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

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

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

Revised Edition (April 1983)

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

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

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

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

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

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

This manual has two sections:

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

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

The publication has seven appendixes:

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

A glossary and bibliography are included.
Prerequisite Publication:

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

Suggested Reading:

BASIC for the IBM Personal Computer  
Part Number 6025010

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

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

*MACRO Assembler* for the IBM Personal Computer  
Part Number 6024002

Related publications are listed in the bibliography.
TABLE OF CONTENTS

Section 1: Hardware

IBM Personal Computer XT System Unit .................. 1-3
IBM Personal Computer Math Coprocesser ............... 1-25
IBM Keyboard ........................................ 1-65
IBM Expansion Unit .................................... 1-71
IBM 80 CPS Printers ..................................... 1-81
IBM Printer Adapter ..................................... 1.107
IBM Monochrome Display and Printer Adapter ............ 1-113
IBM Monochrome Display ................................ 1.121
IBM Color/Graphics Display Adapter ..................... 1.123
IBM Color Display ....................................... 1.149
IBM 5-½” Diskette Drive Adapter ........................ 1-151
IBM 5-½” Diskette Drive ................................ 1-175
Diskettes .................................................. 1.177
IBM Fixed Disk Drive Adapter ............................ 1-179
IBM 10MB Fixed Disk Drive ............................... 1.195
IBM Memory Expansion Options ......................... 1.197
IBM Game Control Adapter ............................... 1-203
IBM Prototype Card ...................................... 1-209
IBM Asynchronous Communications Adapter .............. 1-215
IBM Binary Synchronous Communications Adapter ....... 1-245
IBM Synchronous Data Link Control (SDLC) Communication Adapter .................................. 1-265
IBM Communications Adapter Cable ..................... 1-295

Section 2: ROM BIOS and System Usage

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

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

System BIOS .............................................. A-2
Fixed Disk BIOS .......................................... A-85

Appendix B: 8088 Assembly Instruction
Set Reference ............................................. B-1
Appendix C: Of Characters, Keystrokes, and Colors ......................... C-1

Appendix D: Logic Diagrams ............................................. D-1

System Board ............................................................. D-2
Type 1 Keyboard ......................................................... D-12
Type 2 Keyboard ......................................................... D-14
Expansion Board ....................................................... D-15
Extender Card ............................................................ D-16
Receiver Card ............................................................. D-19
Printer ................................................................. D-22
Printer Adapter .......................................................... D-25
Monochrome Display Adapter ......................................... D-26
Color/Graphics Monitor Adapter ....................................... D-36
Color Display ......................................................... D-42
Monochrome Display .................................................. D-44
5-1/4 Inch Diskette Drive Adapter ..................................... D-45
5-1/4 Inch Diskette Drive - Type 1 .................................... D-49
5-1/4 Inch Diskette Drive - Type 2 .................................... D-52
Fixed Disk Drive Adapter ............................................... D-54
Fixed Disk Drive - Type 1 ............................................... D-60
Fixed Disk Drive - Type 2 ............................................... D-63
32K Memory Expansion Option ........................................ D-66
64K Memory Expansion Option ......................................... D-69
64/256K Memory Expansion Option .................................... D-72
Game Control Adapter .................................................. D-76
Prototype Card ......................................................... D-77
Asynchronous Communications Adapter ............................. D-78
Binary Synchronous Communications Adapter ..................... D-79
SDLC Communications Adapter ......................................... D-81

Appendix E: Specifications ............................................... E-1

Appendix F: Communications ............................................. F-1

Appendix G: Switch Settings ............................................. G-1

Glossary ...................................................................... H-1

Index .......................................................................... I-1
INDEX TAB LISTING

Section 1: Hardware ..............................................

Section 2: ROM BIOS and System Usage ..................

Appendix A: ROM BIOS Listings .........................

Appendix B: 8088 Assembly Instruction .................
Set Reference

Appendix C: Of Characters, Keystrokes, ...............
and Color

Appendix D: Logic Diagrams ...............................
SECTION 1: HARDWARE

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

1-2 System Unit
IBM Personal Computer XT System Unit

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

System Board

The system board fits horizontally in the base of the system unit and is approximately 8-1/2 by 12 inches. It is a multilayer, single-land-per-channel design with ground and internal planes provided. DC power and a signal from the power supply enter the board through two six-pin connectors. Other connectors on the board are for attaching the keyboard and speaker. Eight 62-pin card edge-sockets are also mounted on the board. The I/O channel is bussed across these eight I/O slots. Slot J8 is slightly different from the others in that any card placed in it is expected to respond with a 'card selected' signal whenever the card is selected.

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

The system board consists of five functional areas: the processor subsystem and its support elements, the read-only memory (ROM) subsystem, the read/write (R/W) memory subsystem, integrated I/O adapters, and the I/O channel. All are described in this section.
The heart of the system board is the Intel 8088 microprocessor. This processor is an 8-bit external bus version of Intel's 16-bit 8086 processor, and is software-compatible with the 8086. Thus, the 8088 supports 16-bit operations, including multiply and divide, and supports 20 bits of addressing (1 megabyte of storage). It also operates in maximum mode, so a co-processor can be added as a feature. The processor operates at 4.77 MHz. This frequency, which is derived from a 14.31818-MHz crystal, is divided by 3 for the processor clock, and by 4 to obtain the 3.58-MHz color burst signal required for color televisions.

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

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

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

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

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

The system board supports both ROM and R/W memory. It has space for 64K by 8 of ROM or EPROM. Two module sockets are provided, each of which can accept a 32K or 8K device. One socket has 32K by 8 of ROM, the other 8K by 8 bytes. This ROM contains the power-on self-test, I/O drivers, dot patterns for 128 characters in graphics mode, and a diskette bootstrap loader. The ROM is packaged in 28-pin modules and has an access time and a cycle time of 250 ns each.

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

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

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

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

The speaker drive circuit is capable of approximately 1/2 watt of power. The control circuits allow the speaker to be driven three different ways: 1.) a direct program control register bit may be toggled to generate a pulse train; 2.) the output from Channel 2 of the timer counter may be programmed to generate a waveform to the speaker; 3.) the clock input to the timer counter can be modulated with a program-controlled I/O register bit. All three methods may be performed simultaneously.
System Board Data Flow (Part 1 of 2)
<table>
<thead>
<tr>
<th>Hex Range</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>000-00F</td>
<td>DMA Chip 8237A-5</td>
</tr>
<tr>
<td>020-021</td>
<td>Interrupt 8259A</td>
</tr>
<tr>
<td>040-043</td>
<td>Timer 8253-5</td>
</tr>
<tr>
<td>060063</td>
<td>PPI 8255A-5</td>
</tr>
<tr>
<td>080083</td>
<td>DMA Page Registers</td>
</tr>
<tr>
<td>0AX*</td>
<td>NMI Mask Register</td>
</tr>
<tr>
<td>OCX</td>
<td>Reserved</td>
</tr>
<tr>
<td>CEX</td>
<td>Reserved</td>
</tr>
<tr>
<td>200-20F</td>
<td>Game Control</td>
</tr>
<tr>
<td>210217</td>
<td>Expansion Unit</td>
</tr>
<tr>
<td>220-24F</td>
<td>Reserved</td>
</tr>
<tr>
<td>278-27F</td>
<td>Reserved</td>
</tr>
<tr>
<td>2F0-2F7</td>
<td>Reserved</td>
</tr>
<tr>
<td>2F8-2FF</td>
<td>Asynchronous Communications (Secondary)</td>
</tr>
<tr>
<td>300-31F</td>
<td>Prototype Card</td>
</tr>
<tr>
<td>320-32F</td>
<td>Fixed Disk</td>
</tr>
<tr>
<td>378-37F</td>
<td>Printer</td>
</tr>
<tr>
<td>380-38C**</td>
<td>SDLC Communications</td>
</tr>
<tr>
<td>380-389**</td>
<td>Binary Synchronous Communications (Secondary)</td>
</tr>
<tr>
<td>3A0-3A9</td>
<td>Binary Synchronous Communications (Primary)</td>
</tr>
<tr>
<td>3B0-3BF</td>
<td>IBM Monochrome Display/Printer</td>
</tr>
<tr>
<td>3C0-3CF</td>
<td>Reserved</td>
</tr>
<tr>
<td>3D0-3DF</td>
<td>Color/Graphics</td>
</tr>
<tr>
<td>3EO-3E7</td>
<td>Reserved</td>
</tr>
<tr>
<td>3F0-3F7</td>
<td>Diskette</td>
</tr>
<tr>
<td>3F8-3FF</td>
<td>Asynchronous Communications (Primary)</td>
</tr>
</tbody>
</table>

* 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

1-8 System Unit
<table>
<thead>
<tr>
<th>Number</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMI</td>
<td>Parity</td>
</tr>
<tr>
<td>0</td>
<td>Timer</td>
</tr>
<tr>
<td>1</td>
<td>Keyboard</td>
</tr>
<tr>
<td>2</td>
<td>Reserved</td>
</tr>
<tr>
<td>3</td>
<td>Asynchronous Communications (Secondary)</td>
</tr>
<tr>
<td></td>
<td>SDLC Communications</td>
</tr>
<tr>
<td></td>
<td>BSC (Secondary)</td>
</tr>
<tr>
<td>4</td>
<td>Asynchronous Communications (Primary)</td>
</tr>
<tr>
<td></td>
<td>SDLC Communications</td>
</tr>
<tr>
<td></td>
<td>BSC (Primary)</td>
</tr>
<tr>
<td>5</td>
<td>Fixed Disk</td>
</tr>
<tr>
<td>6</td>
<td>Diskette</td>
</tr>
<tr>
<td>7</td>
<td>Printer</td>
</tr>
</tbody>
</table>

8088 Hardware Interrupt Listing
8255A I/O Bit Map

1-10 System Unit
<table>
<thead>
<tr>
<th>Start Address</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decimal</td>
<td>Hex</td>
</tr>
<tr>
<td>0</td>
<td>00000</td>
</tr>
<tr>
<td>16K</td>
<td>04000</td>
</tr>
<tr>
<td>32K</td>
<td>08000</td>
</tr>
<tr>
<td>48K</td>
<td>0C000</td>
</tr>
<tr>
<td>64K</td>
<td>10000</td>
</tr>
<tr>
<td>80K</td>
<td>14000</td>
</tr>
<tr>
<td>96K</td>
<td>18000</td>
</tr>
<tr>
<td>112K</td>
<td>1C000</td>
</tr>
<tr>
<td>128K</td>
<td>20000</td>
</tr>
<tr>
<td>144K</td>
<td>24000</td>
</tr>
<tr>
<td>160K</td>
<td>28000</td>
</tr>
<tr>
<td>176K</td>
<td>2C000</td>
</tr>
<tr>
<td>192K</td>
<td>30000</td>
</tr>
<tr>
<td>208K</td>
<td>34000</td>
</tr>
<tr>
<td>224K</td>
<td>38000</td>
</tr>
<tr>
<td>240K</td>
<td>3C000</td>
</tr>
<tr>
<td>256K</td>
<td>40000</td>
</tr>
<tr>
<td>272K</td>
<td>44000</td>
</tr>
<tr>
<td>288K</td>
<td>48000</td>
</tr>
<tr>
<td>304K</td>
<td>4C000</td>
</tr>
<tr>
<td>320K</td>
<td>50000</td>
</tr>
<tr>
<td>336K</td>
<td>54000</td>
</tr>
<tr>
<td>352K</td>
<td>58000</td>
</tr>
<tr>
<td>368K</td>
<td>5C000</td>
</tr>
<tr>
<td>384K</td>
<td>60000</td>
</tr>
<tr>
<td>400K</td>
<td>64000</td>
</tr>
<tr>
<td>416K</td>
<td>68000</td>
</tr>
<tr>
<td>432K</td>
<td>6C000</td>
</tr>
<tr>
<td>448K</td>
<td>70000</td>
</tr>
<tr>
<td>464K</td>
<td>74000</td>
</tr>
<tr>
<td>480K</td>
<td>78000</td>
</tr>
<tr>
<td>496K</td>
<td>7C000</td>
</tr>
<tr>
<td>512K</td>
<td>80000</td>
</tr>
<tr>
<td>528K</td>
<td>84000</td>
</tr>
<tr>
<td>544K</td>
<td>88000</td>
</tr>
<tr>
<td>560K</td>
<td>8C000</td>
</tr>
<tr>
<td>576K</td>
<td>90000</td>
</tr>
<tr>
<td>592K</td>
<td>94000</td>
</tr>
<tr>
<td>608K</td>
<td>98000</td>
</tr>
<tr>
<td>624K</td>
<td>9C000</td>
</tr>
</tbody>
</table>

System Memory Map (Part 1 of 2)
<table>
<thead>
<tr>
<th>Start Address</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>640K A0000</td>
<td>128K Reserved</td>
</tr>
<tr>
<td>656K A4000</td>
<td></td>
</tr>
<tr>
<td>672K A8000</td>
<td></td>
</tr>
<tr>
<td>688K A8000</td>
<td></td>
</tr>
<tr>
<td>704K B0000</td>
<td>Monochrome</td>
</tr>
<tr>
<td>720K B4000</td>
<td></td>
</tr>
<tr>
<td>736K B8000</td>
<td>Color/Graphics</td>
</tr>
<tr>
<td>752K BO000</td>
<td></td>
</tr>
<tr>
<td>768K C0000</td>
<td></td>
</tr>
<tr>
<td>784K C4000</td>
<td></td>
</tr>
<tr>
<td>800K C8000</td>
<td>Fixed Disk Control</td>
</tr>
<tr>
<td>816K C0000</td>
<td></td>
</tr>
<tr>
<td>832K D0000</td>
<td>192K Read Only Memory</td>
</tr>
<tr>
<td>848K D4000</td>
<td>Expansion and Control</td>
</tr>
<tr>
<td>864K D8000</td>
<td></td>
</tr>
<tr>
<td>880K D0000</td>
<td></td>
</tr>
<tr>
<td>896K E0000</td>
<td></td>
</tr>
<tr>
<td>912K E4000</td>
<td></td>
</tr>
<tr>
<td>928K E8000</td>
<td></td>
</tr>
<tr>
<td>944K E0000</td>
<td></td>
</tr>
<tr>
<td>960K F0000</td>
<td>64K Base System ROM</td>
</tr>
<tr>
<td>976K F4000</td>
<td>BIOS and BASIC</td>
</tr>
<tr>
<td>992K F8000</td>
<td></td>
</tr>
<tr>
<td>1008K FC000</td>
<td></td>
</tr>
</tbody>
</table>

System Memory Map (Part 2 of 2)
System Board Component Diagram

System Board Switch Settings

All system board switch settings for total system memory, number of diskette drives, and type of display are located in "Appendix G: Switch Settings."
I/O Channel

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

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

A 'ready' line is available on the I/O channel to allow operation with slow I/O or memory devices. If the channel's ready line is not activated by an addressed device, all processor-generated memory read and write cycles take four 210-ns clock or 840-ns/byte. All processor-generated I/O read and write cycles require five clocks for a cycle time of 1.05 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 processor. Activating this line results in a Non-Maskable Interrupt (NMI) to the 8088 processor. Memory expansion options use this line to report parity errors.

The I/O channel is repowered to provide sufficient drive to power all eight (J1 through J8) expansion slots, assuming two low-power Schottky (LS) loads per slot. The IBM I/O adapters typically use only one load.

Timing requirements on slot J8 are much stricter than those on slots J1 through J7. Slot J8 also requires the card to provide a signal designating when the card is selected. The following pages describe the system board's I/O channel.
I/O Channel Diagram
# 1/0 Channel Description

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

<table>
<thead>
<tr>
<th>Signal</th>
<th>I/O</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OSC</td>
<td>O</td>
<td>Oscillator: High-speed clock with a 70-ns period (14.31818 MHz). It has a 50% duty cycle.</td>
</tr>
<tr>
<td>CLK</td>
<td>O</td>
<td>System clock: It is a divide-by-three of the oscillator and has a period of 210 ns (4.77 MHz). The clock has a 33% duty cycle.</td>
</tr>
<tr>
<td>RESET DRV</td>
<td>O</td>
<td>This line is used to reset or initialize system logic upon power-up or during a low line voltage outage. This signal is synchronized to the falling edge of clock and is active high.</td>
</tr>
<tr>
<td>AO-A19</td>
<td>O</td>
<td>Address bits 0 to 19: These lines are used to address memory and I/O devices within the system. The 20 address lines allow access of up to 1 megabyte of memory. A0 is the least significant bit (LSB) and A19 is the most significant bit (MSB). These lines are generated by either the processor or DMA controller. They are active high.</td>
</tr>
<tr>
<td>D0-D7</td>
<td>I/O</td>
<td>Data Bits 0 to 7: These lines provide data bus bits 0 to 7 for the processor, memory, and I/O devices. D0 is the least significant bit (LSB) and D7 is the most significant bit (MSB). These lines are active high.</td>
</tr>
<tr>
<td>ALE</td>
<td>O</td>
<td>Address Latch Enable: This line is provided by the 8288 Bus Controller and is used on the system board to latch valid addresses from the processor. It is available to the I/O channel as an indicator of a valid processor address (when used with AEN). Processor addresses are latched with the failing edge of ALE.</td>
</tr>
<tr>
<td>Signal</td>
<td>1/0</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-----</td>
<td>-------------</td>
</tr>
<tr>
<td>I/O CHK</td>
<td>I</td>
<td>I/O Channel Check: This line provides the processor with parity (error) information on memory or devices in the I/O channel. When this signal is active low, a parity error is indicated.</td>
</tr>
<tr>
<td>I/O CH RDY</td>
<td>I</td>
<td>I/O Channel Ready: This line, normally high (ready), is pulled low (not ready) by a memory or I/O device to lengthen I/O or memory cycles. It allows slower devices to attach to the I/O channel with a minimum of difficulty. Any slow device using this line should drive it low immediately upon detecting a valid address and a read or write command. This line should never be held low longer than 10 clock cycles. Machine cycles (I/O or memory) are extended by an integral number of CLK cycles (210 ns).</td>
</tr>
<tr>
<td>IRQ2-IRQ7</td>
<td>I</td>
<td>Interrupt Request 2 to 7: These lines are used to signal the processor that an I/O device requires attention. They are prioritized with IRQ2 as the highest priority and IRQ7 as the lowest. An Interrupt Request is generated by raising an IRQ line (low to high) and holding it high until it is acknowledged by the processor (interrupt service routine).</td>
</tr>
<tr>
<td>IOR</td>
<td>O</td>
<td>I/O Read Command: This command line instructs an I/O device to drive its data onto the data bus. It may be driven by the processor or the DMA controller. This signal is active low.</td>
</tr>
<tr>
<td>IOW</td>
<td>O</td>
<td>I/O Write Command: This command line instructs an I/O device to read the data on the data bus. It may be driven by the processor or the DMA controller. This signal is active low.</td>
</tr>
<tr>
<td>Signal</td>
<td>I/O</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-----</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>MEMR</td>
<td>O</td>
<td>Memory Read Command: This command line instructs the memory to drive its data onto the data bus. It may be driven by the processor or the DMA controller. This signal is active low.</td>
</tr>
<tr>
<td>MEMW</td>
<td>O</td>
<td>Memory Write Command: This command line instructs the memory to store the data present on the data bus. It may be driven by the processor or the DMA controller. This signal is active low.</td>
</tr>
<tr>
<td>DRQ1-DRQ3</td>
<td>I</td>
<td>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.</td>
</tr>
<tr>
<td>DACKO-DACK3</td>
<td>O</td>
<td>DMA Acknowledge 0 to 3: These lines are used to acknowledge DMA requests (DRQ1-DRQ3) and to refresh system dynamic memory (DACK0). They are active low.</td>
</tr>
<tr>
<td>AEN</td>
<td>O</td>
<td>Address Enable: This line is used to de-gate the processor and other devices from the I/O channel to allow DMA transfers to take place. When this line is active (high), the DMA controller has control of the address bus, data bus, read command lines (memory and I/O), and the write command lines (memory and I/O).</td>
</tr>
<tr>
<td>TIC</td>
<td>O</td>
<td>Terminal Count: This line provides a pulse when the terminal count for any DMA channel is reached. This signal is active high.</td>
</tr>
<tr>
<td>Signal</td>
<td>I/O Description</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>CARD SLCTD</td>
<td>-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.</td>
<td></td>
</tr>
</tbody>
</table>

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

- $+5 \text{ Vdc } \pm 5\%$, located on 2 connector pins
- $-5 \text{ Vdc } \pm 10\%$, located on 1 connector pin
- $+12 \text{ Vdc } \pm 5\%$, located on 1 connector pin
- $-12 \text{ Vdc } \pm 10\%$, located on 1 connector pin
- GND (Ground), located on 3 connector pins
**Speaker Interface**

The sound system has a small, permanent-magnet, 2-¼ inch speaker. The speaker can be driven from one or both of two sources:

- An 8255A-5 PPI output bit. The address and bit are defined in the "I/O Address Map."

