage. (1) Program-addressable storage from which instructions and other data can be loaded directly into registers for subsequent execution or processing. (2) Contrast with auxiliary storage.

mark. A symbol or symbols that indicate the beginning or the end of a field, of a word, of an item of data, or of a set of data such as a file, a record, or a block.

mask. (1) A pattern of characters that is used to control the retention or elimination of portions of another pattern of characters. (2) To use a pattern of characters to control the retention or elimination of portions of another pattern of characters.

masked. Synonym for disabled.

matrix. (1) A rectangular array of elements, arranged in rows and columns, that may be manipulated according to the rules of matrix algebra. (2) In computers, a logic network in the form of an array of input leads and output leads with logic elements connected at some of their intersections.

matrix printer. A printer in which each character is represented by a pattern of dots; for example, a stylus printer, a wire printer. Synonymous with dot printer.

Mb. 1 048 576 bytes.

mega (M). Prefix 1 000 000.

megahertz (MHz). 1 000 000 hertz.

memory. Term for main storage.

meter (m). A unit of length (equivalent to 39.37 inches).

MFM. Modified frequency modulation.

MHz. Megahertz; 1 000 000 hertz.

micro (μ). Prefix 0.000 001.

microcode. (1) One or more microinstructions. (2) A code, representing the instructions of an instruction set, implemented in a part of storage that is not program-addressable.

microinstruction. (1) An instruction of microcode. (2) A basic or elementary machine instruction.

microprocessor. An integrated circuit that accepts coded instructions for execution; the instructions may be entered, integrated, or stored internally.

microsecond (μs). 0.000 001 second.

milli (m). Prefix 0.001.

milliampere (mA). 0.001 ampere.

millisecond (ms). 0.001 second.

mnemonic. A symbol chosen to assist the human memory; for example, an abbreviation such as "mpy" for "multiply."

mode. (1) A method of operation; for example, the binary mode, the interpretive mode, the alphanumeric mode. (2) The most frequent value in the statistical sense.

modem (modulator-demodulator). A device that converts serial (bit by bit) digital signals from a business machine (or data

communication equipment) to analog signals that 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.

modulation. The process by which some characteristic of one wave (usually high frequency) is varied in accordance with another wave or signal (usually low frequency). This technique is used in modems to make business-machine signals compatible with communication facilities.

modulation rate. The reciprocal of the measure of the shortest nominal time interval between successive significant instants of the modulated signal. If this measure is expressed in seconds, the modulation rate is expressed in baud.

module. (1) A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading. (2) A packaged functional hardware unit designed for use with other components.

modulo check. A calculation performed on values entered into a system. This calculation is designed to detect errors.

monitor. Synonym for cathode ray tube display (CRT display).

most-significant digit. The leftmost (non-zero) digit. See also high-order position.

ms. Millisecond; 0.001 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.

multiprogramming. (1) Pertaining to the concurrent execution of two or more computer programs by a computer. (2) A mode of operation that provides for the interleaved execution of two or more computer programs by a single processor.

n. Prefix nano; 0.000 000 001.

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.

NAND gate. A gate in which the output is 0 only if all inputs are 1.

nano (n). Prefix 0.000 000 001.

nanosecond (ns). 0.000 000 001 second.

negative true. Synonym for active low.

negative-going edge. The edge of a pulse or signal changing in a negative direction. Synonymous with falling edge.

non-return-to-zero change-on-ones recording (NRZI). A transmission encoding method in which the data terminal equipment changes the signal to the opposite state to send a binary 1 and leaves it in the same state to send a binary 0.

non-return-to-zero (inverted) recording (NRZI). Deprecated term for non-return-to-zero change-on-ones recording.

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.

NOR gate. A gate in which the output is 0 only if at least one input is 1.

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 change-on-ones recording.

ns. Nanosecond; 0.000 000 001 second.

NUL. The null character.

null character (NUL). A control character that is used to accomplish media-fill or time-fill, and that may be inserted into or removed from, a sequence of characters without affecting the meaning of the sequence; however, the control of the equipment or the format may be affected by this character.

odd-even check. Synonym for parity check.

offline. Pertaining to the operation of a functional unit without the continual control of a computer.

one-shot. A circuit that delivers one output pulse of desired duration for each input (trigger) pulse.

open circuit. (1) A discontinuous circuit; that is, one that is broken at one or more points and, consequently, cannot conduct current. Contrast with closed circuit. (2) Pertaining to a no-load condition; for example, the open-circuit voltage of a power supply.

open collector. A switching transistor without an internal connection between its collector and the voltage supply. A connection from the collector to the voltage supply is made through an external (pull-up) resistor.

operand. (1) An entity to which an operation is applied. (2) That which is operated upon. An operand is usually identified by an address part of an instruction.

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.

OR gate. A gate in which the output is 1 only if at least one input is 1.

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.

overflow indicator. (1) An indicator that signifies when the last line on a page has been printed or passed. (2) An indicator that is set on if the result of an arithmetic operation exceeds the capacity of the accumulator.

overrun. Loss of data because a receiving device is unable to accept data at the rate it is transmitted.