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

---

**Speaker Drive System Block Diagram**

---

**Speaker Tone Generation**

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

---

<table>
<thead>
<tr>
<th>Pin</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Data</td>
</tr>
<tr>
<td>2</td>
<td>Key</td>
</tr>
<tr>
<td>3</td>
<td>Ground</td>
</tr>
<tr>
<td>4</td>
<td>+5 Volts</td>
</tr>
</tbody>
</table>

---

**Speaker Connector**

1-20  System Unit
Power Supply

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

The system board takes approximately 2 to 4 A of +5 Vdc, thus allowing approximately 11 A of +5 Vdc for the adapters in the system expansion slots. The +12 Vdc power level is designed to power the internal 5-1/4 inch diskette drive and the 10 M fixed disk drive. The −5 Vdc level is used for analog circuits in the diskette adapter phase lock loop. The +12 Vdc and −12 Vdc are used for powering the EIA drivers for the communications adapters. All four power levels are bussed across the eight system expansion slots.

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

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

<table>
<thead>
<tr>
<th>Voltage @ 50/60 Hz</th>
<th>Nominal Vac</th>
<th>Minimum Vac</th>
<th>Maximum Vac</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>110</td>
<td>90</td>
<td>137</td>
</tr>
</tbody>
</table>

Input Requirements

Frequency: 50/60 Hz +/- 3 Hz

Current: 4.1 A max @ 90 Vac

<table>
<thead>
<tr>
<th>Voltage (Vdc)</th>
<th>Current (Amps)</th>
<th>Regulation (Tolerance)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Nominal</td>
<td>Minimum</td>
</tr>
<tr>
<td>+5.0</td>
<td>2.3</td>
<td>15.0</td>
</tr>
<tr>
<td>-5.0</td>
<td>0.0</td>
<td>0.3</td>
</tr>
<tr>
<td>+12.0</td>
<td>0.4</td>
<td>4.2</td>
</tr>
<tr>
<td>-12.0</td>
<td>0.0</td>
<td>0.25</td>
</tr>
</tbody>
</table>

Vdc Output

<table>
<thead>
<tr>
<th>Voltage (Vac)</th>
<th>Current (Amps)</th>
<th>Voltage Limits (Vac)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Nominal</td>
<td>Minimum</td>
</tr>
<tr>
<td>120</td>
<td>0.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Vac Output
Power Supply Connectors and Pin Assignments

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

5.25 inch Diskette Drive Power Connector
Fixed Disk Drive Power Connector

Power Supply and Connectors
Over-Voltage/Over-Current Protection

<table>
<thead>
<tr>
<th>Voltage Nominal Vac</th>
<th>Type Protection</th>
<th>Rating Amps</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>Fuse</td>
<td>5</td>
</tr>
</tbody>
</table>

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

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

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

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

The sense levels of the dc outputs are:

<table>
<thead>
<tr>
<th>Output (Vdc)</th>
<th>Minimum (Vdc)</th>
<th>Sense Voltage Nominal (Vdc)</th>
<th>Maximum (Vdc)</th>
</tr>
</thead>
<tbody>
<tr>
<td>+5</td>
<td>+4.5</td>
<td>+5.0</td>
<td>+5.5</td>
</tr>
<tr>
<td>-5</td>
<td>-4.3</td>
<td>-5.0</td>
<td>-5.5</td>
</tr>
<tr>
<td>+12</td>
<td>+10.8</td>
<td>+12.0</td>
<td>+13.2</td>
</tr>
<tr>
<td>-12</td>
<td>-10.2</td>
<td>-12.0</td>
<td>-13.2</td>
</tr>
</tbody>
</table>
IBM Personal Computer Math Coprocessor

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

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

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

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

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

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

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

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Bits</th>
<th>Significant Digits (Decimal)</th>
<th>Approximate Range (decimal)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word Integer</td>
<td>16</td>
<td>4</td>
<td>(-32,768 \leq X \leq 32,767)</td>
</tr>
<tr>
<td>Short Integer</td>
<td>32</td>
<td>9</td>
<td>(-2 \times 10^9 \leq X \leq 2 \times 10^9)</td>
</tr>
<tr>
<td>Long Integer</td>
<td>64</td>
<td>18</td>
<td>(-9 \times 10^{18} \leq X \leq 9 \times 10^{18})</td>
</tr>
<tr>
<td>Packed Decimal</td>
<td>80</td>
<td>18</td>
<td>(-9.9\ldots9 \leq X \leq +9.9\ldots9) (18 digits)</td>
</tr>
<tr>
<td>Short Real*</td>
<td>32</td>
<td>6-7</td>
<td>(8.43 \times 10^{-37} \leq</td>
</tr>
<tr>
<td>Long Real*</td>
<td>64</td>
<td>15-16</td>
<td>(4.19 \times 10^{-307} \leq</td>
</tr>
<tr>
<td>Temporary Real</td>
<td>80</td>
<td>19</td>
<td>(3.4 \times 10^{-4932} \leq</td>
</tr>
</tbody>
</table>

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

Data Types

1-26 Coprocessor
Hardware Interface

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

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

1. Exception and Interrupt Enable bits of the control word are set to 1's.

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

3. NMI Mask REG is set to zero.

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

Because a memory parity error may also cause an interrupt to the 8088 NMI line, the program should check that a parity error did not occur (by reading the 8255 port), then clear exceptions by executing the FNSAVE or the FNCLEX instruction. In most cases, the status word would be looked at, and the exception would be identified and acted upon.
The NMI Mask REG and the coprocessors interrupt are tied to the NMI line through the NMI interrupt logic. Minor conversions of software designed for use with an 8087 must be made before existing software will be compatible with the IBM Personal Computer Math Coprocessor.

Coprocessor Interconnection
Control Unit

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

Coprocessor Block Diagram
Register Stack

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

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

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

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

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

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

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

![Status Word Format](image)

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

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

<table>
<thead>
<tr>
<th>Exception Masks (1 = Exception is Masked)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invalid Operation</td>
</tr>
<tr>
<td>Denormalized Operand</td>
</tr>
<tr>
<td>Zero Divide</td>
</tr>
<tr>
<td>Overflow</td>
</tr>
<tr>
<td>Underflow</td>
</tr>
<tr>
<td>Precision</td>
</tr>
<tr>
<td>(Reserved)</td>
</tr>
<tr>
<td>Interrupt-Enable Mask (1)</td>
</tr>
<tr>
<td>Precision Control(2)</td>
</tr>
<tr>
<td>Rounding Control(3)</td>
</tr>
<tr>
<td>Infinity Control(4)</td>
</tr>
<tr>
<td>(Reserved)</td>
</tr>
</tbody>
</table>

(1) Interrupt-Enable Mask:
- 0 = Interrupts Enabled
- 1 = Interrupts Disabled (Masked)

(2) Precision Control:
- 00 = 24 bits
- 01 = (reserved)
- 10 = 53 bits
- 11 = 64 bits

(3) Rounding Control:
- 00 = Round to Nearest or Even
- 01 = Round Down (toward \( \infty \))
- 10 = Round Up (toward \( \infty \))
- 11 = Chop (Truncate Toward Zero)

(4) Infinity Control:
- 0 = Projective
- 1 = Affine

Control Word Format
Tag Word

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

\[
\begin{array}{cccccccc}
15 & 14 & 13 & 12 & 11 & 10 & 7 & 0 \\
\text{TAG(7)} & \text{TAG(6)} & \text{TAG(5)} & \text{TAG(4)} & \text{TAG(3)} & \text{TAG(2)} & \text{TAG(1)} & \text{TAG(0)}
\end{array}
\]

Tag values:
- 00 = Valid (Normal or Unnormal)
- 01 = Zero (True)
- 10 = Special (Not-A-Number, $\infty$, or Denormal)
- 11 = Empty

Tag Word Format

Exception Pointers

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

\[
\begin{array}{c}
\text{OPERAND ADDRESS}^{(1)} \\
\text{INSTRUCTION OPCODE}^{(2)} \\
\text{INSTRUCTION ADDRESS}^{(1)}
\end{array}
\]

\[
\begin{array}{cc}
10 & 0
\end{array}
\]

(1) 20-bit physical address
(2) 11 least significant bits of opcode: 5 most significant bits are always \text{COPROCESSOR HOOK (11011B)}

Exception Pointers Format

Coprocessor 1-33
Number System

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

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

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

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

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

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

// source/destination, source

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

FADD

FADD source

FADD destination, source

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

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

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>(no operands)</td>
<td>14</td>
<td>10-17</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

FADD

Addition

FADD // source/destination,source

FADDP destination,source

FIADD source

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

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>/ST,ST(i)/ST(i),ST</td>
<td>85</td>
<td>70-100</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>short-real</td>
<td>105+EA</td>
<td>90-120+EA</td>
<td>4</td>
<td>2-4</td>
</tr>
<tr>
<td>long-real</td>
<td>110+EA</td>
<td>95-125+EA</td>
<td>8</td>
<td>2-4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST(i),ST</td>
<td>90</td>
<td>75-105</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

1-36 Coprocessor
FIADD

Exceptions: I, D, O, P

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Typical</td>
<td>Range</td>
<td></td>
<td></td>
</tr>
<tr>
<td>word-integer</td>
<td>120+EA</td>
<td>102-137+EA</td>
<td>2</td>
<td>FIADD DISTANCE_TRAVELLED</td>
</tr>
<tr>
<td>short-integer</td>
<td>125+EA</td>
<td>108-143+EA</td>
<td>4</td>
<td>FIADD PULSE_COUNT[SI]</td>
</tr>
</tbody>
</table>

FBLD

FBLD Source

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

<table>
<thead>
<tr>
<th>FBLD</th>
<th>Exceptions: I</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>packed-decimal</td>
<td>300+EA</td>
</tr>
</tbody>
</table>

FBSTP

FBSTP destination

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

<table>
<thead>
<tr>
<th>FBSTP</th>
<th>Exceptions: I</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>packed-decimal</td>
<td>530+EA</td>
</tr>
</tbody>
</table>
FCHS

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

<table>
<thead>
<tr>
<th>FCHS (no operands)</th>
<th>Exceptions: I</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>15</td>
</tr>
</tbody>
</table>

FCLEX/FNCLEX

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

<table>
<thead>
<tr>
<th>FCLEX/FNCLEX (no operands)</th>
<th>Exceptions: None</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>5</td>
</tr>
</tbody>
</table>

FCOM

FCOM/ /source

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

<table>
<thead>
<tr>
<th>FCOM</th>
<th>Exceptions: I, D</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td>//ST(i)</td>
<td>Typical</td>
</tr>
<tr>
<td>short-real</td>
<td>45</td>
</tr>
<tr>
<td>long-real</td>
<td>65+EA</td>
</tr>
<tr>
<td></td>
<td>70+EA</td>
</tr>
</tbody>
</table>

1-38 Coprocessor
<table>
<thead>
<tr>
<th>C3</th>
<th>C0</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>ST &gt; source</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>ST &lt; source</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>ST = source</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>ST ? source</td>
</tr>
</tbody>
</table>

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

**FCOMP**

**FCOMP/ /source**

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

<table>
<thead>
<tr>
<th>FCOMP</th>
<th>Exceptions: I, D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>//ST(i)</td>
<td>47</td>
</tr>
<tr>
<td>short-real</td>
<td>68+EA</td>
</tr>
<tr>
<td>long-real</td>
<td>72+EA</td>
</tr>
</tbody>
</table>

**FCOMPP**

**FCOMPP/ /source**

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

<table>
<thead>
<tr>
<th>FCOMPP (no operands)</th>
<th>Exceptions: I, D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>50</td>
</tr>
</tbody>
</table>
FDECSTP

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

<table>
<thead>
<tr>
<th>FDECSTP (no operands)</th>
<th>Exceptions: None</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td>(no operands)</td>
<td>Typical 9</td>
</tr>
</tbody>
</table>

FDISI/FNDISI

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

<table>
<thead>
<tr>
<th>FDISI/FNDISI (no operands)</th>
<th>Exceptions: None</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td>(no operands)</td>
<td>Typical 5</td>
</tr>
</tbody>
</table>
FDIV

Normal division

FDIV / /source/ destination,source

FDIVP destination,source

FIDIV source

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

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Typical</td>
<td>Range</td>
<td></td>
<td></td>
</tr>
<tr>
<td>//ST(i),ST</td>
<td>198</td>
<td>193-203</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>short-real</td>
<td>220+EA</td>
<td>215-225+EA</td>
<td>4</td>
<td>2-4</td>
</tr>
<tr>
<td>long-real</td>
<td>225+EA</td>
<td>220-230+EA</td>
<td>8</td>
<td>2-4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Typical</td>
<td>Range</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST(i),ST</td>
<td>202</td>
<td>197-207</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Typical</td>
<td>Range</td>
<td></td>
<td></td>
</tr>
<tr>
<td>word-integer</td>
<td>230+EA</td>
<td>224-238+EA</td>
<td>2</td>
<td>2-4</td>
</tr>
<tr>
<td>short-integer</td>
<td>236+EA</td>
<td>230-243+EA</td>
<td>4</td>
<td>2-4</td>
</tr>
</tbody>
</table>
FDIVR

Reversed Division

FDIVR / /source/ destination, source

FDIVRP destination, source

FIDIVR source

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

<table>
<thead>
<tr>
<th>FDIVR</th>
<th>Exceptions: I, D, Z, O, U, P</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Typical</strong></td>
</tr>
<tr>
<td>//ST,ST(i)/ST(i),ST</td>
<td>199</td>
</tr>
<tr>
<td>short-real</td>
<td>221+EA</td>
</tr>
<tr>
<td>long-real</td>
<td>226+EA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FDIVRP</th>
<th>Exceptions: I, D, Z, O, U, P</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Typical</strong></td>
</tr>
<tr>
<td>ST(i),ST</td>
<td>203</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FIDIVR</th>
<th>Exceptions: I, D, Z, O, U, P</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Typical</strong></td>
</tr>
<tr>
<td>word-integer</td>
<td>230+EA</td>
</tr>
<tr>
<td>short-integer</td>
<td>237+EA</td>
</tr>
</tbody>
</table>
FENI/FNENI

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

<table>
<thead>
<tr>
<th>FENI/FNENI (no operands)</th>
<th>Exceptions: None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>5</td>
</tr>
</tbody>
</table>

FFREE

FFREE destination

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

<table>
<thead>
<tr>
<th>FFREE</th>
<th>Exceptions: None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>ST(i)</td>
<td>11</td>
</tr>
</tbody>
</table>

FICOM

FICOM source

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

<table>
<thead>
<tr>
<th>FICOM</th>
<th>Exceptions: I, D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>word-integer</td>
<td>80+EA</td>
</tr>
<tr>
<td>short-integer</td>
<td>85+EA</td>
</tr>
</tbody>
</table>
FICOMP

FICOMP source

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

<table>
<thead>
<tr>
<th>FICOMP</th>
<th>Exceptions: I, D</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>word-integer</td>
<td>82+EA</td>
</tr>
<tr>
<td>short-integer</td>
<td>87+EA</td>
</tr>
</tbody>
</table>

FILD

FILD source

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

<table>
<thead>
<tr>
<th>FILD</th>
<th>Exceptions: I</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>word-integer</td>
<td>50+EA</td>
</tr>
<tr>
<td>short-integer</td>
<td>56+EA</td>
</tr>
<tr>
<td>long-integer</td>
<td>64+EA</td>
</tr>
</tbody>
</table>

FINCSTP

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

<table>
<thead>
<tr>
<th>FINCSTP (no operands)</th>
<th>Exceptions: None</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>9</td>
</tr>
</tbody>
</table>

1-44  Coprocessor
FINIT/FNINIT

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

<table>
<thead>
<tr>
<th>FINIT/FNINIT (no operands)</th>
<th>Exceptions: None</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Word</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Infinity Control</td>
<td>0</td>
<td>Projective</td>
</tr>
<tr>
<td>Rounding Control</td>
<td>00</td>
<td>Round to nearest</td>
</tr>
<tr>
<td>Precision Control</td>
<td>11</td>
<td>64 bits</td>
</tr>
<tr>
<td>Interrupt-enable Mask</td>
<td>1</td>
<td>Interrupts disabled</td>
</tr>
<tr>
<td>Exception Masks</td>
<td>11111</td>
<td>All exceptions masked</td>
</tr>
<tr>
<td>Status Word</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Busy</td>
<td>0</td>
<td>Not Busy</td>
</tr>
<tr>
<td>Condition Code</td>
<td>???</td>
<td>(Indeterminate)</td>
</tr>
<tr>
<td>Stack Top</td>
<td>000</td>
<td>Empty stack</td>
</tr>
<tr>
<td>Interrupt Request</td>
<td>0</td>
<td>No interrupt</td>
</tr>
<tr>
<td>Exception Flags</td>
<td>00000</td>
<td>No exceptions</td>
</tr>
<tr>
<td>Tag Word</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tags</td>
<td>11</td>
<td>Empty</td>
</tr>
<tr>
<td>Registers</td>
<td>N.C.</td>
<td>Not changed</td>
</tr>
<tr>
<td>Exception Pointers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instruction Code</td>
<td>N.C.</td>
<td>Not changed</td>
</tr>
<tr>
<td>Instruction Address</td>
<td>N.C.</td>
<td>Not changed</td>
</tr>
<tr>
<td>Operand Address</td>
<td>N.C.</td>
<td>Not changed</td>
</tr>
</tbody>
</table>

Coprocessor 1-45
FIST

FIST destination

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

<table>
<thead>
<tr>
<th>FIST</th>
<th>Exceptions: I, P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>word-integer</td>
<td>86+EA</td>
</tr>
<tr>
<td>short-integer</td>
<td>88+EA</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

FISTP

FISTP destination

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

<table>
<thead>
<tr>
<th>FISTP</th>
<th>Exceptions: I, P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>word-integer</td>
<td>88+EA</td>
</tr>
<tr>
<td>short-integer</td>
<td>90+EA</td>
</tr>
<tr>
<td>long-integer</td>
<td>100+EA</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1-46 Coprocessor
FLD

FLD source

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

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Typical</td>
<td>Range 17-22</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>ST(i)</td>
<td>20</td>
<td>38-55+EA</td>
<td>4</td>
<td>2-4</td>
</tr>
<tr>
<td>short-real</td>
<td>43+EA</td>
<td>40-60+EA</td>
<td>8</td>
<td>2-4</td>
</tr>
<tr>
<td>long-real</td>
<td>46+EA</td>
<td>53-65+EA</td>
<td>10</td>
<td>2-4</td>
</tr>
<tr>
<td>temp-real</td>
<td>57+EA</td>
<td>53-65+EA</td>
<td>10</td>
<td>2-4</td>
</tr>
</tbody>
</table>

FLDCW

FLDCW source

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

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Typical</td>
<td>Range 10+EA</td>
<td>2</td>
<td>2-4</td>
</tr>
<tr>
<td>2-bytes</td>
<td></td>
<td>7-14+EA</td>
<td>2</td>
<td>FLDCW CONTROL WORD</td>
</tr>
</tbody>
</table>

Coprocessor 1-47
FLDENV

FLDENV source

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

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Typical</td>
<td>Typical</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14-bytes</td>
<td>40+EA</td>
<td>35-45+EA</td>
<td>14</td>
<td>2-4</td>
</tr>
</tbody>
</table>

FLDLG2

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

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>(no operands)</td>
<td>21</td>
<td>18-24</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

FLDLN2

FLDLN2 (load log base e of 2) loads (pushes) the value of LOG\textsubscript{2}2 onto the stack.

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>(no operands)</td>
<td>20</td>
<td>17-23</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>
FLDL2E

FLDL2E (load log base 2 of e) loads (pushes) the value \( \log_2 e \) onto the stack.

<table>
<thead>
<tr>
<th>FLDL2E (no operands)</th>
<th>Exceptions: 1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>18</td>
</tr>
</tbody>
</table>

FLDL2T

FLDL2T (load log base 2 of 10) loads (pushes) the value \( \log_2 10 \) onto the stack.

<table>
<thead>
<tr>
<th>FLDL2T (no operands)</th>
<th>Exceptions: 1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>19</td>
</tr>
</tbody>
</table>

FLDPI

FLDPI (load \( \pi \)) loads (pushes) \( \pi \) onto the stack.

<table>
<thead>
<tr>
<th>FLDPI (no operands)</th>
<th>Exceptions: 1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>19</td>
</tr>
</tbody>
</table>
FLDZ

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

<table>
<thead>
<tr>
<th>FLDZ (no operands)</th>
<th>Exceptions: 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>14</td>
</tr>
</tbody>
</table>

FLD1

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

<table>
<thead>
<tr>
<th>FLD1 (no operands)</th>
<th>Exceptions: 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>18</td>
</tr>
</tbody>
</table>
**FMUL**

Multiplication

FMUL / /source/destination,source  
FMULP destination,source  
FIMUL source

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

<table>
<thead>
<tr>
<th>FMUL</th>
<th>Exceptions: I, D, O, U, P</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>ST(i),ST,ST,ST(i)</td>
<td>97</td>
</tr>
<tr>
<td>ST(i),ST,ST,ST(i)</td>
<td>138</td>
</tr>
<tr>
<td>short-real</td>
<td>118+EA</td>
</tr>
<tr>
<td>long-real</td>
<td>120+EA</td>
</tr>
<tr>
<td>long-real</td>
<td>151+EA</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>FMULP</th>
<th>Exceptions: I, D, O, U, P</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>ST(i),ST</td>
<td>100</td>
</tr>
<tr>
<td>ST(i),ST</td>
<td>142</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>FIMUL</th>
<th>Exceptions: I, D, O, P</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>word-integer</td>
<td>130+EA</td>
</tr>
<tr>
<td>short-integer</td>
<td>136+EA</td>
</tr>
</tbody>
</table>
FNOP

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

<table>
<thead>
<tr>
<th>FNOP (no operands)</th>
<th>Exceptions: None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>13</td>
</tr>
</tbody>
</table>

FPATAN

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

<table>
<thead>
<tr>
<th>FPATAN (no operands)</th>
<th>Exceptions: U, P (operands not checked)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>650</td>
</tr>
</tbody>
</table>

FPREM

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

<table>
<thead>
<tr>
<th>FPREM (no operands)</th>
<th>Exceptions: I, D, U</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>125</td>
</tr>
</tbody>
</table>

1-52  Coprocessor
FPTAN

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

<table>
<thead>
<tr>
<th>FPTAN</th>
<th>Exceptions: I, P (operands not checked)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>450</td>
</tr>
</tbody>
</table>

FRNDINT

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

<table>
<thead>
<tr>
<th>FRNDINT (no operands)</th>
<th>Exceptions: I, P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>45</td>
</tr>
</tbody>
</table>

FRSTOR

FRSTOR source

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

<table>
<thead>
<tr>
<th>FRSTOR</th>
<th>Exceptions: None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>94-bytes</td>
<td>210+EA</td>
</tr>
</tbody>
</table>
FSAVE/FNSAVE

FSAVE/FNSAVE destination

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

<table>
<thead>
<tr>
<th>FSAVE/FNSAVE</th>
<th>Exceptions: None</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Typical</strong></td>
</tr>
<tr>
<td>94-bytes</td>
<td>210+EA</td>
</tr>
</tbody>
</table>

FScale

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

\[ ST - ST \cdot 2^{ST(1)} \]

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

<table>
<thead>
<tr>
<th>FSCALE (no operands)</th>
<th>Exceptions: I, O, U</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Typical</strong></td>
</tr>
<tr>
<td>(no operands)</td>
<td>35</td>
</tr>
</tbody>
</table>
FSQRT

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

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

<table>
<thead>
<tr>
<th>FSQRT (no operands)</th>
<th>Exceptions: I, D, P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>183</td>
</tr>
</tbody>
</table>

FST

FST destination

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

<table>
<thead>
<tr>
<th>FST</th>
<th>Exceptions: I, O, U, P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>ST(i)</td>
<td>18</td>
</tr>
<tr>
<td>short-real</td>
<td>87+EA</td>
</tr>
<tr>
<td>long-real</td>
<td>100+EA</td>
</tr>
</tbody>
</table>
FSTCW/FNSTCW

FSTCW/FNSTCW destination

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

<table>
<thead>
<tr>
<th>FSTCW/FNSTCW</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Typical Range</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bytes</td>
<td>15+EA 12-18+EA</td>
<td>4</td>
<td>2-4</td>
<td>FSTCW SAVE_CONTROL</td>
</tr>
</tbody>
</table>

FSTENV/FNSTENV

FSTENV/FNSTENV destination

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

<table>
<thead>
<tr>
<th>FSTENV/FNSTENV</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Typical Range</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14-bytes</td>
<td>45+EA 40-50+EA</td>
<td>16</td>
<td>2-4</td>
<td>FSTENV [8P]</td>
</tr>
</tbody>
</table>
FSTP

FSTP destination

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

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Typical</td>
<td>Range</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST(i)</td>
<td>20</td>
<td>17-24</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>short-real</td>
<td>89+EA</td>
<td>86-92+EA</td>
<td>6</td>
<td>2.4</td>
</tr>
<tr>
<td>long-real</td>
<td>102+EA</td>
<td>98-106+EA</td>
<td>10</td>
<td>2.4</td>
</tr>
<tr>
<td>temp-real</td>
<td>55+EA</td>
<td>52-58+EA</td>
<td>12</td>
<td>2.4</td>
</tr>
</tbody>
</table>

FSTSW/FNSTSW

FSTSW/FNSTSW destination

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

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Typical</td>
<td>Range</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-bytes</td>
<td>14+EA</td>
<td>12-18+EA</td>
<td>4</td>
<td>2.4</td>
</tr>
</tbody>
</table>

FSTSW SAVE_STATUS
FSUB

Subtraction

FSUB / /source/destination,source

FSUBP destination,source

FISUB source

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

### FSUB

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Typical</td>
<td>Range</td>
<td></td>
<td></td>
</tr>
<tr>
<td>//ST,ST(i)/ST(i),ST</td>
<td>85</td>
<td>70-100</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>short-real</td>
<td>105+EA</td>
<td>90-120+EA</td>
<td>4</td>
<td>2-4</td>
</tr>
<tr>
<td>long-real</td>
<td>110+EA</td>
<td>95-125+EA</td>
<td>8</td>
<td>2-4</td>
</tr>
</tbody>
</table>

### FSUBP

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Typical</td>
<td>Range</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ST(i),ST</td>
<td>90</td>
<td>75-105</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

### FISUB

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Typical</td>
<td>Range</td>
<td></td>
<td></td>
</tr>
<tr>
<td>word-integer</td>
<td>120+EA</td>
<td>102-137+EA</td>
<td>2</td>
<td>2-4</td>
</tr>
<tr>
<td>short-integer</td>
<td>125+EA</td>
<td>108-143+EA</td>
<td>4</td>
<td>2-4</td>
</tr>
</tbody>
</table>

1-58 Coprocessor
FSUBR
Reversed Subtraction
FSUBR / /source/destination,source
FSUBRP destination,source
FISUBR source

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

<table>
<thead>
<tr>
<th>FSUBR</th>
<th>Exceptions: I, D, O, U, P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>//ST,ST(i)/ST(i),ST</td>
<td>87</td>
</tr>
<tr>
<td>short-real</td>
<td>105+EA</td>
</tr>
<tr>
<td>long-real</td>
<td>110+EA</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FSUBRP</th>
<th>Exceptions: I, D, O, U, P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>ST(i),ST</td>
<td>90</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FISUBR</th>
<th>Exceptions: I, D, O, P</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operands</td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>word-integer</td>
<td>120+EA</td>
</tr>
<tr>
<td>short-integer</td>
<td>125+EA</td>
</tr>
</tbody>
</table>
FTST

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

<table>
<thead>
<tr>
<th>FTST (no operands)</th>
<th>Exceptions: I, D</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Typical</strong></td>
</tr>
<tr>
<td>(no operands)</td>
<td>42</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C3</th>
<th>C0</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>ST is positive and nonzero</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>ST is negative and nonzero</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>ST is zero (+ or -)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>ST is not comparable (that is, it is a NAN or projective ∞)</td>
</tr>
</tbody>
</table>

FWAIT

FWAIT (processor instruction)

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

<table>
<thead>
<tr>
<th>FWAIT (no operands)</th>
<th>Exceptions: Non (CPU instruction)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Typical</strong></td>
</tr>
<tr>
<td>(no operands)</td>
<td>3+5n</td>
</tr>
</tbody>
</table>
FXAM

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

<table>
<thead>
<tr>
<th>FXAM</th>
<th>Exceptions: None</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td><strong>Execution Clocks</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Typical</strong></td>
</tr>
<tr>
<td>(no operands)</td>
<td>17</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Condition Code</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>C3 C2 C1 C0</td>
<td></td>
</tr>
<tr>
<td>0 0 0 0</td>
<td>+ Unnormal</td>
</tr>
<tr>
<td>0 0 0 1</td>
<td>+ NAN</td>
</tr>
<tr>
<td>0 0 1 0</td>
<td>+ Unnormal</td>
</tr>
<tr>
<td>0 0 1 1</td>
<td>+ NaN</td>
</tr>
<tr>
<td>0 1 0 0</td>
<td>+ Normal</td>
</tr>
<tr>
<td>0 1 0 1</td>
<td>+∞</td>
</tr>
<tr>
<td>0 1 1 0</td>
<td>- Normal</td>
</tr>
<tr>
<td>0 1 1 1</td>
<td>-∞</td>
</tr>
<tr>
<td>1 0 0 0</td>
<td>+ 0</td>
</tr>
<tr>
<td>1 0 0 1</td>
<td>Empty</td>
</tr>
<tr>
<td>1 0 1 0</td>
<td>- 0</td>
</tr>
<tr>
<td>1 0 1 1</td>
<td>Empty</td>
</tr>
<tr>
<td>1 1 0 0</td>
<td>+ Denormal</td>
</tr>
<tr>
<td>1 1 0 1</td>
<td>Empty</td>
</tr>
<tr>
<td>1 1 1 0</td>
<td>- Denormal</td>
</tr>
<tr>
<td>1 1 1 1</td>
<td>Empty</td>
</tr>
</tbody>
</table>
FXCH

FXCH/ /destination

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

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>//ST(i)</td>
<td>12 10-15</td>
<td>0</td>
<td>2</td>
<td>FXCH ST(2)</td>
</tr>
</tbody>
</table>

FXTRACT

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

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>(no operands)</td>
<td>50 27-55</td>
<td>0</td>
<td>2</td>
<td>FXTRACT</td>
</tr>
</tbody>
</table>
**FYL2X**

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

**LOG$_n$2•LOG$_2$X**

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>(no operands)</td>
<td>950</td>
<td>900-1100</td>
<td>0</td>
<td>2 FYL2X</td>
</tr>
</tbody>
</table>

**FYL2XP1**

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

<table>
<thead>
<tr>
<th>Operands</th>
<th>Execution Clocks</th>
<th>Transfers 8088</th>
<th>Bytes</th>
<th>Coding Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>(no operands)</td>
<td>850</td>
<td>700-1000</td>
<td>0</td>
<td>2 FYL2XP1</td>
</tr>
</tbody>
</table>
F2XM1

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

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

<table>
<thead>
<tr>
<th>F2XM1</th>
<th>Exceptions: U, P (operands not checked)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operands</strong></td>
<td>Execution Clocks</td>
</tr>
<tr>
<td></td>
<td>Typical</td>
</tr>
<tr>
<td>(no operands)</td>
<td>500</td>
</tr>
</tbody>
</table>

1-64 Coprocessor
IBM Keyboard

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

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

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

The keyboard interface is defined so that system software has maximum flexibility in defining certain keyboard operations. This is accomplished by having the keyboard return scan codes rather than American Standard Code for Information Interchange (ASCII) codes. In addition, all keys are typematic and generate both a make and a break scan code. For example, key 1 produces scan code hex 01 on make and code hex 81 on break. Break codes are formed by adding hex 80 to make codes. The keyboard I/O driver can define keyboard keys as shift keys or typematic, as required by the application.
The microcomputer (Intel 8048) in the keyboard performs several functions, including a power-on self-test when requested by the system unit. This test checks the microcomputer ROM, tests memory, and checks for stuck keys. Additional functions are: keyboard scanning, buffering of up to 16 key scan codes, maintaining bidirectional serial communications with the system unit, and executing the hand-shake protocol required by each scan-code transfer.

The following pages have figures that show the keyboard, the scan codes, and the keyboard interface connector specifications.
Keyboard Interface Block Diagram
Note: Nomenclature is on both the top and front face of the keybutton as shown. The number to the upper left designates the button position.
<table>
<thead>
<tr>
<th>Key Position</th>
<th>Scan Code in Hex</th>
<th>Key Position</th>
<th>Scan Code in Hex</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>01</td>
<td>43</td>
<td>2B</td>
</tr>
<tr>
<td>2</td>
<td>02</td>
<td>44</td>
<td>2C</td>
</tr>
<tr>
<td>3</td>
<td>03</td>
<td>45</td>
<td>2D</td>
</tr>
<tr>
<td>4</td>
<td>04</td>
<td>46</td>
<td>2E</td>
</tr>
<tr>
<td>5</td>
<td>05</td>
<td>47</td>
<td>2F</td>
</tr>
<tr>
<td>6</td>
<td>06</td>
<td>48</td>
<td>30</td>
</tr>
<tr>
<td>7</td>
<td>07</td>
<td>49</td>
<td>31</td>
</tr>
<tr>
<td>8</td>
<td>08</td>
<td>50</td>
<td>32</td>
</tr>
<tr>
<td>9</td>
<td>09</td>
<td>51</td>
<td>33</td>
</tr>
<tr>
<td>10</td>
<td>0A</td>
<td>52</td>
<td>34</td>
</tr>
<tr>
<td>11</td>
<td>0B</td>
<td>53</td>
<td>35</td>
</tr>
<tr>
<td>12</td>
<td>0C</td>
<td>54</td>
<td>36</td>
</tr>
<tr>
<td>13</td>
<td>0D</td>
<td>55</td>
<td>37</td>
</tr>
<tr>
<td>14</td>
<td>0E</td>
<td>56</td>
<td>38</td>
</tr>
<tr>
<td>15</td>
<td>0F</td>
<td>57</td>
<td>39</td>
</tr>
<tr>
<td>16</td>
<td>10</td>
<td>58</td>
<td>3A</td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>59</td>
<td>3B</td>
</tr>
<tr>
<td>18</td>
<td>12</td>
<td>60</td>
<td>3C</td>
</tr>
<tr>
<td>19</td>
<td>13</td>
<td>61</td>
<td>3D</td>
</tr>
<tr>
<td>20</td>
<td>14</td>
<td>62</td>
<td>3E</td>
</tr>
<tr>
<td>21</td>
<td>15</td>
<td>63</td>
<td>3F</td>
</tr>
<tr>
<td>22</td>
<td>16</td>
<td>64</td>
<td>40</td>
</tr>
<tr>
<td>23</td>
<td>17</td>
<td>65</td>
<td>41</td>
</tr>
<tr>
<td>24</td>
<td>18</td>
<td>66</td>
<td>42</td>
</tr>
<tr>
<td>25</td>
<td>19</td>
<td>67</td>
<td>43</td>
</tr>
<tr>
<td>26</td>
<td>1A</td>
<td>68</td>
<td>44</td>
</tr>
<tr>
<td>27</td>
<td>1B</td>
<td>69</td>
<td>45</td>
</tr>
<tr>
<td>28</td>
<td>1C</td>
<td>70</td>
<td>46</td>
</tr>
<tr>
<td>29</td>
<td>1D</td>
<td>71</td>
<td>47</td>
</tr>
<tr>
<td>30</td>
<td>1E</td>
<td>72</td>
<td>48</td>
</tr>
<tr>
<td>31</td>
<td>1F</td>
<td>73</td>
<td>49</td>
</tr>
<tr>
<td>32</td>
<td>20</td>
<td>74</td>
<td>4A</td>
</tr>
<tr>
<td>33</td>
<td>21</td>
<td>75</td>
<td>4B</td>
</tr>
<tr>
<td>34</td>
<td>22</td>
<td>76</td>
<td>4C</td>
</tr>
<tr>
<td>35</td>
<td>23</td>
<td>77</td>
<td>4D</td>
</tr>
<tr>
<td>36</td>
<td>24</td>
<td>78</td>
<td>4E</td>
</tr>
<tr>
<td>37</td>
<td>25</td>
<td>79</td>
<td>4F</td>
</tr>
<tr>
<td>38</td>
<td>26</td>
<td>80</td>
<td>50</td>
</tr>
<tr>
<td>39</td>
<td>27</td>
<td>81</td>
<td>51</td>
</tr>
<tr>
<td>40</td>
<td>28</td>
<td>82</td>
<td>52</td>
</tr>
<tr>
<td>41</td>
<td>29</td>
<td>83</td>
<td>53</td>
</tr>
<tr>
<td>42</td>
<td>2A</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Keyboard Scan Codes**
Rear Panel

Keyboard Connector

5-Pin DIN Connector

<table>
<thead>
<tr>
<th>Pin</th>
<th>TTL Signal</th>
<th>Signal Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+ Keyboard Clock</td>
<td>+5 Vdc</td>
</tr>
<tr>
<td>2</td>
<td>+ Keyboard Data</td>
<td>+5 Vdc</td>
</tr>
<tr>
<td>3</td>
<td>– Keyboard Reset (Not used by keyboard)</td>
<td>+5 Vdc</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Power Supply Voltages</th>
<th>Voltage</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>+5 Vdc</td>
</tr>
</tbody>
</table>

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

Expansion Unit Cable

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

Power Supply

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

Expansion Board

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

Decoupling capacitors provided on the expansion board aid in noise filtering.
Expansion Board Block Diagram

1-72 Expansion Unit
Expansion Channel

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

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

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

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

(*) With respect to the system unit.

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

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

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

The DIP switch on the extender card should be set to indicate the maximum contiguous read/write memory housed in the system unit. The extender card switch settings are located in "Appendix G: Switch Settings." Switch positions 1 through 4 correspond to address bits hex A19 to hex A16, respectively.

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

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

<table>
<thead>
<tr>
<th>Location</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory FXXXX(*)</td>
<td>Write to memory to latch address bits</td>
</tr>
<tr>
<td>Port 210</td>
<td>Write to latch expansion bus data (ED0-ED7)</td>
</tr>
<tr>
<td>Port 210</td>
<td>Read to verify expansion bus data (ED0-ED7)</td>
</tr>
<tr>
<td>Port 211</td>
<td>Read high-order address bits (A8 - A15)</td>
</tr>
<tr>
<td>Port 211</td>
<td>Write to clear wait test latch</td>
</tr>
<tr>
<td>Port 212</td>
<td>Read low-order address bits (A0 - A7)</td>
</tr>
<tr>
<td>Port 213</td>
<td>Write 00 to disable expansion unit</td>
</tr>
<tr>
<td>Port 213</td>
<td>Write 01 to enable expansion unit</td>
</tr>
<tr>
<td>Port 213</td>
<td>Read status of expansion unit</td>
</tr>
<tr>
<td></td>
<td>DO = enable/disable</td>
</tr>
<tr>
<td></td>
<td>D1 = wait-state request flag</td>
</tr>
<tr>
<td></td>
<td>D2-D3 = not used</td>
</tr>
<tr>
<td></td>
<td>D4-D7 = switch position</td>
</tr>
<tr>
<td></td>
<td>1 = Off</td>
</tr>
<tr>
<td></td>
<td>0 = On</td>
</tr>
</tbody>
</table>

(*) Example: Write to memory location F123;4=00
Read Port 211 = 12
Read Port 212 = 34

(All values in hex)

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

1-76 Expansion Unit
Receiver Card

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

Receiver Card Programming Considerations

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

<table>
<thead>
<tr>
<th>Location</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory FXXXX(*)</td>
<td>Write to memory to latch address bits</td>
</tr>
<tr>
<td>Port 214</td>
<td>Write to latch data bus bits (D0 - D7)</td>
</tr>
<tr>
<td>Port 214</td>
<td>Read data bus bits (D0 - D7)</td>
</tr>
<tr>
<td>Port 215</td>
<td>Read high-order address bits (A8 - A15)</td>
</tr>
<tr>
<td>Port 215</td>
<td>Read low-order address bits (A0 - A7)</td>
</tr>
</tbody>
</table>

(*) Example: Write to memory location F123:4=00  
Read Port 215 =12  
Read Port 216 =34

(All values in hex)

The expansion unit is automatically enabled upon power-up. The expansion unit and the system unit will be written to, if the expansion unit is not disabled when writing to FXXXX. However, the system unit and the expansion unit are read back separately.
Receiver Card Block Diagram
Expansion Unit Interface Information

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

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

E = Extended

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

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

The printers can also accept commands setting the line-feed control desired for the application. They attach to the system unit through the printer adapter or the combination monochrome display and printer adapter. The cable is a 25-lead shielded cable with a 25-pin D-shell connector at the system unit end, and a 36-pin connector at the printer end.
| (1) Print Method:       | Serial-impact dot matrix         |
| (2) Print Speed:        | 80 cps                         |
| (3) Print Direction:    | Bidirectional with logical seeking |
| (4) Number of Pins in Head: | 9                            |
| (5) Line Spacing:       | 1/16 inch (4.23 mm) or programmable |
| (6) Printing Characteristics Matrix: | 9 x 9                        |
| Character Set:          | Full 96-character ASCII with descenders plus 9 international characters/symbols. |
| Graphic Character:      | See "Additional Printer Specifications" |
| (7) Printing Sizes:     |                                 |
|                         | Characters per inch | Maximum characters per inch |
|                         | 10                   | 80                         |
|                         | 5                    | 40                         |
|                         | 16.5                 | 132                        |
|                         | 8.25                 | 66                         |
| (8) Media Handling:     | Adjustable sprocket pin feed                                           |
| Paper Feed:             | 4 inch (101.6 mm) to 10 inch (254 mm)                                      |
| Paper Width Range:      | One original plus two carbon copies (total thickness not to exceed 0.012 inch (0.3 mm)). |
| Copies:                 | Minimum paper thickness is 0.0025 inch (0.064 mm). |
| Paper Path:             | Rear                                                                 |
| (9) Interfaces:         | Parallel 8-bit Data and Control Lines                                   |
| Standard:               |                                                        |
| (10) Inked Ribbon:      | Black                                                                  |
| Type:                   | Cartridge                                                              |
| Life Expectancy:        | 3 million characters                                                   |
| (11) Environmental Conditions |                                       |
| Operating Temperature Range: | 41 to 95°F (5 to 35°C)                                      |
| Operating Humidity:     | 10 to 80% non-condensing                                              |
| (12) Power Requirement: |                                                        |
| Voltage:                | 120 Vac, 60 Hz                                                        |
| Current:                | 1 A maximum                                                           |
| Power Consumption:      | 100 VA maximum                                                        |
| (13) Physical Characteristics: |                            |
| Height:                 | 4.2 inches (107 mm)                                                   |
| Width:                  | 14.7 inches (374 mm)                                                  |
| Depth:                  | 12.0 inches (305 mm)                                                  |
| Weight:                 | 12 pounds (5.5 kg)                                                    |

**Printer Specifications**

1-82 Printers
(6) Printing Characteristics:
IBM 80 CPS Matrix Printer
Graphics
IBM 80 CPS Graphics Printer

64 block characters.