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.

parameter. (1) A variable that is given a constant value for a specified application and that may denote the application. (2) A name in a procedure that is used to refer to an argument passed to that procedure.

parity bit. A binary digit appended to a group of binary digits to make the sum of all the digits either always odd (odd parity) or always even (even parity).

parity check. (1) A redundancy check that uses a parity bit. (2) Synonymous with odd-even check.

PEL. Picture element.

personal computer. A small home or business computer that has a processor and keyboard and that can be connected to a television or some other monitor. An optional printer is usually available.

phototransistor. A transistor whose switching action is controlled by light shining on it.

picture element (PEL). The smallest displayable unit on a display.

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.

positive true. Synonym for active high.

positive-going edge. The edge of a pulse or signal changing in a positive direction. Synonymous with rising edge.

potentiometer. A variable resistor with three terminals, one at each end and one on a slider (wiper).

power supply. A device that produces the power needed to operate electronic equipment.

printed circuit. A pattern of conductors (corresponding to the wiring of an electronic circuit) formed on a board of insulating material.

printed-circuit board. A usually copper-clad plastic board used to make a printed circuit.

priority. A rank assigned to a task that determines its precedence in receiving system resources.

processing program. A program that performs such functions as compiling, assembling, or translating for a particular programming language.

processing unit. A functional unit that consists of one or more processors and all or part of internal storage.

processor. (1) In a computer, a functional unit that interprets and executes instructions. (2) A functional unit, a part of another unit such as a terminal or a processing unit, that interprets and executes instructions. (3) Deprecated term for processing program. (4) See microprocessor.

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.

programmable read-only memory (PROM). A read-only memory that can be programmed by the user.

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.

programming system. One or more programming languages and the necessary software for using these languages with particular automatic data-processing equipment.

PROM. Programmable read-only memory.

propagation delay. (1) The time necessary for a signal to travel from one point on a circuit to another. (2) The time delay between a signal change at an input and the corresponding change at an output.

protocol. (1) A specification for the format and relative timing of information exchanged between communicating parties. (2) The set of rules governing the operation of functional units of a communication system that must be followed if communication is to be achieved.

pulse. A variation in the value of a quantity, short in relation to the time schedule of interest, the final value being the same as the initial value.

radio frequency (RF). An ac frequency that is higher than the highest audio frequency. So called because of the application to radio communication.

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 radix). The permissible values of the character in any digit place range from 0 to one less than the radix.

RAM. Random access memory. Read/write memory.

random access memory (RAM). Read/write memory.

RAS. In the IBM Personal Computer, row address strobe.

raster. In computer graphics, a predetermined pattern of lines that provides uniform coverage of a display space.

read. To acquire or interpret data from a storage device, from a data medium, or from another source.

read-only memory (ROM). A storage device whose contents cannot be modified. The memory is retained when power is removed.

read/write memory. A storage device whose contents can be modified. Also called RAM.

recoverable error. An error condition that allows continued execution of a program.

red-green-blue-intensity (RGBI). The description of a direct-drive color monitor that accepts input signals of red, green, blue, and intensity.

redundancy check. A check that depends on extra characters attached to data for the detection of errors. See cyclic redundancy check.

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) A storage device in which specific data is stored.

retry. To resend the current block of data (from the last EOB or ETB) a prescribed number of times, or until it is entered correctly or accepted.

reverse video. A form of highlighting a character, field, or cursor by reversing the color of the character, field, or cursor with its background; for example, changing a red character on a black background to a black character on a red background.

RF. Radio frequency.

RF modulator. The device used to convert the composite video signal to the antenna level input of a home TV.

RGBI. Red-green-blue-intensity.

rising edge. Synonym for positive-going edge.

ROM. Read-only memory.

ROM/BIOS. The ROM resident basic input/output system, which provides the level control of the major I/O devices in the computer system.

row address strobe (RAS). A signal that latches the row address in a memory chip.

RS-232C. A standard by the EIA for communication between computers and external equipment.

RTS. Request to send. Associated with modem control.

run. A single continuous performance of a computer program or routine.

schematic. The representation, usually in a drawing or diagram form, of a logical or physical structure.

Schottky TTL. A version (S series) of TTL with faster switching speed, but requiring more power. See also transistor-transistor logic and low power Schottky TTL.

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.

serializer/deserializer (SERDES). A device that serializes output from, and deserializes input to, a business machine.

setup. (1) In a computer that consists of an assembly of individual computing units, the arrangement of interconnections between the units, and the adjustments needed for the computer to operate. (2) The preparation of a computing system to perform a job or job step. Setup is usually performed by an operator and often involves performing routine functions, such as mounting tape reels. (3) The preparation of the system for normal operation.

short circuit. A low-resistance path through which current flows, rather than through a component or circuit.

signal. A variation of a physical quantity, used to convey data.

sink. A device or circuit into which current drains.

software. (1) Computer programs, procedures, and rules concerned with the operation of a data processing system. (2) Contrast with hardware.

source. The origin of a signal or electrical energy.

square wave. An alternating or pulsating current or voltage whose waveshape is square.

square wave generator. A signal generator delivering an output signal having a square waveform.