(6) Printing Characteristics:
Extra Character Set.

Set 1
Additional ASCII numbers 160 to 175 contain European characters. Numbers 176 to 223 contain graphic characters. Numbers 224 to 239 contain selected Greek characters. Numbers 240 to 255 contain math and extra symbols.

Set 2
The difference in set 2 are ASCII numbers 3, 4, 5, 6, and 21. ASCII numbers 128 to 175 contain European characters.

Graphics
There are 20 block characters and programmable graphics.

(7) Printing Sizes:

<table>
<thead>
<tr>
<th>Characters per inch</th>
<th>Maximum characters per line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subscript:</td>
<td>10</td>
</tr>
<tr>
<td>Superscript:</td>
<td>10</td>
</tr>
</tbody>
</table>

Additional Printer Specifications
Setting the DIP Switches

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

<table>
<thead>
<tr>
<th>Switch Number</th>
<th>Function</th>
<th>On</th>
<th>Off</th>
<th>Factory-Set Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1</td>
<td>Not Applicable</td>
<td>—</td>
<td>—</td>
<td>On</td>
</tr>
<tr>
<td>1-2</td>
<td>CR</td>
<td>Print Only</td>
<td>Print &amp; Line Feed</td>
<td>On</td>
</tr>
<tr>
<td>1-3</td>
<td>Buffer Full</td>
<td>Print Only</td>
<td>Print &amp; Line Feed</td>
<td>Off</td>
</tr>
<tr>
<td>1-4</td>
<td>Cancel Code</td>
<td>Invalid</td>
<td>Valid</td>
<td>Off</td>
</tr>
<tr>
<td>1-5</td>
<td>Delete Code</td>
<td>Invalid</td>
<td>Valid</td>
<td>On</td>
</tr>
<tr>
<td>1-6</td>
<td>Error</td>
<td>Sounds</td>
<td>Does Not Sound</td>
<td>On</td>
</tr>
<tr>
<td>1-7</td>
<td>Character Generator</td>
<td>N.A.</td>
<td>Graphic Patterns Select</td>
<td>Off</td>
</tr>
<tr>
<td></td>
<td>(Graphic Pattern Select)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1-8</td>
<td>SLCT IN Signal Fixed</td>
<td>Fixed</td>
<td>Not Fixed</td>
<td>On</td>
</tr>
<tr>
<td></td>
<td>Internally</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Functions and Conditions of DIP Switch 1 (Matrix)**
<table>
<thead>
<tr>
<th>Switch Number</th>
<th>Function</th>
<th>On</th>
<th>Off</th>
<th>Factory-Set Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>Not Applicable</td>
<td></td>
<td></td>
<td>On</td>
</tr>
<tr>
<td>2-2</td>
<td>Not Applicable</td>
<td></td>
<td></td>
<td>On</td>
</tr>
<tr>
<td>2-3</td>
<td>Auto Feed XT Signal</td>
<td>Fixed</td>
<td>Not Fixed</td>
<td>Off</td>
</tr>
<tr>
<td></td>
<td>Internally</td>
<td>Internally</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-4</td>
<td>Coding Table Select</td>
<td>N.A.</td>
<td>Standard</td>
<td>Off</td>
</tr>
</tbody>
</table>

**Functions and Conditions of DIP Switch 2 (Matrix)**

<table>
<thead>
<tr>
<th>Switch Number</th>
<th>Function</th>
<th>On</th>
<th>Off</th>
<th>Factory-Set Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1</td>
<td>Not Applicable</td>
<td></td>
<td></td>
<td>On</td>
</tr>
<tr>
<td>1-2</td>
<td>CR</td>
<td>Print Only</td>
<td>Print &amp; Line Feed</td>
<td>On</td>
</tr>
<tr>
<td>1-3</td>
<td>Buffer Full</td>
<td>Print Only</td>
<td>Print &amp; Line Feed</td>
<td>Off</td>
</tr>
<tr>
<td>1-4</td>
<td>Cancel Code</td>
<td>Invalid</td>
<td>Valid</td>
<td>Off</td>
</tr>
<tr>
<td>1-5</td>
<td>Not Applicable</td>
<td></td>
<td></td>
<td>On</td>
</tr>
<tr>
<td>1-6</td>
<td>Error Buzzer</td>
<td>Sound</td>
<td>Does Not Sound</td>
<td>On</td>
</tr>
<tr>
<td>1-7</td>
<td>Character Generator</td>
<td>Set 2</td>
<td>Set 1</td>
<td>Off</td>
</tr>
<tr>
<td>1-8</td>
<td>SLCT IN Signal</td>
<td>Fixed</td>
<td>Not Fixed</td>
<td>On</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Internally</td>
<td>Internally</td>
<td></td>
</tr>
</tbody>
</table>

**Functions and Conditions of DIP Switch 1 (Graphics)**

<table>
<thead>
<tr>
<th>Switch Number</th>
<th>Function</th>
<th>On</th>
<th>Off</th>
<th>Factory-Set Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>Form Length</td>
<td>12 Inches</td>
<td>11 Inches</td>
<td>Off</td>
</tr>
<tr>
<td>2-2</td>
<td>Line Spacing</td>
<td>1/8 Inch</td>
<td>1/6 Inch</td>
<td>Off</td>
</tr>
<tr>
<td>2-3</td>
<td>Auto Feed XT Signal</td>
<td>Fixed</td>
<td>Not Fixed</td>
<td>Off</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Internally</td>
<td>Internally</td>
<td></td>
</tr>
<tr>
<td>2-4</td>
<td>1 Inch Skip Over Perforation</td>
<td>Valid</td>
<td>Not Valid</td>
<td>Off</td>
</tr>
</tbody>
</table>

**Functions and Conditions of DIP Switch 2 (Graphics)**

Printers 1-85
Parallel Interface Description

Specifications:

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

Connector: Plug: 57-30360 (Amphenol)

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

Data transfer sequence:

[Diagram showing timing waveforms for BUSY, ACKNLG, DATA, and STROBE signals with time durations indicated: 0.5 µs (Minimum) and 5 µs for certain transitions.]

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

Connector Pin Assignment and Descriptions of Interface Signals (Part 1 of 3)
<table>
<thead>
<tr>
<th>Signal Pin No.</th>
<th>Return Pin No.</th>
<th>Signal</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>30</td>
<td>PE</td>
<td>Out</td>
<td>A &quot;high&quot; signal indicates that the printer is out of paper.</td>
</tr>
<tr>
<td>13</td>
<td></td>
<td>SLCT</td>
<td>Out</td>
<td>This signal indicates that the printer is in the selected state.</td>
</tr>
<tr>
<td>14</td>
<td></td>
<td>AUTO</td>
<td>In</td>
<td>With this signal being at &quot;low&quot; level, the paper is automatically fed one line after printing. (The signal level can be fixed to &quot;low&quot; with DIP SW pin 2-3 provided on the control circuit board.)</td>
</tr>
<tr>
<td>15</td>
<td></td>
<td>NC</td>
<td></td>
<td>Not used.</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>OV</td>
<td></td>
<td>Logic GND level.</td>
</tr>
<tr>
<td>17</td>
<td></td>
<td>CHASSIS-GND</td>
<td></td>
<td>Printer chassis GND. In the printer, the chassis GND and the logic GND are isolated from each other.</td>
</tr>
<tr>
<td>18</td>
<td></td>
<td>NC</td>
<td></td>
<td>Not used.</td>
</tr>
<tr>
<td>19-30</td>
<td></td>
<td>GND</td>
<td></td>
<td>&quot;Twisted-Pair Return&quot; signal; GND level.</td>
</tr>
<tr>
<td>31</td>
<td></td>
<td>INT</td>
<td>In</td>
<td>When the level of this signal becomes &quot;low&quot; the printer controller is reset to its initial state and the print buffer is cleared. This signal is normally at &quot;high&quot; level, and its pulse width must be more than 50 µs at the receiving terminal.</td>
</tr>
</tbody>
</table>

Connector Pin Assignment and Descriptions of Interface Signals (Part 2 of 3)
<table>
<thead>
<tr>
<th>Signal Pin No.</th>
<th>Return Pin No.</th>
<th>Signal</th>
<th>Direction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td></td>
<td>ERROR</td>
<td>Out</td>
<td>The level of this signal becomes &quot;low&quot; when the printer is in &quot;Paper End&quot; state, &quot;Offline&quot; state and &quot;Error&quot; state.</td>
</tr>
<tr>
<td>33</td>
<td>—</td>
<td>GND</td>
<td>—</td>
<td>Same as with pin numbers 19 to 30.</td>
</tr>
<tr>
<td>34</td>
<td>—</td>
<td>NC</td>
<td>—</td>
<td>Not used.</td>
</tr>
<tr>
<td>35</td>
<td></td>
<td></td>
<td></td>
<td>Pulled up to +5 Vdc through 4.7 k-ohms resistance.</td>
</tr>
<tr>
<td>36</td>
<td>—</td>
<td>SLCT IN</td>
<td>In</td>
<td>Data entry to the printer is possible only when the level of this signal is &quot;low&quot;. (Internal fixing can be carried out with DIP SW 1-8. The condition at the time of shipment is set &quot;low&quot; for this signal.)</td>
</tr>
</tbody>
</table>

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

**Connector Pin Assignment and Descriptions of Interface Signals (Part 3 of 3)**
Printer Modes for the IBM 80 CPS Printers

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

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

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

<table>
<thead>
<tr>
<th>Printer Modes</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Compressed</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Emphasized</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Double Strike</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Subscript</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Superscript</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Double Width</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Underline</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

1-90 Printers
Printer Control Codes

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

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

The descriptions that follow assume that the printer DIP switches have not been changed from their factory settings.
<table>
<thead>
<tr>
<th>Printer Code</th>
<th>Printer Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUL</td>
<td>Null</td>
</tr>
<tr>
<td></td>
<td>Used with ESC B and ESC D as a list terminator. NUL is also used with other printer control codes to select options (for example, ESC S).</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>LPRINT CHR$ (0);</td>
</tr>
<tr>
<td>BEL</td>
<td>Bell</td>
</tr>
<tr>
<td></td>
<td>Sounds the printer buzzer for 1 second.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>LPRINT CHR$ (7);</td>
</tr>
<tr>
<td>HT</td>
<td>Horizontal Tab</td>
</tr>
<tr>
<td></td>
<td>Tabs to the next horizontal tab stop. Tab stops are set with ESC D. No tab stops are set when the printer is powered on. (Graphics Printer sets a tab stop every 8 columns when powered on.)</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>LPRINT CHR$ (9);</td>
</tr>
<tr>
<td>LF</td>
<td>Line Feed</td>
</tr>
<tr>
<td></td>
<td>Spaced the paper up one line. Line spacing is 1/6-inch unless reset by ESC A, ESC O, ESC 1, ESC 2 or ESC 3.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>LPRINT CHR$ (10);</td>
</tr>
<tr>
<td>VT</td>
<td>Vertical Tab</td>
</tr>
<tr>
<td></td>
<td>Spaced the paper to the next vertical tab position. (Graphics Printer does not allow vertical tabs to be set; therefore, the VT code is treated as LF.)</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>LPRINT CHR$ (11);</td>
</tr>
<tr>
<td>FF</td>
<td>Form Feed</td>
</tr>
<tr>
<td></td>
<td>Advances the paper to the top of the next page.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The location of the paper, when the printer is powered on, determines the top of the page. The next top of page is 11 inches from that position. ESC C can be used to change the page length.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>LPRINT CHR$ (12);</td>
</tr>
<tr>
<td>CR</td>
<td>Carriage Return</td>
</tr>
<tr>
<td></td>
<td>Ends the line that the printer is on and prints the data remaining in the printer buffer. (No Line Feed operation takes place.)</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> IBM Personal Computer BASIC adds a Line Feed unless 128 is added [for example, CHR$ (141)].</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>LPRINT CHR$ (13);</td>
</tr>
</tbody>
</table>

1-92 Printers
<table>
<thead>
<tr>
<th>Printer Code</th>
<th>Printer Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>SO</td>
<td><strong>Shift Out (Double Width)</strong>&lt;br&gt;Changes the printer to the Double Width print mode. <strong>Note:</strong> A Carriage Return, Line Feed or DC4 cancels Double Width print mode.</td>
</tr>
<tr>
<td></td>
<td>Example:&lt;br&gt;LPRINT CHR$(14);</td>
</tr>
<tr>
<td>SI</td>
<td><strong>Shift In (Compressed)</strong>&lt;br&gt;Changes the printer to the Compressed Character print mode.</td>
</tr>
<tr>
<td></td>
<td>Example:&lt;br&gt;LPRINT CHR$(15);</td>
</tr>
<tr>
<td>DC1</td>
<td><strong>Device Control 1 (Printer Selected)</strong>&lt;br&gt;(Graphics Printer ignores DC1)&lt;br&gt;Printer accepts data from the system unit. Printer DIP switch 1-8 must be set to the Off position.</td>
</tr>
<tr>
<td></td>
<td>Example:&lt;br&gt;LPRINT CHR$(17);</td>
</tr>
<tr>
<td>DC2</td>
<td><strong>Device Control 2 (Compressed Off)</strong>&lt;br&gt;Stops printing in the Compressed print mode.</td>
</tr>
<tr>
<td></td>
<td>Example:&lt;br&gt;LPRINT CHR$(18);</td>
</tr>
<tr>
<td>DC3</td>
<td><strong>Device Control 3 (Printer Deselected)</strong>&lt;br&gt;(Graphics Printer ignores DC3)&lt;br&gt;Printer does not accept data from the system unit. The system unit must have the printer select line low, and DIP switch 1-8 must be in the Off position.</td>
</tr>
<tr>
<td></td>
<td>Example:&lt;br&gt;LPRINT CHR$(19);</td>
</tr>
<tr>
<td>DC4</td>
<td><strong>Device Control 4 (Double Width Off)</strong>&lt;br&gt;Stops printing in the Double Width print mode.</td>
</tr>
<tr>
<td></td>
<td>Example:&lt;br&gt;LPRINT CHR$(20);</td>
</tr>
<tr>
<td>CAN</td>
<td><strong>Cancel</strong>&lt;br&gt;Clears the printer buffer. Control codes, except SO, remain in effect.</td>
</tr>
<tr>
<td></td>
<td>Example:&lt;br&gt;LPRINT CHR$(24);</td>
</tr>
<tr>
<td>ESC</td>
<td><strong>Escape</strong>&lt;br&gt;Lets the printer know that the next data sent is a printer command. <strong>(See the following list of commands.)</strong></td>
</tr>
<tr>
<td></td>
<td>Example:&lt;br&gt;LPRINT CHR$(27);</td>
</tr>
<tr>
<td>Printer Code</td>
<td>Printer Function</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------</td>
</tr>
</tbody>
</table>
| ESC -        | Escape Minus (Underline)  
              Format: ESC -;n;  
              (Graphics Printer only)  
              ESC - followed by a 1, prints all of the following data with an underline.  
              ESC - followed by a 0 (zero), cancels the Underline print mode.  
              Example:  
              `LPRINT CHR$(27);CHR$(45);CHR$(1);` |
| ESC 0        | Escape Zero (1/8-Inch Line Feeding)  
              Changes paper feeding to 1/8 inch.  
              Example:  
              `LPRINT CHR$(27);CHR$(48);` |
| ESC 1        | Escape One (7/72-Inch Line Feeding)  
              Changes paper feed to 7/72 inch.  
              Example:  
              `LPRINT CHR$(27);CHR$(49);` |
| ESC 2        | Escape Two (Starts Variable Line Feeding)  
              ESC 2 is an execution command for ESC A. If no ESC A command has been given, line feeding returns to 1/6-inch.  
              Example:  
              `LPRINT CHR$(27);CHR$(50);` |
| ESC 3        | Escape Three (Variable Line Feeding)  
              Format: ESC 3;n;  
              (Graphics Printer only)  
              Changes the paper feeding to n/216-inch. The example below sets the paper feeding to 54/216 (1/4) inch. The value of n must be between 1 and 255.  
              Example:  
              `LPRINT CHR$(27);CHR$(51);CHR$(54);` |
| ESC 6        | Escape Six (Select Character Set 2)  
              (Graphics Printer only)  
              Selects character set 2. (See "Printer Character Set 2.")  
              Example:  
              `LPRINT CHR$(27);CHR$(54);` |
| ESC 7        | Escape Seven (Select Character Set 1.)  
              (Graphics Printer only)  
              Selects character set 1. (See "Printer Character Set 1.")  
              Character set 1 is selected when the printer is powered on or reset.  
              Example:  
              `LPRINT CHR$(27);CHR$(55);` |
| ESC 8        | Escape Eight (Ignore Paper End)  
              Allows the printer to print to the end of the paper. The printer ignores the Paper End switch.  
              Example:  
              `LPRINT CHR$(27);CHR$(56);` |
<table>
<thead>
<tr>
<th>Printer Code</th>
<th>Printer Function</th>
</tr>
</thead>
</table>
| ESC 9        | **Escape Nine (Cancel Ignore Paper End)**  
Cancels the Ignore Paper End command. ESC 9 is selected when the printer is powered on or reset.  
Example:  
LPRINT CHR$(27);CHR$(57); | |
| ESC <        | **Escape Less Than (Home Head)**  
(Graphics Printer only)  
The print head will return to the left margin to print the line following ESC <. This will occur for one line only.  
Example:  
LPRINT CHR$(27);CHR$(60); | |
| ESC A        | **Escape A (Sets Variable Line Feeding)**  
Format: ESC A;n;  
Escape A sets the line-feed to \( \frac{n}{72} \)-inch. The example below tells the printer to set line feeding to 24/72-inch. ESC 2 must be sent to the printer before the line feeding will change. For example, ESC A;24 (text) ESC 2 (text). The text following ESC A;24 will space at the previously set line-feed increments. The text following ESC 2 will be printed with new line-feed increments of 24/72-inch. Any increment between 1/72 and 85/72 may be used.  
Example:  
LPRINT CHR$(27);CHR$(65);CHR$(24);CHR$(27);CHR$(50); | |
| ESC B        | **Escape B (Set Vertical Tabs)**  
Format: ESC B;n_1;n_2;...n_k;NUL;  
(Graphics Printer ignores ESC B)  
Sets vertical tab stop positions. Up to 64 vertical tab stop positions are recognized by the printer. The n’s, in the format above, are used to indicate tab stop positions. Tab stop numbers must be received in ascending numeric order. The tab stop numbers will not become valid until the NUL code is entered. Once vertical tab stops are established, they will be valid until new tab stops are specified. (If the printer is reset or powered Off, set tab stops are cleared.) If no tab stop is set, the Vertical Tab command behaves as a Line Feed command. ESC B followed only by NUL will cancel tab stops. The form length must be set by the ESC C command prior to setting tabs.  
Example:  
LPRINT CHR$(27);CHR$(66);CHR$(10);CHR$(20);CHR$(40);CHR$(0); | |
<table>
<thead>
<tr>
<th>Printer Code</th>
<th>Printer Function</th>
</tr>
</thead>
</table>
| ESC C        | **Escape C (Set Lines per Page)**  
Format: ESC C;n;  
Sets the page length. The ESC C command must have a value following it to specify the length of page desired. (Maximum form length for the printer is 127 lines.)  
The example below sets the page length to 55 lines. The printer defaults to 66 lines per page when powered on or reset.  
Example:  
LPRINT CHR$(27);CHR$(67);CHR$(55); |
| ESC D        | **Escape D (Set Horizontal Tab Stops)**  
Format: ESC D;n₁;n₂;...nₖ;NUL;  
Sets the horizontal tab stop positions. The example below shows the horizontal tab stop positions set at printer column positions of 10, 20, and 40. They are followed by CHR$(0), the NUL code. They must also be in ascending numeric order as shown. Tab stops can be set between 1 and 80. When in the Compressed print mode, tab stops can be set up to 132.  
The maximum number of tabs that can be set is 112. The Graphics Printer can have a maximum of 28 tab stops. The HT (CHR$(9)) is used to execute a tab operation.  
Example:  
LPRINT CHR$(27);CHR$(67);CHR$(0);CHR$(12); |
| ESC E        | **Escape E (Emphasized)**  
Changes the printer to the Emphasized print mode. The speed of the printer is reduced to half speed during the Emphasized print mode.  
Example:  
LPRINT CHR$(27);CHR$(69); |
| ESC F        | **Escape F (Emphasized Off)**  
Stops printing in the Emphasized print mode.  
Example:  
LPRINT CHR$(27);CHR$(70); |
| ESC G        | **Escape G (Double Strike)**  
Changes the printer to the Double Strike print mode. The paper is spaced 1/216 of an inch before the second pass of the print head.  
Example:  
LPRINT CHR$(27);CHR$(71); |
<table>
<thead>
<tr>
<th>Printer Code</th>
<th>Printer Function</th>
</tr>
</thead>
</table>
| ESC H        | **Escape H (Double Strike Off)**  
Stops printing in the Double Strike mode.  
Example:  
LPRINT CHR$(27);CHR$(72); | |
| ESC J        | **Escape J (Set Variable Line Feeding)**  
Format: ESC J;n;  
(Graphics Printer only)  
When ESC J is sent to the printer, the paper will feed in increments of n/216 of an inch. The value of n must be between 1 and 255.  
The example below gives a line feed of 50/216-inch. ESC J is canceled after the line feed takes place.  
Example:  
LPRINT CHR$(27);CHR$(74);CHR$(50); | |
| ESC K        | **Escape K (480 Bit-Image Graphics Mode)**  
Format ESC K;n₁;n₂;v₁;v₂;...vₖ;  
(Graphics Printer only)  
Changes from the Text mode to the Bit-Image Graphics mode.  
n₁ and n₂ are one byte, which specify the number of bit-image data bytes to be transferred. v₁ through vₖ are the bytes of the bit-image data. The number of bit-image data bytes (k) is equal to n₁ +256n₂ and cannot exceed 480 bytes. At every horizontal position, each byte can print up to 8 vertical dots. Bit-image data may be mixed with text data on the same line.  
**Note:** Assign values to n₁ and n₂ as follows:  
n₁ represents values from 0 - 255.  
n₂ represents values from 0 - 1 x 256.  
MSB is most significant bit and LSB is least significant bit. |

<table>
<thead>
<tr>
<th>n₂</th>
<th>MSB</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>14</td>
<td>13</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>n₁</th>
<th>MSB</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

Printers 1-97
Data sent to the printer.

<table>
<thead>
<tr>
<th>Text (20 characters)</th>
<th>ESC</th>
<th>K</th>
<th>n=360</th>
<th>Bit-image data</th>
<th>Next data</th>
</tr>
</thead>
</table>

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

Data sent to the printer.

<table>
<thead>
<tr>
<th>Data A</th>
<th>ESC</th>
<th>K</th>
<th>n₁</th>
<th>n₂</th>
<th>Data B</th>
<th>Data C</th>
<th>ESC</th>
<th>K</th>
<th>n₁</th>
<th>n₂</th>
<th>Data D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text data</td>
<td>Length of data</td>
<td>Bit-image data</td>
<td>Text data</td>
<td>Length of data</td>
<td>Bit-image data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Example:

```plaintext
TYPE B:GRAPH.TXT
1 'OPEN PRINTER IN RANDOM MODE WITH LENGTH OF 255
2 OPEN "LPT1:," AS #1
3 WIDTH "LPT1:",255
4 PRINT #1,CHR$(13);CHR$(10);
5 SLASH$=CHR$(1)+CHR$(02)+CHR$(04)+CHR$(08)
6 SLASH$=SLASH$+CHR$(16)+CHR$(32)+CHR$(64)+CHR$(128)+CHR$(0)
7 GAP$=CHR$(0)+CHR$(0)+CHR$(0)
8 NDOTS=480
9 'ESC K N1 N2
10 PRINT #1,CHR$(27);"K";CHR$(NDOTS MOD 256);CHR$(FIX (NDOTS/256));
11 ' SEND NDOTS NUMBER OF BIT IMAGE BYTES
12 FOR I=1 TO NDOTS/12 'NUMBER OF SLASHES TO PRINT USING GRAPHICS
13 PRINT #1,SLASH$;GAP$;
14 NEXT I
15 CLOSE
16 END
```

This example will give you a row of slashes printed in the 480 Bit-Image mode.
<table>
<thead>
<tr>
<th>Printer Code</th>
<th>Printer Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESC L</td>
<td><strong>Escape L (960 Bit-Image Graphics Mode)</strong>&lt;br&gt;Format: ESC L;n₁,n₂;v₁,v₂;...vₖ.&lt;br&gt;(Graphics Printer only)&lt;br&gt;Changes from the Text mode to the Bit-Image Graphics mode. The input is similar to ESC K. The 960 Bit-Image mode prints at half the speed of the 480 Bit-Image Graphics mode, but can produce a denser graphic image. The number of bytes of bit-image Data (k) is n₁ + 256n₂ but cannot exceed 960. n₁ is in the range of 0 to 255.</td>
</tr>
<tr>
<td>ESC N</td>
<td><strong>Escape N (Set Skip Perforation)</strong>&lt;br&gt;Format ESC N;n;&lt;br&gt;(Graphics Printer only)&lt;br&gt;Sets the Skip Perforation function. The number following ESC N sets the value for the number of lines of Skip Perforation. The example shows a 12-line skip perforation. This will print 54 lines and feed the paper 12 lines. The value of n must be between 1 and 127. ESC N must be reset anytime the page length (ESC C) is changed.&lt;br&gt;Example:&lt;br&gt;CHR$(27);CHR$(78);CHR$(12);</td>
</tr>
<tr>
<td>ESC O</td>
<td><strong>Escape O (Cancel Skip Perforation)</strong>&lt;br&gt;(Graphics Printer only)&lt;br&gt;Cancels the Skip Perforation function.&lt;br&gt;Example:&lt;br&gt;LPRINT CHR$(27);CHR$(79);</td>
</tr>
<tr>
<td>ESC S</td>
<td><strong>Escape S (Subscript/Superscript)</strong>&lt;br&gt;Format: ESC S;n;&lt;br&gt;(Graphics Printer only)&lt;br&gt;Changes the printer to the Subscript print mode when ESC S is followed by a 1, as in the example below. When ESC S is followed by a 0 (zero), the printer will print in the Superscript print mode.&lt;br&gt;Example:&lt;br&gt;LPRINT CHR$(27);CHR$(83);CHR$(1);</td>
</tr>
<tr>
<td>ESC T</td>
<td><strong>Escape T (Subscript/Superscript Off)</strong>&lt;br&gt;(Graphics Printer only)&lt;br&gt;The printer stops printing in the Subscript or Superscript print mode.&lt;br&gt;Example:&lt;br&gt;LPRINT CHR$(27);CHR$(84);</td>
</tr>
<tr>
<td>ESC U</td>
<td><strong>Escape U (Unidirectional Printing)</strong>&lt;br&gt;Format: ESC U;n;&lt;br&gt;(Graphics Printer only)&lt;br&gt;The printer will print from left to right following the input of ESC U;1. When ESC U is followed by a 0 (zero), the left to right printing operation is canceled. The Unidirectional print mode (ESC U) ensures a more accurate print-start position for better print quality.&lt;br&gt;Example:&lt;br&gt;LPRINT CHR$(27);CHR$(85);CHR$(1);</td>
</tr>
<tr>
<td>Printer Code</td>
<td>Printer Function</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------</td>
</tr>
<tr>
<td><strong>ESC W</strong></td>
<td><strong>Escape W (Double Width)</strong></td>
</tr>
<tr>
<td></td>
<td>Format: ESC W;n;</td>
</tr>
<tr>
<td></td>
<td>(Graphics Printer only)</td>
</tr>
<tr>
<td></td>
<td>Changes the printer to the Double Width print mode when ESC W is followed by a 1. This mode is not canceled by a line-feed operation and must be canceled with ESC W followed by a 0 (zero).</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>LPRINT CHRS(27);CHRS(87);CHRS(1);</td>
</tr>
<tr>
<td><strong>ESC Y</strong></td>
<td><strong>Escape Y (960 Bit-Image Graphics Mode Normal Speed)</strong></td>
</tr>
<tr>
<td></td>
<td>Format: ESC Y n1;n2;v1;v2;...vk;</td>
</tr>
<tr>
<td></td>
<td>(Graphics Printer only)</td>
</tr>
<tr>
<td></td>
<td>Changes from the Text mode to the 960 Bit-Image Graphics mode. The printer prints at normal speed during this operation and cannot print dots on consecutive dot positions. The input of data is similar to ESC L.</td>
</tr>
<tr>
<td><strong>ESC Z</strong></td>
<td><strong>Escape Z (1920 Bit-Image Graphics Mode)</strong></td>
</tr>
<tr>
<td></td>
<td>Format: ESC Z;n1;n2;v1;v2;...vk;</td>
</tr>
<tr>
<td></td>
<td>(Graphics Printer only)</td>
</tr>
<tr>
<td></td>
<td>Changes from the Text mode to the 1920 Bit-Image Graphics mode. The input is similar to the other Bit-Image Graphics modes. ESC Z can print only every third dot position.</td>
</tr>
<tr>
<td><strong>DEL</strong></td>
<td><strong>Delete (Clear Printer Buffer)</strong></td>
</tr>
<tr>
<td></td>
<td>(Graphics Printer ignores DEL)</td>
</tr>
<tr>
<td></td>
<td>Clears the printer buffer. Control codes, except SO, still remain in effect. DIP switch 1-5 must be in the Off position.</td>
</tr>
<tr>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>LPRINT CHRS(127);</td>
</tr>
</tbody>
</table>

1-100 Printers
<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LT</td>
<td>VT</td>
<td>FF</td>
<td>CR</td>
<td>SO</td>
<td>SI</td>
<td>DC1</td>
<td>DC2</td>
<td>DC3</td>
<td>DC4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAN</td>
<td>ESC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP</td>
<td>&quot;</td>
<td>#</td>
<td>$</td>
<td>%</td>
<td>&amp;</td>
<td>'</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>41</td>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
<td>46</td>
<td>47</td>
<td>48</td>
<td>49</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(</td>
<td>)</td>
<td>*</td>
<td>+</td>
<td>,</td>
<td>-</td>
<td>.</td>
<td>/</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>50</td>
<td>51</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
<td>56</td>
<td>57</td>
<td>58</td>
<td>59</td>
</tr>
<tr>
<td>60</td>
<td>61</td>
<td>62</td>
<td>63</td>
<td>64</td>
<td>65</td>
<td>66</td>
<td>67</td>
<td>68</td>
<td>69</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>;</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>71</td>
<td>72</td>
<td>73</td>
<td>74</td>
<td>75</td>
<td>76</td>
<td>77</td>
<td>78</td>
<td>79</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;</td>
<td>=</td>
<td>&gt;</td>
<td>?</td>
<td>@</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
</tr>
<tr>
<td>80</td>
<td>81</td>
<td>82</td>
<td>83</td>
<td>84</td>
<td>85</td>
<td>86</td>
<td>87</td>
<td>88</td>
<td>89</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>G</td>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>M</td>
<td>N</td>
<td>O</td>
</tr>
<tr>
<td>90</td>
<td>91</td>
<td>92</td>
<td>93</td>
<td>94</td>
<td>95</td>
<td>96</td>
<td>97</td>
<td>98</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>S</td>
<td>T</td>
<td>U</td>
<td>V</td>
<td>W</td>
<td>X</td>
<td>Y</td>
</tr>
<tr>
<td>100</td>
<td>101</td>
<td>102</td>
<td>103</td>
<td>104</td>
<td>105</td>
<td>106</td>
<td>107</td>
<td>108</td>
<td>109</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Z</td>
<td>[</td>
<td>\</td>
<td>]</td>
<td>^</td>
<td>_</td>
<td>`</td>
<td>a</td>
<td>b</td>
<td>c</td>
</tr>
<tr>
<td>110</td>
<td>111</td>
<td>112</td>
<td>113</td>
<td>114</td>
<td>115</td>
<td>116</td>
<td>117</td>
<td>118</td>
<td>119</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>d</td>
<td>e</td>
<td>f</td>
<td>g</td>
<td>h</td>
<td>i</td>
<td>j</td>
<td>k</td>
<td>l</td>
<td>m</td>
</tr>
<tr>
<td>120</td>
<td>121</td>
<td>122</td>
<td>123</td>
<td>124</td>
<td>125</td>
<td>126</td>
<td>127</td>
<td>128</td>
<td>129</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n</td>
<td>o</td>
<td>p</td>
<td>q</td>
<td>r</td>
<td>s</td>
<td>t</td>
<td>u</td>
<td>v</td>
<td>w</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>y</td>
<td>z</td>
<td>[</td>
<td>]</td>
<td>~</td>
<td>DEL</td>
<td>NUL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Matrix Printer Character Set (Part 2 of 2)
<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUL</td>
<td>BEL</td>
<td>HT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
</tr>
<tr>
<td>LF</td>
<td>VT</td>
<td>FF</td>
<td>CR</td>
<td>SO</td>
<td>SI</td>
<td>DC2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
</tr>
<tr>
<td>DC4</td>
<td>CAN</td>
<td>ESC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>31</td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
</tr>
<tr>
<td>SP</td>
<td>!</td>
<td>'</td>
<td>#</td>
<td>$</td>
<td>%</td>
<td>&amp;</td>
<td>'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>41</td>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
<td>46</td>
<td>47</td>
<td>48</td>
<td>49</td>
</tr>
<tr>
<td>(</td>
<td>)</td>
<td>*</td>
<td>+</td>
<td>,</td>
<td>-</td>
<td>.</td>
<td>/</td>
<td>O</td>
<td>1</td>
</tr>
<tr>
<td>50</td>
<td>51</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
<td>56</td>
<td>57</td>
<td>58</td>
<td>59</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>:</td>
<td>;</td>
</tr>
<tr>
<td>60</td>
<td>61</td>
<td>62</td>
<td>63</td>
<td>64</td>
<td>65</td>
<td>66</td>
<td>67</td>
<td>68</td>
<td>69</td>
</tr>
<tr>
<td>&lt;</td>
<td>=</td>
<td>&gt;</td>
<td>?</td>
<td>@</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
</tr>
<tr>
<td>70</td>
<td>71</td>
<td>72</td>
<td>73</td>
<td>74</td>
<td>75</td>
<td>76</td>
<td>77</td>
<td>78</td>
<td>79</td>
</tr>
<tr>
<td>F</td>
<td>G</td>
<td>H</td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>M</td>
<td>N</td>
<td>O</td>
</tr>
<tr>
<td>80</td>
<td>81</td>
<td>82</td>
<td>83</td>
<td>84</td>
<td>85</td>
<td>86</td>
<td>87</td>
<td>88</td>
<td>89</td>
</tr>
<tr>
<td>P</td>
<td>Q</td>
<td>R</td>
<td>S</td>
<td>T</td>
<td>U</td>
<td>V</td>
<td>W</td>
<td>X</td>
<td>Y</td>
</tr>
<tr>
<td>90</td>
<td>91</td>
<td>92</td>
<td>93</td>
<td>94</td>
<td>95</td>
<td>96</td>
<td>97</td>
<td>98</td>
<td>99</td>
</tr>
<tr>
<td>Z</td>
<td>[</td>
<td>\</td>
<td>]</td>
<td>^</td>
<td>_</td>
<td>`</td>
<td>a</td>
<td>b</td>
<td>c</td>
</tr>
<tr>
<td>100</td>
<td>101</td>
<td>102</td>
<td>103</td>
<td>104</td>
<td>105</td>
<td>106</td>
<td>107</td>
<td>108</td>
<td>109</td>
</tr>
<tr>
<td>d</td>
<td>e</td>
<td>f</td>
<td>g</td>
<td>h</td>
<td>i</td>
<td>j</td>
<td>k</td>
<td>l</td>
<td>m</td>
</tr>
<tr>
<td>110</td>
<td>111</td>
<td>112</td>
<td>113</td>
<td>114</td>
<td>115</td>
<td>116</td>
<td>117</td>
<td>118</td>
<td>119</td>
</tr>
<tr>
<td>n</td>
<td>o</td>
<td>p</td>
<td>q</td>
<td>r</td>
<td>s</td>
<td>t</td>
<td>u</td>
<td>v</td>
<td>w</td>
</tr>
<tr>
<td>120</td>
<td>121</td>
<td>122</td>
<td>123</td>
<td>124</td>
<td>125</td>
<td>126</td>
<td>127</td>
<td>128</td>
<td>129</td>
</tr>
<tr>
<td>x</td>
<td>y</td>
<td>z</td>
<td>{</td>
<td>}</td>
<td>~</td>
<td>NUL</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

Graphics Printer Character Set 2 (Part 1 of 2)
Graphics Printer Character Set 2 (Part 2 of 2)
IBM Printer Adapter

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

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

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

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

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

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

1-108 Printer Adapter
Programming Considerations

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

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

A description of each instruction follows.

<table>
<thead>
<tr>
<th>IBM Monochrome Display &amp; Printer Adapter</th>
<th>Printer Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output to address hex 3BC</td>
<td>Output to address hex 378</td>
</tr>
<tr>
<td>Bit 7</td>
<td>Bit 3</td>
</tr>
<tr>
<td>Pin 9</td>
<td>Pin 5</td>
</tr>
<tr>
<td>Bit 6</td>
<td>Bit 2</td>
</tr>
<tr>
<td>Pin 8</td>
<td>Pin 4</td>
</tr>
<tr>
<td>Bit 5</td>
<td>Bit 1</td>
</tr>
<tr>
<td>Pin 7</td>
<td>Pin 3</td>
</tr>
<tr>
<td>Bit 4</td>
<td>Bit 0</td>
</tr>
<tr>
<td>Pin 6</td>
<td>Pin 2</td>
</tr>
</tbody>
</table>

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

It is essential that the external device not try to pull these lines to ground.
<table>
<thead>
<tr>
<th>IBM Monochrome Display &amp; Printer Adapter</th>
<th>Printer Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output to address hex 3BE</td>
<td>Output to address hex 37A</td>
</tr>
<tr>
<td>Bit 4</td>
<td>Bit 3</td>
</tr>
<tr>
<td>IRQ</td>
<td>Pin 17</td>
</tr>
<tr>
<td>Enable</td>
<td>Pin 16</td>
</tr>
<tr>
<td></td>
<td>Pin 14</td>
</tr>
<tr>
<td></td>
<td>Pin 1</td>
</tr>
</tbody>
</table>

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

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

<table>
<thead>
<tr>
<th>IBM Monochrome Display &amp; Printer Adapter</th>
<th>Printer Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input from address Hex 3BC</td>
<td>Input from address hex 378</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin 11</td>
<td>Pin 10</td>
<td>Pin 12</td>
<td>Pin 13</td>
<td>Pin 15</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

This instruction causes the data present on pins 1, 14, 15, 17, and the IRQ bit to read by the processor. In the absence of external drive applied to these pins, data read by the processor will exactly match data last written to hex 3BE in the same bit positions. Note that data bits 0-2 are not included. If external drivers are dotted to these pins, that data will be ORed with data applied to the pins by the hex 3BE latch.

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQ</td>
<td>Enable</td>
<td>Pin 17</td>
<td>Por=0</td>
<td>Pin 16</td>
<td>Por=0</td>
<td>Pin 14</td>
<td>Pin 11</td>
</tr>
<tr>
<td>Pour=0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

These pins assume the states shown after a reset from the processor.
Note: All outputs are software-generated, and all inputs are real-time signals (not latched).