SS. Start-stop.

start bit. (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.

start-of-text (STX). A transmission control character that precedes a text and may be used to terminate the message heading.

start-stop system. A data transmission system in which each character is preceded by a start bit and is followed by a stop bit.

start-stop (SS) transmission. (1) Asynchronous transmission such that a group of signals representing a character is preceded by a start bit and followed by a stop bit. (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.

static memory. RAM using flip-flops as the memory elements. Data is retained as long as power is applied to the flip-flops. Contrast with dynamic memory.

stop bit. (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.

storage. (1) A storage device. (2) A device, or part of a device, that can retain data. (3) The retention of data in a storage device. (4) The placement of data into a storage device.

strobe. An instrument that emits adjustable-rate flashes of light. Used to measure the speed of rotating or vibrating objects.

STX. Start-of-text.

symbol. (1) A conventional representation of a concept. (2) A representation of something by reason of relationship, association, or convention.

synchronization. The process of adjusting the corresponding significant instants of two signals to obtain the desired phase relationship between these instants.

Synchronous Data Link Control (SDLC). A protocol for management of data transfer over a data link.

synchronous transmission. (1) Data transmission in which the time of occurrence of each signal representing a bit is related to a fixed time frame. (2) Data transmission in which the sending and receiving devices are operating continuously at substantially the same frequency and are maintained, by means of correction, in a desired phase relationship.

syntax. (1) The relationship among characters or groups of characters, independent of their meanings or the manner of their interpretation and use. (2) The structure of expressions in a language. (3) The rules governing the structure of a language. (4) The relationships among symbols.

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 character, respectively.

time-out. (1) A parameter related to an enforced event designed to occur at the conclusion of a predetermined elapsed time. A time-out condition can be cancelled by the receipt of an appropriate time-out cancellation signal. (2) A time interval

allotted for certain operations to occur; for example, response to polling or addressing before system operation is interrupted and must be restarted.

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, or disk, that is accessible to a given reading head position.

transistor-transistor logic (TTL). A popular logic circuit family that uses multiple-emitter transistors.

translate. To transform data from one language to another.

transmission. (1) The sending of data from one place for reception elsewhere. (2) In ASCII and data communication, a series of characters including headings and text. (3) The dispatching of a signal, message, or other form of intelligence by wire, radio, telephone, or other means. (4) One or more blocks or messages. For BSC and start-stop devices, a transmission is terminated by an EOT character. (5) Synonymous with data transmission.

TTL. Transistor-transistor logic.

V. Volt.

video. Computer data or graphics displayed on a cathode ray tube, monitor, or display.

volt. The basic practical unit of electric pressure. The potential that causes electrons to flow through a circuit.

W. Watt.

watt. The practical unit of electric power.

word. (1) A character string or a bit string considered as an entity. (2) See computer word.

write. To make a permanent or transient recording of data in a storage device or on a data medium.

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 Libary.* This book describes Motorola components and their technical specifications.

National Semiconductor Corporation. *250 Asynchronous Communications Element* book documents physical and operating characteristics of the INS 8250.

Index

Special Characters

A

B

C

D

E

I

K

L

M

N

O

P

Q

R

S

T

V

W

Numerals

The Personal Computer
Hardware Library

Reader's Comment Form

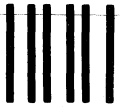
Technical Reference
IBM Personal Computer XT
IBM Portable Personal Computer 6361459

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

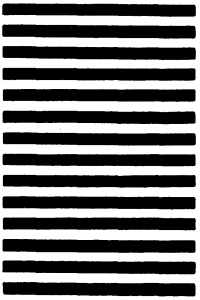
IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

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. 321 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

Tape Please do not staple Tape

The Personal Computer
Hardware Library

Reader's Comment Form

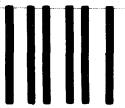
Technical Reference
IBM Personal Computer XT
IBM Portable Personal Computer 6361459

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

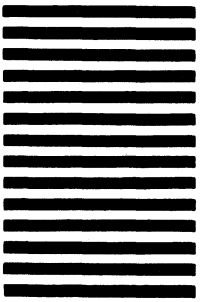
IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

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. 321 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

Tape

Please do not staple

Reader's Comment Form

Technical Reference
IBM Personal Computer XT
IBM Portable Personal Computer 6361459

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. 321 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

The Personal Computer
Hardware Library

Reader's Comment Form

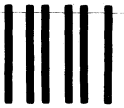
Technical Reference
IBM Personal Computer XT
IBM Portable Personal Computer 6361459

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

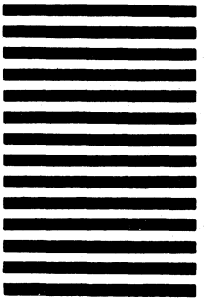
IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

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. 321 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

6361459

Printed in United States of America