### At Standard TTL Levels

<table>
<thead>
<tr>
<th>Signal Name</th>
<th>Adapter Pin Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Strobe</td>
<td>1</td>
</tr>
<tr>
<td>+Data Bit 0</td>
<td>2</td>
</tr>
<tr>
<td>+Data Bit 1</td>
<td>3</td>
</tr>
<tr>
<td>+Data Bit 2</td>
<td>4</td>
</tr>
<tr>
<td>+Data Bit 3</td>
<td>5</td>
</tr>
<tr>
<td>+Data Bit 4</td>
<td>6</td>
</tr>
<tr>
<td>+Data Bit 5</td>
<td>7</td>
</tr>
<tr>
<td>+Data Bit 6</td>
<td>8</td>
</tr>
<tr>
<td>+Data Bit 7</td>
<td>9</td>
</tr>
<tr>
<td>- Acknowledge</td>
<td>10</td>
</tr>
<tr>
<td>+Busy</td>
<td>11</td>
</tr>
<tr>
<td>+P.End (out of paper)</td>
<td>12</td>
</tr>
<tr>
<td>+Select</td>
<td>13</td>
</tr>
<tr>
<td>- Auto Feed</td>
<td>14</td>
</tr>
<tr>
<td>- Error</td>
<td>15</td>
</tr>
<tr>
<td>- Initialize Printer</td>
<td>16</td>
</tr>
<tr>
<td>- Select Input</td>
<td>17</td>
</tr>
<tr>
<td>Ground</td>
<td>18-25</td>
</tr>
</tbody>
</table>

### Connector Specifications

1-112 Printer Adapter
IBM Monochrome Display and Printer Adapter

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

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

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

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

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

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

Characteristics of the monitor adapter are listed below:

- 80 by 25 screen
- Direct-drive output
- 9 by 14 character box
- 7 by 9 character
- 18 kHz monitor
- Character attributes
IBM Monochrome Adapter Block Diagram
Programming Considerations

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

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

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

System configurations that have both an IBM Monochrome Display Adapter and Printer Adapter, and an IBM Color/Graphics Monitor Adapter, must ensure that both adapters are properly initialized after a power-on reset. Damage to either display may occur if not properly initialized.
The IBM Monochrome Display and Printer Adapter supports 256 different character codes. In the character set are alphanumerics and block graphics. Each character in the display buffer has a corresponding character attribute. The character code must be an even address, and the attribute code must be an odd address in the display buffer.

```
 7 6 5 4 3 2 1 0  
| | | | | | | |  
| Character Code  |
Even Address (M)

 7 6 5 4 3 2 1 0  
| BL R G B I R G B |
| | | | | | | |  
| Attribute Code  |
Odd Address (M+1)
```

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

<table>
<thead>
<tr>
<th>Background R G B</th>
<th>Foreground R G B</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
<td>0 0 0</td>
<td>Non-Display</td>
</tr>
<tr>
<td>0 0 0</td>
<td>0 0 1</td>
<td>Underline</td>
</tr>
<tr>
<td>0 0 0</td>
<td>1 1 1</td>
<td>White Character/Black Background</td>
</tr>
<tr>
<td>1 1 1</td>
<td>0 0 0</td>
<td>Reverse Video</td>
</tr>
</tbody>
</table>
The 4K display buffer supports one screen of 25 rows of 80 characters, plus a character attribute for each display character. The starting address of the buffer is hex B0000. The display buffer can be read from using DMA; however, at least one wait-state will be inserted by the processor. The duration of the wait-state will vary, because the processor/monitor access is synchronized with the character clock on this adapter.

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

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

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

*The 6845 Index and Data Registers are used to program the CRT controller to interface the high-resolution IBM Monochrome Display.
<table>
<thead>
<tr>
<th>Bit Number</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>+High Resolution Mode</td>
</tr>
<tr>
<td>1</td>
<td>Not Used</td>
</tr>
<tr>
<td>2</td>
<td>Not Used</td>
</tr>
<tr>
<td>3</td>
<td>+Video Enable</td>
</tr>
<tr>
<td>4</td>
<td>Not Used</td>
</tr>
<tr>
<td>5</td>
<td>+Enable Blink</td>
</tr>
<tr>
<td>6, 7</td>
<td>Not Used</td>
</tr>
</tbody>
</table>

6845 CRT Control Port 1 (Hex 3B8)

<table>
<thead>
<tr>
<th>Bit Number</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>+Horizontal Drive</td>
</tr>
<tr>
<td>1</td>
<td>Reserved</td>
</tr>
<tr>
<td>2</td>
<td>Reserved</td>
</tr>
<tr>
<td>3</td>
<td>+Black/White Video</td>
</tr>
</tbody>
</table>

6845 CRT Status Port (Hex 3BA)

1-118 Monochrome Adapter
At Standard TTL Levels

<table>
<thead>
<tr>
<th>Connector</th>
<th>Description</th>
<th>Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM</td>
<td>Monochrome Display</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Ground</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Ground</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Not Used</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Not Used</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Not Used</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>+Intensity</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>+Video</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>+Horizontal</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>- Vertical</td>
<td>9</td>
</tr>
</tbody>
</table>

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

Connector Specifications
Notes:

1-120  Monochrome Adapter
The high-resolution IBM Monochrome Display attaches to the system unit through two cables approximately 3 feet (914 millimeters) in length. One cable is a signal cable that contains the direct drive interface from the IBM Monochrome Display and Printer Adapter.

The second cable provides ac power to the display from the system unit. This allows the system-unit power switch to also control the display unit. An additional benefit is a reduction in the requirements for wall outlets to power the system. The display contains an 11-½ inch (283 millimeters), diagonal 90° deflection CRT. The CRT and analog circuits are packaged in an enclosure so the display may either sit on top of the system unit or on a nearby tabletop or desk. The unit has both brightness and contrast adjustment controls on the front surface that are easily accessible to the operator.
Operating Characteristics

Screen

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

Video Signal

- Maximum bandwidth of 16.257 MHz.

Vertical Drive

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

Horizontal Drive

- Positive-level, \texttt{TTL-compatibility} at a frequency of 18.432 \texttt{kHz}. 

1-122 Monochrome Display
IBM Color/Graphics Monitor Adapter

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

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

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

The monitor adapter contains 16K bytes of storage. As an example, a 40-column by 25-row display screen uses 1000 bytes to store character information, and 1000 bytes to store attribute/color information. This would mean that up to eight display screens can be stored in the adapter memory. Similarly, in an 80-column by 25-row mode, four display screens may be stored in the adapter. The entire 16K bytes of storage on the display adapter are directly addressable by the processor, which allows maximum software flexibility in managing the screen.
In **A/N** color modes, it is also possible to select the color of the screen's border. One of sixteen colors can be selected.

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

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

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

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

- 16 special characters for game support
- 15 characters for word-processing editing support
- 96 characters for the standard ASCII graphics set
- 48 characters for foreign-language support
- 48 characters for business block-graphics support (allowing drawing of charts, boxes, and tables using single and double lines)
- 16 selected Greek characters
- 15 selected scientific-notation characters
The color/graphics monitor adapter function is packaged on a single card. The direct-drive and composite-video ports are right-angle mounted connectors on the adapter, and extend through the rear panel of the unit. The direct-drive video port is a 9-pin D-shell female connector. The composite-video port is a standard female phono-jack.

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

A block diagram of the color/graphics adapter is on the following page.
Color/Graphics Monitor Adapter Block Diagram

1-126 Color Graphics Adapter
Descriptions of Major Components

Motorola 6845 CRT Controller

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

Mode Set Register

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

Display Buffer

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

Character Generator

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

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

Composite Color Generator

This generator produces base band video color information.

Alphanumeric Mode

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

<table>
<thead>
<tr>
<th>Display-Character Code Byte</th>
<th>Attribute Byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 6 5 4 3 2 1 0</td>
<td>7 6 5 4 3 2 1 0</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Attribute Function</th>
<th>Attribute Byte</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>7 6 5 4 3 2 1 0</td>
</tr>
<tr>
<td>B RBG B</td>
<td></td>
</tr>
<tr>
<td>FG Background Foreground</td>
<td></td>
</tr>
<tr>
<td>Normal</td>
<td>B 0 0 0 1 1 1 1</td>
</tr>
<tr>
<td>Reverse Video</td>
<td>B 1 1 1 1 0 0 0</td>
</tr>
<tr>
<td>Nondisplay (Black)</td>
<td>B 0 0 0 1 0 0 0</td>
</tr>
<tr>
<td>Nondisplay (White)</td>
<td>B 1 1 1 1 1 1 1</td>
</tr>
</tbody>
</table>

I = Highlighted Foreground (Character)
B = Blinking Foreground (Character)
The attribute byte definitions are:

```
    7 6 5 4 3 2 1 0
    B R G B I R G B
```

Foreground Color
Intensity
Background Color
Blinking

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

The low-resolution alphanumeric mode has the following features:

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

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

**Monochrome vs Color/Graphics Character Attributes**

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

<table>
<thead>
<tr>
<th>Attribute Byte</th>
<th>Monochrome Display Adapter</th>
<th>Color/Graphics Monitor Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Background Color</td>
<td>Character Color</td>
</tr>
<tr>
<td>B 0 0 0 0</td>
<td>Black</td>
<td>White</td>
</tr>
<tr>
<td>B 1 1 1 1</td>
<td>White</td>
<td>Black</td>
</tr>
<tr>
<td>B 0 0 0 0</td>
<td>Black</td>
<td>Black</td>
</tr>
<tr>
<td>B 1 1 1 1</td>
<td>White</td>
<td>White</td>
</tr>
</tbody>
</table>
The monochrome display adapter will produce white characters on a white background with any other code. The color/graphics adapter will change foreground and background colors according to the color value selected. The color values for the various red, green, blue, and intensity bit settings are given in the table below.

<table>
<thead>
<tr>
<th>R</th>
<th>G</th>
<th>B</th>
<th>I</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Black</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Blue</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Green</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Cyan</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Red</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Magenta</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Brown</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>White</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Gray</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Light Blue</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Light Green</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Light Cyan</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Light Red</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Light Magenta</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Yellow</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>White (High Intensity)</td>
</tr>
</tbody>
</table>

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

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

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

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

Low-Resolution Color-Graphics Mode

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

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

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

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

<table>
<thead>
<tr>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>C0</td>
<td>C1</td>
<td>C0</td>
<td>C1</td>
<td>C0</td>
<td>C1</td>
<td>C0</td>
</tr>
<tr>
<td>First Display PEL</td>
<td>Second Display PEL</td>
<td>Third Display PEL</td>
<td>Fourth Display PEL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Memory Address (in hex)</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>B8000</td>
<td>Even Scans (0,2,4,...198) 8,000 bytes</td>
</tr>
<tr>
<td>B9F3F</td>
<td>Not Used</td>
</tr>
<tr>
<td>BA000</td>
<td>Odd Scans (1.3.5...199) 8,000 Bytes</td>
</tr>
<tr>
<td>BBF3F</td>
<td>Not Used</td>
</tr>
<tr>
<td>BBFFF</td>
<td></td>
</tr>
</tbody>
</table>

Address hex **B8000** contains PEL instruction for the upper-left corner of the display area.
Color selection is determined by the following logic:

<table>
<thead>
<tr>
<th>C1</th>
<th>CO</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Dot takes on the color of 1 of 16 preselected background colors</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Selects first color of preselected Color Set 1 or Color Set 2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Selects second color of preselected Color Set 1 or Color Set 2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Selects third color of preselected Color Set 1 or Color Set 2</td>
</tr>
</tbody>
</table>

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

The two colors are:

<table>
<thead>
<tr>
<th>Color Set 1</th>
<th>Color Set 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color 1 is Green</td>
<td>Color 1 is Cyan</td>
</tr>
<tr>
<td>Color 2 is Red</td>
<td>Color 2 is Magenta</td>
</tr>
<tr>
<td>Color 3 is Brown</td>
<td>Color 3 is White</td>
</tr>
</tbody>
</table>

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

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

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

- Formats 8 PELs per byte in the following manner:
Description of Basic Operations

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

<table>
<thead>
<tr>
<th>Memory Address</th>
<th>Display Buffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Even) Starting Address</td>
<td></td>
</tr>
<tr>
<td>B8000</td>
<td>Character Code A</td>
</tr>
<tr>
<td>B8001</td>
<td>Attribute A</td>
</tr>
<tr>
<td>B8002</td>
<td>Character Code B</td>
</tr>
<tr>
<td>B8003</td>
<td>Attribute B</td>
</tr>
<tr>
<td>B87CE</td>
<td>Character Code X</td>
</tr>
<tr>
<td>B87CF</td>
<td>Attribute X</td>
</tr>
</tbody>
</table>

(Example of a 40 by 25 Screen)

```
AB
X
```

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

In the graphics mode, the displayed dots and colors (up to 16K bytes) are also fetched from the display buffer. The bit configuration for each graphics mode is explained in “Graphics Mode.”
<table>
<thead>
<tr>
<th>I</th>
<th>R</th>
<th>G</th>
<th>B</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Black</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Blue</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Green</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Cyan</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Red</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Magenta</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Brown</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>White</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Gray</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Light Blue</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Light Green</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Light Cyan</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Light Red</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Light Magenta</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Yellow</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>High Intensity White</td>
</tr>
</tbody>
</table>

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

Summary of Available Colors

Programming Considerations

Programming the 6845 CRT Controller

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

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

The following table defines the values that must be loaded into the 6845 CRT Controller registers to control the different modes of operation supported by the attachment:
<table>
<thead>
<tr>
<th>Address Register</th>
<th>Register Number</th>
<th>Register Type</th>
<th>Units</th>
<th>I/O</th>
<th>40 by 25 Alpha-numeric</th>
<th>80 by 25 Alpha-numeric</th>
<th>Graphic Modes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>R0</td>
<td>Horizontal Total</td>
<td>Character</td>
<td>Write Only</td>
<td>38</td>
<td>71</td>
<td>38</td>
</tr>
<tr>
<td>1</td>
<td>R1</td>
<td>Horizontal Displayed</td>
<td>Character</td>
<td>Write Only</td>
<td>28</td>
<td>50</td>
<td>28</td>
</tr>
<tr>
<td>2</td>
<td>R2</td>
<td>Horizontal Sync Position</td>
<td>Character</td>
<td>Write Only</td>
<td>2D</td>
<td>5A</td>
<td>2D</td>
</tr>
<tr>
<td>3</td>
<td>R3</td>
<td>Horizontal Sync Width</td>
<td>Character</td>
<td>Write Only</td>
<td>0A</td>
<td>0A</td>
<td>0A</td>
</tr>
<tr>
<td>4</td>
<td>R4</td>
<td>Vertical Total</td>
<td>Character Row</td>
<td>Write Only</td>
<td>1F</td>
<td>1F</td>
<td>7F</td>
</tr>
<tr>
<td>5</td>
<td>R5</td>
<td>Vertical Total Adjust</td>
<td>Scan Line</td>
<td>Write Only</td>
<td>06</td>
<td>06</td>
<td>06</td>
</tr>
<tr>
<td>6</td>
<td>R6</td>
<td>Vertical Displayed</td>
<td>Character Row</td>
<td>Write Only</td>
<td>19</td>
<td>19</td>
<td>64</td>
</tr>
<tr>
<td>7</td>
<td>R7</td>
<td>Vertical Sync Position</td>
<td>Character Row</td>
<td>Write Only</td>
<td>1C</td>
<td>1C</td>
<td>70</td>
</tr>
<tr>
<td>8</td>
<td>R8</td>
<td>Interface Mode</td>
<td>-</td>
<td>Write Only</td>
<td>02</td>
<td>02</td>
<td>02</td>
</tr>
<tr>
<td>9</td>
<td>R9</td>
<td>Maximum Scan Line Address</td>
<td>Scan Line</td>
<td>Write Only</td>
<td>07</td>
<td>07</td>
<td>01</td>
</tr>
<tr>
<td>A</td>
<td>R10</td>
<td>Cursor Start</td>
<td>Scan Line</td>
<td>Write Only</td>
<td>06</td>
<td>06</td>
<td>06</td>
</tr>
<tr>
<td>B</td>
<td>R11</td>
<td>Cursor End</td>
<td>Scan Line</td>
<td>Write Only</td>
<td>07</td>
<td>07</td>
<td>07</td>
</tr>
<tr>
<td>C</td>
<td>R12</td>
<td>Start Address (H)</td>
<td>-</td>
<td>Write Only</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>D</td>
<td>R13</td>
<td>Start Address (L)</td>
<td>-</td>
<td>Write Only</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>E</td>
<td>R14</td>
<td>Cursor Address (H)</td>
<td>-</td>
<td>Read/Write</td>
<td>XX</td>
<td>XX</td>
<td>XX</td>
</tr>
<tr>
<td>F</td>
<td>R15</td>
<td>Cursor Address (L)</td>
<td>-</td>
<td>Read/Write</td>
<td>XX</td>
<td>XX</td>
<td>XX</td>
</tr>
<tr>
<td>10</td>
<td>R16</td>
<td>Light Pen (H)</td>
<td>-</td>
<td>Read Only</td>
<td>XX</td>
<td>XX</td>
<td>XX</td>
</tr>
<tr>
<td>11</td>
<td>R17</td>
<td>Light Pen (L)</td>
<td>-</td>
<td>Read Only</td>
<td>XX</td>
<td>XX</td>
<td>XX</td>
</tr>
</tbody>
</table>

**Note:** All register values are given in hexadecimal

**6845 Register Description**

**1-138 Color Graphics Adapter**
Programming the Mode Control and Status Register

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

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

\( Z = \text{don't care condition} \)
Color-Select Register

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

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

Bits 0, 1, 2, 3 These bits select the screen's border color in the 40 x 25 alphanumeric mode. They select the screen's background color (C0-C1) in the medium-resolution (320 by 200) color-graphics mode.

Bits 4 This bit, when set, will select an alternate, intensified set of colors. Selects background colors in the alphanumeric mode.

Bit 5 This bit is only used in the medium-resolution (320 by 200) color-graphics mode. It is used to select the active set of screen colors for the display.

1-140 Color Graphics Adapter
When bit 5 is set to 1, colors are determined as follows:

<table>
<thead>
<tr>
<th>C1</th>
<th>C0</th>
<th>Set Selected</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Background (Defined by bits 0-3 of port hex 3D9)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Cyan</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Magenta</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>White</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>C1</th>
<th>C0</th>
<th>Set Selected</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Background (Defined by bits 0-3 of port hex 3D9)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Green</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Red</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Brown</td>
</tr>
</tbody>
</table>

**Mode-Select Register**

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

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

<table>
<thead>
<tr>
<th>Bit 0</th>
<th>80 x 25 Alphanumeric Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 1</td>
<td>Graphics Select</td>
</tr>
<tr>
<td>Bit 2</td>
<td>Black/White Select</td>
</tr>
<tr>
<td>Bit 3</td>
<td>Enable Video Signal</td>
</tr>
<tr>
<td>Bit 4</td>
<td>High-Resolution (640 x 200) Black/White Mode</td>
</tr>
<tr>
<td>Bit 5</td>
<td>Change Background Intensity to Blink Bit</td>
</tr>
<tr>
<td>Bit 6</td>
<td>Not Used</td>
</tr>
<tr>
<td>Bit 7</td>
<td>Not Used</td>
</tr>
</tbody>
</table>
Bit 0  A  1 selects 80 by 25 alphanumeric mode  
       A  0 selects 40 by 25 alphanumeric mode  

Bit 1  A  1 selects 320 by 200 graphics mode  
       A  0 selects alphanumeric mode  

Bit 2  A  1 selects black-and-white mode  
       A  0 selects color mode  

Bit 3  A  1 enables the video signal at certain times when modes  
       are being changed. The video signal should be disabled  
       when changing modes.  

Bit 4  A  1 selects the high-resolution (640 by 200)  
       black-and-white graphics mode. One color of 8 can be  
       selected on direct-drive sets in this mode by using register  
       hex 3D9.  

Bit 5  When on, this bit will change the character background  
       intensity to the blinking attribute function for  
       alphanumeric modes. When the high-order attribute bit is  
       not selected, 16 background colors (or intensified colors)  
       are available. For normal operation, this bit should be set  
       to 1 to allow the blinking function.
Mode Register Summary

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Enable Blink Attribute
Enable Video Signal
Select Black-and-White Mode
Select 320 x 200 Graphics
80 x 25 Alphanumeric Select

z = don’t care condition

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

Status Register

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

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

Color Graphics Adapter 1-143
Bit 0  This bit, when active, indicates that a regen buffer memory access can be made without interfering with the display.

Bit 1  This bit, when active, indicates that a positive-going edge from the light-pen has set the light pen's trigger. This trigger is reset upon power-on and may also be cleared by performing an I/O Out command to hex address 3DB. No specific data setting is required; the action is address-activated.

Bit 2  The light-pen switch status is reflected in this status bit. The switch is not latched or debounced. A 0 indicates that the switch is on.

Bit 3  This bit, when active, indicates that the raster is in a vertical retrace mode. This is a good time to perform screen-buffer updating.

**Sequence of Events for Changing Modes**

1. Determine the mode of operation.

2. Reset 'video enable' bit in mode-select register.

3. Program 6845 to select mode.

4. Program mode/color select registers including re-enabling video.
Memory Requirements

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

<table>
<thead>
<tr>
<th>Read/Write Memory Address Space (in hex)</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Read/Write Memory</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Display Buffer (16K Bytes)</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>01000</td>
</tr>
<tr>
<td>A0000</td>
</tr>
<tr>
<td>B8000</td>
</tr>
<tr>
<td>BCO00</td>
</tr>
<tr>
<td>C0000</td>
</tr>
<tr>
<td>128K Reserved Regen Area</td>
</tr>
</tbody>
</table>
## Connector Specifications (Part 1 of 2)

### At Standard TTL Levels

<table>
<thead>
<tr>
<th>IBM Color Display or other Direct-Drive Monitor</th>
<th>Color/Graphics Direct-Drive Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ground</td>
<td>Ground</td>
</tr>
<tr>
<td>Ground</td>
<td>2</td>
</tr>
<tr>
<td>Red</td>
<td>3</td>
</tr>
<tr>
<td>Green</td>
<td>4</td>
</tr>
<tr>
<td>Blue</td>
<td>5</td>
</tr>
<tr>
<td>Intensity</td>
<td>6</td>
</tr>
<tr>
<td>Reserved</td>
<td>7</td>
</tr>
<tr>
<td>Horizontal Drive</td>
<td>8</td>
</tr>
<tr>
<td>Vertical Drive</td>
<td>9</td>
</tr>
</tbody>
</table>

### Composite Phono Jack Hookup to Monitor

1. Composite Video Signal of Approximately 1.5 Volts
   - Peak to Peak Amplitude 1
   - Chassis Ground 2

2. Color/Graphics Composite Jack
P1 (4-Pin Berg Strip) for RF Modulator

P2 (6-Pin Berg Strip) for Light-Pen Connector

Color/Graphics Adapter

RF Modulator Interface

<table>
<thead>
<tr>
<th>RF Modulator</th>
<th>Color/Graphics Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>+12 Volts 1</td>
<td>(key) Not Used 2</td>
</tr>
<tr>
<td>(key) Not Used 2</td>
<td></td>
</tr>
<tr>
<td>Composite Video Output 3</td>
<td></td>
</tr>
<tr>
<td>Logic Ground 4</td>
<td></td>
</tr>
</tbody>
</table>

Light Pen Interface

<table>
<thead>
<tr>
<th>Light Pen</th>
<th>Color/Graphics Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Light Pen Input 1</td>
<td></td>
</tr>
<tr>
<td>(key) Not Used 2</td>
<td></td>
</tr>
<tr>
<td>- Light Pen Switch 3</td>
<td></td>
</tr>
<tr>
<td>Chassis Ground 4</td>
<td></td>
</tr>
<tr>
<td>+5 Volts 5</td>
<td></td>
</tr>
<tr>
<td>+12 Volts 6</td>
<td></td>
</tr>
</tbody>
</table>

Connector Specifications (Part 2 of 2)
Notes:
IBM Color Display

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

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

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

Screen

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

Video Signal

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

Vertical Drive

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

Horizontal Drive

- Positive-level, TTL-compatibility, at a frequency of 15.75 kHz.
IBM 5-1/4” Diskette Drive Adapter

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

The adapter is designed for double-density, MFM-coded, diskette drives and uses write precompensation with an analog phase-lock loop for clock and data recovery. The adapter is a general-purpose device using the NEC μPD765 compatible controller. Therefore, the diskette drive parameters are programmable. In addition, the attachment supports the diskette drive's write-protect feature. The adapter is buffered on the I/O bus and uses the system board's direct memory access (DMA) for record data transfers. An interrupt level is also used to indicate when an operation is complete and that a status condition requires processor attention.

In general, the 5-1/4 inch diskette drive adapter presents a high-level command interface to software I/O drivers. A block diagram of the 5-1/4 inch diskette drive adapter is on the following page.
5-1/4 Inch Diskette Drive Adapter Block Diagram
**Functional Description**

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

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

**Digital-Output Register**

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

**Bits 0 and 1**

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

<table>
<thead>
<tr>
<th>Bit</th>
<th>Drive</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0 (A)</td>
</tr>
<tr>
<td>0</td>
<td>1 (B)</td>
</tr>
<tr>
<td>1</td>
<td>2 (C)</td>
</tr>
<tr>
<td>1</td>
<td>3 (D)</td>
</tr>
</tbody>
</table>

**Bit 2**

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

**Bit 3**

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

**Bits 4, 5, 6, and 7**

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

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

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

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

1-154  Diskette Adapter
The FDC is capable of performing 15 different commands. Each command is initiated by a multi-byte transfer from the processor, and the result after execution of the command may also be a multi-byte transfer back to the processor. Because of this multi-byte interchange of information between the FDC and the processor, it is convenient to consider each command as consisting of three phases:

**Command Phase**

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

**Execution Phase**

The FDC performs the operation it was instructed to do.

**Result Phase**

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

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

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

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

**Symbol Descriptions (Part 2 of 2)**
### Command Summary

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

<table>
<thead>
<tr>
<th>Phase</th>
<th>R/W</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Command</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Command Codes</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>MT</td>
<td>MF</td>
<td>SK</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>HD</td>
<td>US1</td>
<td>US0</td>
<td>Sector ID information prior to command execution.</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Data transfer between the FDD and main system.</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Status information after command execution.</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Sector ID information after command execution.</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Execution</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Result</strong></td>
<td>R</td>
<td>ST</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Command</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Command Codes</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>MT</td>
<td>MF</td>
<td>SK</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Sector ID information prior to command execution.</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>HD</td>
<td>US1</td>
<td>US0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Data transfer between the FDD and main system.</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Status information after command execution.</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Sector ID information after command execution.</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1-158  Diskette Adapter
<table>
<thead>
<tr>
<th>Phase</th>
<th>R/W</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>W</td>
<td>MT</td>
<td>MF</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Command Codes</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>HD</td>
<td>US1</td>
<td>US0</td>
<td>Sector ID information to command execution.</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>EOT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>GPL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>DTL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Execution</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Data transfer between the main system and FDD.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Status information after command execution.</td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>ST0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Sector ID information after command execution.</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>MT</td>
<td>MF</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Command Codes</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>HD</td>
<td>US1</td>
<td>US0</td>
<td>Sector ID information prior to command execution.</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>EOT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>GPL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>DTL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Execution</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Data transfer between FDD and main system.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Status ID information after command execution.</td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>ST0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Sector ID information after command execution.</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phase</td>
<td>R/W</td>
<td>Data Bus</td>
<td>Remarks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>-----</td>
<td>----------</td>
<td>----------------------------------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Command</strong></td>
<td></td>
<td><strong>Read a Track</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>0 MF SK 0 0 0 1 0</td>
<td>Command Codes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>X X X X X HD US1 US0</td>
<td>Sector ID information prior to command execution.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>EOT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>GPL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>DTL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Execution</strong></td>
<td></td>
<td></td>
<td>Data transfer between the FDD and main system.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Result</strong></td>
<td>R</td>
<td>ST 0</td>
<td>Status information after command execution.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Command</strong></td>
<td>W</td>
<td>0 MF 0 0 1 0 1 0</td>
<td>Command Codes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Execution</strong></td>
<td></td>
<td><strong>Read ID</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>X X X X X HD US1 US0</td>
<td>The first correct ID information on the cylinder is stored in data register.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Result</strong></td>
<td>R</td>
<td>ST 0</td>
<td>Status information after command execution.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phase</td>
<td>R/W</td>
<td>D7</td>
<td>D6</td>
<td>D5</td>
<td>D4</td>
<td>D3</td>
<td>D2</td>
<td>D1</td>
<td>D0</td>
<td>Remarks</td>
</tr>
<tr>
<td>-------------</td>
<td>-----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>-------------------------------</td>
</tr>
<tr>
<td>Command</td>
<td>W</td>
<td>0</td>
<td>MF</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Command Codes</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Bytes/Sector</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Sector/Track</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Gap 3</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>filler byte.</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>FDC formats an entire cylinder.</td>
</tr>
<tr>
<td>Execution</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Status information after command execution.</td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>ST</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Status information after command execution.</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td></td>
<td>ST</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>In this case, the ID information has no meaning.</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td></td>
<td></td>
<td>ST</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td></td>
<td></td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td></td>
<td></td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td></td>
<td></td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td></td>
<td></td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<p>| Command     | W   | MT | MF | SK | 1  | 0  | 0  | 0  | 1  | Command Codes                |
|             | W   | X  | X  | X  | X  |    |    |    |    | Sector ID information prior to command execution. |
|             | W   |    |    |    |    |    |    |    |    |                               |
|             | W   |    |    |    |    |    |    |    |    | Data compared between the FDD and the main system. |
|             | W   |    |    |    |    |    |    |    |    | Status information after command execution. |
|             | W   |    |    |    |    |    |    |    |    | Sector ID information after Command execution. |
|             | W   |    |    |    |    |    |    |    |    |                               |
| Execution   |     |    |    |    |    |    |    |    |    |                               |
| Result      | R   | ST | 0  |    |    |    |    |    |    |                               |
|             | R   |    | ST | 1  |    |    |    |    |    |                               |
|             | R   |    |    | ST | 2  |    |    |    |    |                               |
|             | R   |    |    | C  |    |    |    |    |    |                               |
|             | R   |    |    | H  |    |    |    |    |    |                               |
|             | R   |    |    | R  |    |    |    |    |    |                               |
|             | R   |    |    | N  |    |    |    |    |    |                               |</p>
<table>
<thead>
<tr>
<th>Phase</th>
<th>R/W</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>W</td>
<td>MT</td>
<td>MF</td>
<td>SK</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>HD</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>US1</td>
<td>US0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>EOT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>GPL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>STP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Remarks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Command Codes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sector ID information prior to command execution.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Execution</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>ST 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Remarks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data compared between the FDD and main system.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Status information after command execution.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sector ID information after command execution.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Phase</th>
<th>R/W</th>
<th>D7</th>
<th>D6</th>
<th>D5</th>
<th>D4</th>
<th>D3</th>
<th>D2</th>
<th>D1</th>
<th>D0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>W</td>
<td>MT</td>
<td>MF</td>
<td>SK</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>HD</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>US1</td>
<td>US0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>EOT</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>GPL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>W</td>
<td></td>
<td></td>
<td></td>
<td>STP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Remarks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Command Codes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sector ID information prior to command execution.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Execution</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td>R</td>
<td>ST 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>R</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Remarks</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data compared between the FDD and main system.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Status information after command execution.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sector ID information after command execution.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1-162 Diskette Adapter
<table>
<thead>
<tr>
<th>Phase</th>
<th>R/W</th>
<th>D7 D6 D5 D4 D3 D2 D1 D0</th>
<th>Data Bus</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command Execution</td>
<td>W</td>
<td>0 0 0 0 0 1 1 1</td>
<td>Recalibrate</td>
</tr>
<tr>
<td>Command No Result</td>
<td>W</td>
<td>X X X X X 0 US1 US0</td>
<td>Command Codes</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>0 0 0 0 1 0 0 0</td>
<td>Head retracted to track 0</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>PCN</td>
<td></td>
</tr>
<tr>
<td>Command No Result</td>
<td>W</td>
<td>0 0 0 0 0 0 1 1</td>
<td>Specify</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>SRT—HUT</td>
<td>Command Codes</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>HLT—ND</td>
<td></td>
</tr>
<tr>
<td>Command No Result</td>
<td>W</td>
<td>0 0 0 0 0 1 0 0</td>
<td>Sense Drive Status</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>X X X X X X HD US1 US0</td>
<td>Command Codes</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>ST 3</td>
<td>Status information about FDD.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Seek</td>
<td>W</td>
<td>0 0 0 0 1 1 1 1</td>
<td>Command Codes</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>X X X X X X HD US1 US0</td>
<td>Head is positioned over proper cylinder on diskette.</td>
</tr>
<tr>
<td></td>
<td>W</td>
<td>NCN</td>
<td></td>
</tr>
<tr>
<td>Command Result</td>
<td>W</td>
<td>0 0 0 0 1 1 1 1</td>
<td>Invalid</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>Invalid Codes</td>
<td>Invalid command codes (NoOp - FDC goes into standby state).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ST 0</td>
<td>ST 0 = 80.</td>
</tr>
</tbody>
</table>

**Remarks**

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

**Command Status Register 0**

1-164  **Diskette Adapter**
<table>
<thead>
<tr>
<th>No.</th>
<th>Name</th>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>End of Cylinder</td>
<td>EN</td>
<td>When the FDC tries to access a sector beyond the final sector of a cylinder, this flag is set.</td>
</tr>
<tr>
<td>D6</td>
<td>—</td>
<td>—</td>
<td>Not used. This bit is always 0 (low).</td>
</tr>
<tr>
<td>D5</td>
<td>Data Error</td>
<td>DE</td>
<td>When the FDC detects a CRC error in either the ID field or the data field, this flag is set.</td>
</tr>
<tr>
<td>D4</td>
<td>Over Run</td>
<td>OR</td>
<td>If the FDC is not serviced by the main system during data transfers within a certain time interval, this flag is set.</td>
</tr>
<tr>
<td>D3</td>
<td>—</td>
<td>—</td>
<td>Not used. This bit is always 0 (low).</td>
</tr>
<tr>
<td>D2</td>
<td>No Data</td>
<td>ND</td>
<td>During execution of a read data, write deleted data, or scan command, if the FDC cannot find the sector specified in the ID register, this flag is set. During execution of the read ID command, if the FDC cannot read the ID field without an error, then this flag is set. During the execution of the read a cylinder command, if the starting sector cannot be found, then this flag is set.</td>
</tr>
<tr>
<td>D1</td>
<td>Not Writable</td>
<td>NW</td>
<td>During execution of a write data, write deleted data, or format-a-cylinder command, if the FDC detects a write-protect signal from the FDD, then this flag is set.</td>
</tr>
<tr>
<td>D0</td>
<td>Missing Address Mark</td>
<td>MA</td>
<td>If the FDC cannot detect the ID address mark, this flag is set. Also, at the same time, the MD (missing address mark in the data field) of status register 2 is set.</td>
</tr>
</tbody>
</table>

Command Status Register 1

Diskette Adapter 1-165
<table>
<thead>
<tr>
<th>Bit No.</th>
<th>Name</th>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>—</td>
<td>—</td>
<td>Not used. This bit is always 0 (low).</td>
</tr>
<tr>
<td>D6</td>
<td>Control Mark</td>
<td>CM</td>
<td>During execution of the read data or scan command, if the FDC encounters a sector that contains a deleted data address mark, this flag is set.</td>
</tr>
<tr>
<td>D5</td>
<td>Data Error in Data Field</td>
<td>DD</td>
<td>If the FDC detects a CRC error in the data, then this flag is set.</td>
</tr>
<tr>
<td>D4</td>
<td>Wrong Cylinder</td>
<td>WC</td>
<td>This bit is related to the ND bit, and when the contents of C on the medium are different from that stored in the ID register, this flag is set.</td>
</tr>
<tr>
<td>D3</td>
<td>Scan Equal Hit</td>
<td>SH</td>
<td>During execution of the scan command, if the condition of “equal” is satisfied, this flag is set.</td>
</tr>
<tr>
<td>D2</td>
<td>Scan Not Satisfied</td>
<td>SN</td>
<td>During execution of the scan command, if the FDC cannot find a sector on the cylinder that meets the condition, then this flag is set.</td>
</tr>
<tr>
<td>D1</td>
<td>Bad Cylinder</td>
<td>BC</td>
<td>This bit is related to the ND bit, and when the contents of C on the medium are different from that stored in the ID register, and the contents of C is FF, then this flag is set.</td>
</tr>
<tr>
<td>D0</td>
<td>Missing Address Mark in Data Field</td>
<td>MD</td>
<td>When data is read from the medium, if the FDC cannot find a data address mark or deleted data address mark, then this flag is set.</td>
</tr>
</tbody>
</table>

Command Status Register 2

1-166 Diskette Adapter
### Command Status Register 3

<table>
<thead>
<tr>
<th>Bit No.</th>
<th>Name</th>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7</td>
<td>Fault</td>
<td>FT</td>
<td>This bit is the status of the fault signal from the FDD.</td>
</tr>
<tr>
<td>D6</td>
<td>Write Protected</td>
<td>WP</td>
<td>This bit is the status of the write-protected signal from the FDD.</td>
</tr>
<tr>
<td>D5</td>
<td>Ready</td>
<td>RY</td>
<td>This bit is the status of the ready signal from the FDD.</td>
</tr>
<tr>
<td>D4</td>
<td>Track 0</td>
<td>T0</td>
<td>This bit is the status of the track 0 signal from the FDD.</td>
</tr>
<tr>
<td>D3</td>
<td>Two Side</td>
<td>TS</td>
<td>This bit is the status of the two-side signal from the FDD.</td>
</tr>
<tr>
<td>D2</td>
<td>Head Address</td>
<td>HD</td>
<td>This bit is the status of the side-select signal from the FDD.</td>
</tr>
<tr>
<td>D1</td>
<td>Unit Select 1</td>
<td>US 1</td>
<td>This bit is the status of the unit-select-1 signal from the FDD.</td>
</tr>
<tr>
<td>D0</td>
<td>Unit Select 0</td>
<td>US 0</td>
<td>This bit is the status of the unit-select-0 signal from the FDD.</td>
</tr>
</tbody>
</table>

### Programming Summary

<table>
<thead>
<tr>
<th>Register</th>
<th>I/O Address Hex</th>
</tr>
</thead>
<tbody>
<tr>
<td>FDC Data Register</td>
<td>3F5</td>
</tr>
<tr>
<td>FDC Main Status Register</td>
<td>3F4</td>
</tr>
<tr>
<td>Digital Output Register</td>
<td>3F2</td>
</tr>
</tbody>
</table>

**Bit 0**
- **Drive**
  - 00: DR #A
  - 10: DR #C
- **Select**
  - 01: DR #B
  - 11: DR #D

2: Not FDC Reset
3: Enable INT & DMA Requests
4: Drive A Motor Enable
5: Drive B Motor Enable
6: Drive C Motor Enable
7: Drive D Motor Enable

All bits cleared with channel reset.

### DPC Registers
FDC Constants (in hex)

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

Drive Constants

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

Comments

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

System I/O Channel Interface

All signals are TTL-compatible:

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

1-168 Diskette Adapter
The following lines are used by this adapter.

+DO-7  (Bidirectional, load: 1 \textbf{74LS}, driver: 74LS 3-state). These eight lines form a bus by which all commands, status, and data are transferred. Bit 0 is the low-order bit.

+A0-9  (Adapter input, load: 1 \textbf{74LS}) These ten lines form an address bus by which a register is selected to receive or supply the byte transferred through lines DO-7. Bit 0 is the low-order bit.

+AEN  (Adapter input, load: 1 \textbf{74LS}) The content of lines AO-9 is ignored if this line is active.

+IOR  (Adapter input, load: 4 \textbf{74LS}) This line and DACK2 being active indicates that the byte of data for which the DMA count was initialized is now being transferred.

-1OW  (Adapter input, load: 1 \textbf{74LS}) The content of lines DO-7 is stored in the register addressed by lines AO-9 or DACK2 at the trailing edge of this signal.

-DACK2  (Adapter input, load: 2 \textbf{74LS}) This line being active degates output DRQ2, selects the FDC data register as the source/destination of bus DO-7, and indirectly gates TIC to IRQ6.

+T/C  (Adapter input, load: 4 \textbf{74LS}) This line and DACK2 being active indicates that the byte of data for which the DMA count was initialized is now being transferred.

+RESET  (Adapter input, load: 1 \textbf{74LS}) An up level aborts any operation in process and clears the digital output register (DOR).
+DRQ2  (Adapter output, driver: 74LS 3-state)
This line is made active when the attachment is ready to transfer a byte of data to or from main storage.
The line is made inactive by DACK2 becoming active or an I/O read of the FDC data register.

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

**Drive A and B Interface**

All signals are **TTL-compatible**:

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

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

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

**Adapter Outputs**

- Drive Select A and B  (Driver: 7438)
These two lines are used by drives A and B to degate all drivers to the adapter and receivers from the attachment (except motor enable) when the line associated with a drive is inactive.
—Motor Enable A and B (Driver: 7438)
The drive associated with each of these lines must control its spindle motor such that it starts when the line becomes active and stops when the line becomes inactive.

—Step (Driver: 7438)
The selected drive moves the read/write head one cylinder in or out per the direction line for each pulse present on this line.

—Direction (Driver: 7438)
For each recognized pulse of the step line, the read/write head moves one cylinder toward the spindle if this line is active, and away from the spindle if inactive.

—Head Select (Driver: 7438)
Head 1 (upper head) will be selected when this line is active (low).

—Write Data (Driver: 7438)
For each inactive to active transition of this line while write enable is active, the selected drive causes a flux change to be stored on the diskette.

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

- **Index**
  The selected drive supplies one pulse per diskette revolution on this line.

- **Write Protect**
  The selected drive makes this line active if a write-protected diskette is mounted in the drive.

- **Track 0**
  The selected drive makes this line active if the read/write head is over track 0.

- **Read Data**
  The selected drive supplies a pulse on this line for each flux change encountered on the diskette.
Note: Lands 1-33 (odd numbers) are on the back of the board. Lands 2-34 (even numbers) are on the front, or component side.

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

Connector Specifications (Part 1 of 2)
### 37-Pin D-Shell Connector

- **Rear Panel**

- **Pin Number**

<table>
<thead>
<tr>
<th>Pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>At Standard TTL Levels</td>
</tr>
<tr>
<td>2</td>
<td>Unused</td>
</tr>
<tr>
<td>3</td>
<td>Index</td>
</tr>
<tr>
<td>4</td>
<td>Motor Enable C</td>
</tr>
<tr>
<td>5</td>
<td>Drive Select D</td>
</tr>
<tr>
<td>6</td>
<td>Drive Select C</td>
</tr>
<tr>
<td>7</td>
<td>Motor Enable D</td>
</tr>
<tr>
<td>8</td>
<td>Direction (Stepper Motor)</td>
</tr>
<tr>
<td>9</td>
<td>Step Pulse</td>
</tr>
<tr>
<td>10</td>
<td>Write Data</td>
</tr>
<tr>
<td>11</td>
<td>Track 0</td>
</tr>
<tr>
<td>12</td>
<td>Write Enable</td>
</tr>
<tr>
<td>13</td>
<td>Write Protect</td>
</tr>
<tr>
<td>14</td>
<td>Read Data</td>
</tr>
<tr>
<td>15</td>
<td>Select Head 1</td>
</tr>
<tr>
<td>16</td>
<td>Ground</td>
</tr>
</tbody>
</table>

**Connector Specifications (Part 2 of 2)**

1-74 Diskette Adapter
IBM 5-1/4" Diskette Drive

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

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

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

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

The diskette drive also has the following sensor systems:

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

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

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

<table>
<thead>
<tr>
<th>Media</th>
<th>Industry-compatible 5-1/4 inch diskette</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tracks per inch</td>
<td>48</td>
</tr>
<tr>
<td>Number of tracks</td>
<td>40</td>
</tr>
<tr>
<td>Dimensions</td>
<td></td>
</tr>
<tr>
<td>Height</td>
<td>3.38 inches (85.85 mm)</td>
</tr>
<tr>
<td>Width</td>
<td>5.87 inches (149.10 mm)</td>
</tr>
<tr>
<td>Depth</td>
<td>8.00 inches (203.2 mm)</td>
</tr>
<tr>
<td>Weight</td>
<td>4.50 pounds (2.04 kg)</td>
</tr>
<tr>
<td>Temperature</td>
<td></td>
</tr>
<tr>
<td>(Exclusive of media)</td>
<td></td>
</tr>
<tr>
<td>Operating</td>
<td>50°F to 112°F (10°C to 44°C)</td>
</tr>
<tr>
<td>Non operating</td>
<td>-40°F to 140°F (-40°C to 60°C)</td>
</tr>
<tr>
<td>Relative humidity</td>
<td></td>
</tr>
<tr>
<td>(Exclusive of media)</td>
<td></td>
</tr>
<tr>
<td>Operating</td>
<td>20% to 80% (non condensing)</td>
</tr>
<tr>
<td>Non operating</td>
<td>5% to 95% (non condensing)</td>
</tr>
<tr>
<td>Seek Time</td>
<td>6 ms track-to-track</td>
</tr>
<tr>
<td>Head Settling Time</td>
<td>15 ms (last track addressed)</td>
</tr>
<tr>
<td>Error Rate</td>
<td>1 per 10⁶ (recoverable)</td>
</tr>
<tr>
<td></td>
<td>1 per 10¹² (non recoverable)</td>
</tr>
<tr>
<td></td>
<td>1 per 10⁶ (seeks)</td>
</tr>
<tr>
<td>Head Life</td>
<td>20,000 hours (normal use)</td>
</tr>
<tr>
<td>Media Life</td>
<td>3.0 x 10⁶ passes per track</td>
</tr>
<tr>
<td>Disk Speed</td>
<td>300 rpm +/- 1.5% (long term)</td>
</tr>
<tr>
<td>Instantaneous Speed Variation</td>
<td>+/- 3.0%</td>
</tr>
<tr>
<td>Start/Stop Time</td>
<td>250 ms (maximum)</td>
</tr>
<tr>
<td>Transfer Rate</td>
<td>250K bits/sec</td>
</tr>
<tr>
<td>Recording Mode</td>
<td>MFM</td>
</tr>
<tr>
<td>Power</td>
<td>+12 Vdc +/- 0.6 V, 900 mA average</td>
</tr>
<tr>
<td></td>
<td>+5 Vdc +/- 0.25 V, 600 mA average</td>
</tr>
</tbody>
</table>

**Mechanical and Electrical Specifications**

1-176  Diskette Drive
Diskettes

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

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

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

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

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

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

Fixed Disk Controller

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

Status Register

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

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>d</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>e</td>
<td>0</td>
</tr>
</tbody>
</table>

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

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

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

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

Sense Bytes

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

<table>
<thead>
<tr>
<th>Bits</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte 0</td>
<td>Address</td>
<td>Valid</td>
<td>O</td>
<td>Error Type</td>
<td>Error Code</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 1</td>
<td>O</td>
<td>0</td>
<td>d</td>
<td>Head Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 2</td>
<td>Cylinder High</td>
<td>Sector Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 3</td>
<td>Cylinder Low</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Remarks  
d = drive
Byte 0  Bits 0, 1, 2, 3  Error code.

Byte 0  Bits 4, 5  Error type.

Byte 0  Bit 6  Set to 0 (spare).

Byte 0  Bit 7  The address valid bit. Set only when the previous command required a disk address, in which case it is returned as a 1; otherwise, it is a 0.

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

<table>
<thead>
<tr>
<th>Error Type</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bits</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>0 0</td>
<td>0 0 0 0</td>
<td>The controller did not detect any error during the execution of the previous operation.</td>
</tr>
<tr>
<td>0 0</td>
<td>0 0 0 1</td>
<td>The controller did not detect an index signal from the drive.</td>
</tr>
<tr>
<td>0 0</td>
<td>0 0 1 0</td>
<td>The controller did not get a seek-complete signal from the drive after a seek operation (for all non-buffered step seeks).</td>
</tr>
<tr>
<td>0 0</td>
<td>0 0 1 1</td>
<td>The controller detected a write fault from the drive during the last operation.</td>
</tr>
<tr>
<td>0 0</td>
<td>0 1 0 0</td>
<td>After the controller selected the drive, the drive did not respond with a ready signal.</td>
</tr>
<tr>
<td>0 0</td>
<td>0 1 0 1</td>
<td>Not used.</td>
</tr>
<tr>
<td>0 0</td>
<td>0 1 1 0</td>
<td>After stepping the maximum number of cylinders, the controller did not receive the track 00 signal from the drive.</td>
</tr>
<tr>
<td>0 0</td>
<td>0 1 1 1</td>
<td>Not used.</td>
</tr>
<tr>
<td>0 0</td>
<td>1 0 0 0</td>
<td>The drive is still seeking. This status is reported by the Test Drive Ready command for an overlap seek condition when the drive has not completed the seek. No time-out is measured by the controller for the seek to complete.</td>
</tr>
<tr>
<td>Error Type</td>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Bits</td>
<td>5 4 3 2 1 0</td>
<td></td>
</tr>
<tr>
<td>0 1 0 0 0 0</td>
<td></td>
<td>ID Read Error: The controller detected an ECC error in the target ID field on the disk.</td>
</tr>
<tr>
<td>0 1 0 0 0 1</td>
<td></td>
<td>Data Error: The controller detected an uncorrectable ECC error in the target sector during a read operation.</td>
</tr>
<tr>
<td>0 1 0 0 1 0</td>
<td></td>
<td>Address Mark: The controller did not detect the target address mark (AM) on the disk.</td>
</tr>
<tr>
<td>0 1 0 0 1 1</td>
<td></td>
<td>Not used.</td>
</tr>
<tr>
<td>0 1 0 1 0 0</td>
<td></td>
<td>Sector Not Found: The controller found the correct cylinder and head, but not the target sector.</td>
</tr>
<tr>
<td>0 1 0 1 0 1</td>
<td></td>
<td>Seek Error: The cylinder or head address (either or both) did not compare with the expected target address as a result of a seek.</td>
</tr>
<tr>
<td>0 1 0 1 1 0</td>
<td></td>
<td>Not used.</td>
</tr>
<tr>
<td>0 1 0 1 1 1</td>
<td></td>
<td>Not used.</td>
</tr>
<tr>
<td>0 1 1 0 0 0</td>
<td></td>
<td>Correctable Data Error: The controller detected a correctable ECC error in the target field.</td>
</tr>
<tr>
<td>0 1 1 0 0 1</td>
<td></td>
<td>Bad Track: The controller detected a bad track flag during the last operation. No retries are attempted on this error.</td>
</tr>
<tr>
<td>Error Type</td>
<td>Error Code</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Bits 1 0 0 0 0</td>
<td></td>
<td>Invalid Command: The controller has received an invalid command from the system unit.</td>
</tr>
<tr>
<td>1 0 0 0 1</td>
<td></td>
<td>Illegal Disk Address: The controller detected an address that is beyond the maximum range.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Error Type</th>
<th>Error Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bits 1 1 0 0 0</td>
<td></td>
<td>RAM Error: The controller detected a data error during the RAM sector-buffer diagnostic test.</td>
</tr>
<tr>
<td>1 1 0 0 1</td>
<td></td>
<td>Program Memory Checksum Error: During this internal diagnostic test, the controller detected a program-memory checksum error.</td>
</tr>
<tr>
<td>1 1 0 1 0</td>
<td></td>
<td>ECC Polynominal Error: During the controller's internal diagnostic tests, the hardware ECC generator failed its test.</td>
</tr>
</tbody>
</table>
Data Register

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

<table>
<thead>
<tr>
<th>Bits</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte 0</td>
<td>Command Class</td>
<td>Opcode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 1</td>
<td>0</td>
<td>0</td>
<td>d</td>
<td>Head Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 2</td>
<td>Cylinder High</td>
<td>Sector Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 3</td>
<td>Cylinder Low</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 4</td>
<td>Interleave or Block Count</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Byte 5</td>
<td>Control Field</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

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

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

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

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

Byte 5 – Bits 0 through 7 contain the control field.
Control Byte

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

<table>
<thead>
<tr>
<th>Bits</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>r</td>
<td>a</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>s</td>
<td>s</td>
<td>s</td>
</tr>
</tbody>
</table>

Remarks

- \( r \) = retries
- \( s \) = step option
- \( a \) = retry option on data ECC error

Bit 7

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

Bit 6

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

Bits 5, 4, 3

Set to 0.

Bits 2, 1, 0

These bits define the type of drive and select the step option. See the following figure.

<table>
<thead>
<tr>
<th>Bits 2, 1, 0</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
<td>This drive is not specified and defaults to 3 milliseconds per step.</td>
</tr>
<tr>
<td>0 0 1</td>
<td>N/A</td>
</tr>
<tr>
<td>0 1 0</td>
<td>N/A</td>
</tr>
<tr>
<td>0 1 1</td>
<td>N/A</td>
</tr>
<tr>
<td>1 0 0</td>
<td>200 microseconds per step.</td>
</tr>
<tr>
<td>1 0 1</td>
<td>70 microseconds per step (specified by BIOS).</td>
</tr>
<tr>
<td>1 1 0</td>
<td>3 milliseconds per step.</td>
</tr>
<tr>
<td>1 1 1</td>
<td>3 milliseconds per step.</td>
</tr>
</tbody>
</table>

1-186 Fixed Disk Adapter
# Command Summary

<table>
<thead>
<tr>
<th>Command</th>
<th>Data Control Block</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test Drive</td>
<td>Bit 7 6 5 4 3 2 1 0</td>
<td>d = drive (0 or 1)</td>
</tr>
<tr>
<td></td>
<td>Byte 0 0 0 0 0 0 0 0</td>
<td>x = don’t care</td>
</tr>
<tr>
<td>(Class 0, Opcode 00)</td>
<td>Byte 1 0 0 d x x x x</td>
<td>Bytes 2, 3, 4, 5 = don’t care</td>
</tr>
<tr>
<td>Recalibrate</td>
<td>Bit 7 6 5 4 3 2 1 0</td>
<td>d = drive (0 or 1)</td>
</tr>
<tr>
<td>(Class 0, Opcode 01)</td>
<td>Byte 0 0 0 0 0 0 0 1</td>
<td>x = don’t care</td>
</tr>
<tr>
<td></td>
<td>Byte 1 0 0 d x x x x</td>
<td>r = retries</td>
</tr>
<tr>
<td></td>
<td>Byte 5 r 0 0 0 0 s s s</td>
<td>s = Step Option</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Bytes 2, 3, 4 = don’t care</td>
</tr>
<tr>
<td>Reserved</td>
<td>Bit 7 6 5 4 3 2 1 0</td>
<td>This Opcode is not used.</td>
</tr>
<tr>
<td>(Class 0, Opcode 02)</td>
<td>Byte 0 0 0 0 0 0 1 1</td>
<td>d = drive (0 or 1)</td>
</tr>
<tr>
<td>Request Sense</td>
<td>Byte 1 0 0 d x x x x</td>
<td>x = don’t care</td>
</tr>
<tr>
<td>Status</td>
<td></td>
<td>Bytes 2, 3, 4, 5 = don’t care</td>
</tr>
<tr>
<td>(Class 0, Opcode 03)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Format Drive</td>
<td>Bit 7 6 5 4 3 2 1 0</td>
<td>d = drive (0 or 1)</td>
</tr>
<tr>
<td>(Class 0, Opcode 04)</td>
<td>Byte 0 0 0 0 1 0 0 0</td>
<td>r = retries</td>
</tr>
<tr>
<td></td>
<td>Byte 1 0 0 d Head Number</td>
<td>s = step option</td>
</tr>
<tr>
<td></td>
<td>Byte 2 ch 0 0 0 0 0 0</td>
<td>ch = cylinder high</td>
</tr>
<tr>
<td></td>
<td>Byte 3 Cylinder Low</td>
<td>Interleave: 1 to 16 for 512-byte sectors</td>
</tr>
<tr>
<td></td>
<td>Byte 4 0 0 Interleave</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Byte 5 r 0 0 0 0 s s s</td>
<td></td>
</tr>
<tr>
<td>Ready Verify</td>
<td>Bit 7 6 5 4 3 2 1 0</td>
<td>d = drive (0 or 1)</td>
</tr>
<tr>
<td>(Class 0, Opcode 05)</td>
<td>Byte 0 0 0 0 1 0 1</td>
<td>r = retries</td>
</tr>
<tr>
<td></td>
<td>Byte 1 0 0 d Head Number</td>
<td>s = step option</td>
</tr>
<tr>
<td></td>
<td>Byte 2 ch Sector Number</td>
<td>a = retry option on data ECC</td>
</tr>
<tr>
<td></td>
<td>Byte 3 Cylinder Low</td>
<td>ch = cylinder high</td>
</tr>
<tr>
<td></td>
<td>Byte 4 Block Count</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Byte 5 r a 0 0 0 s s s</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Data Control Block</td>
<td>Remarks</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Format Track (Class 0, Opcode 06) | Bit 7 6 5 4 3 2 1 0  
Byte 0 0 0 0 0 1 1 0  
Byte 1 0 0 d Head Number  
Byte 2 ch 0 0 0 0 0 0  
Byte 3 Cylinder Low  
Byte 4 0 0 0 Interleave  
Byte 5 r 0 0 0 0 s s s | d = drive (0 or 1)  
r = retries  
s = step option  
ch = cylinder high  
Interleave: 1 to 16 for 512-byte sectors |
| Format Bad Track (Class 0, Opcode 07) | Bit 7 6 5 4 3 2 1 0  
Byte 0 0 0 0 0 1 1 1  
Byte 1 0 0 d Head Number  
Byte 2 ch 0 0 0 0 0 0  
Byte 3 Cylinder Low  
Byte 4 0 0 0 Interleave  
Byte 5 r 0 0 0 0 s s s | d = drive (0 or 1)  
r = retries  
s = step option  
ch = cylinder high  
Interleave: 1 to 16 for 512-byte sectors |
| Read (Class 0, Opcode 08) | Bit 7 6 5 4 3 2 1 0  
Byte 0 0 0 0 0 1 0 0 0  
Byte 1 0 0 d Head Number  
Byte 2 ch Sector Number  
Byte 3 Cylinder Low  
Byte 5 r a 0 0 0 s s s | d = drive (0 or 1)  
r = retries  
a = retry option on data ECC error  
s = step option  
ch = cylinder high  |
| Reserved (Class 0, Opcode 09) | This Opcode is not used | |
| Write (Class 0, Opcode 0A) | Bit 7 6 5 4 3 2 1 0  
Byte 0 0 0 0 0 1 0 1 0  
Byte 1 0 0 d Head Number  
Byte 2 ch Sector Number  
Byte 3 Cylinder Low  
Byte 4 Block Count  
Byte 5 r 0 0 0 0 s s s | d = drive (0 or 1)  
r = retries  
s = step option  
ch = cylinder high  |
| Seek (Class 0, Opcode 0B) | Bit 7 6 5 4 3 2 1 0  
Byte 0 0 0 0 0 1 0 1 1  
Byte 1 0 0 d Head Number  
Byte 2 ch 0 0 0 0 0 0  
Byte 3 Cylinder Low  
Byte 4 x x x x x x  
Byte 5 r 0 0 0 0 s s s | d = drive (0 or 1)  
r = retries  
s = step option  
x = don’t care  
ch = cylinder high  |
<table>
<thead>
<tr>
<th>Command</th>
<th>Data Control Block</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialize Drive</td>
<td>Bit 0 0 0 1 1 1 0 0</td>
<td>Bytes 1, 2, 3, 4, 5 = don't care</td>
</tr>
<tr>
<td>Characteristics*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Class 0, Opcode 0C)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read ECC Burst Error</td>
<td>Bit 0 0 0 1 1 1 0 1</td>
<td>Bytes 1, 2, 3, 4, 5 = don't care</td>
</tr>
<tr>
<td>Length (Class 0, Opcode 0D)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read Data from Sector</td>
<td>Bit 0 0 0 1 1 1 0 0</td>
<td>Bytes 1, 2, 3, 4, 5 = don't care</td>
</tr>
<tr>
<td>Buffer (Class 0, Opcode 0E)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Write Data to Sector</td>
<td>Bit 0 0 0 1 1 1 1 1</td>
<td>Bytes 1, 2, 3, 4, 5 = don't care</td>
</tr>
<tr>
<td>Buffer (Class 0, Opcode 0F)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RAM Diagnostic</td>
<td>Bit 0 1 1 0 0 0 0 0</td>
<td>Bytes 1, 2, 3, 4, 5 = don't care</td>
</tr>
<tr>
<td>(Class 7, Opcode 00)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td></td>
<td>This Opcode is not used</td>
</tr>
<tr>
<td>(Class 7, Opcode 01)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td></td>
<td>This Opcode is not used</td>
</tr>
<tr>
<td>(Class 7, Opcode 02)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

- Maximum number of cylinders (2 bytes)
- Maximum number of heads (1 byte)
- Start reduced write current cylinder (2 bytes)
- Start write precompensation cylinder (2 bytes)
- Maximum ECC data burst length (1 byte)
<table>
<thead>
<tr>
<th>Command</th>
<th>Data Control Block</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drive</td>
<td>Bit</td>
<td>d = drive (0 or 1)</td>
</tr>
<tr>
<td>Diagnostic</td>
<td>Byte 0</td>
<td>s = step option</td>
</tr>
<tr>
<td>(Class 7, Opcode 03)</td>
<td>Byte 1</td>
<td>r = retries</td>
</tr>
<tr>
<td></td>
<td>Byte 2</td>
<td>x = don’t care</td>
</tr>
<tr>
<td></td>
<td>Byte 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Byte 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Byte 5</td>
<td></td>
</tr>
<tr>
<td>Controller</td>
<td>Bit</td>
<td>Bytes 1, 2, 3, 4, 5 = don’t care</td>
</tr>
<tr>
<td>Internal</td>
<td>Byte 0</td>
<td></td>
</tr>
<tr>
<td>Diagnostics</td>
<td>Byte 0</td>
<td></td>
</tr>
<tr>
<td>(Class 7, Opcode 04)</td>
<td>Byte 0</td>
<td></td>
</tr>
<tr>
<td>Read Long*</td>
<td>Byte 0</td>
<td>d = (0 or 1)</td>
</tr>
<tr>
<td>(Class 7, Opcode 05)</td>
<td>Byte 1</td>
<td>s = step option</td>
</tr>
<tr>
<td></td>
<td>Byte 2</td>
<td>r = retries</td>
</tr>
<tr>
<td></td>
<td>Byte 3</td>
<td>ch = cylinder high</td>
</tr>
<tr>
<td></td>
<td>Byte 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Byte 5</td>
<td></td>
</tr>
<tr>
<td>Write Long**</td>
<td>Bit</td>
<td>d = drive (0 or 1)</td>
</tr>
<tr>
<td>(Class 7, Opcode 06)</td>
<td>Byte 0</td>
<td>s = step option</td>
</tr>
<tr>
<td></td>
<td>Byte 1</td>
<td>r = retries</td>
</tr>
<tr>
<td></td>
<td>Byte 2</td>
<td>ch = cylinder high</td>
</tr>
<tr>
<td></td>
<td>Byte 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Byte 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Byte 5</td>
<td></td>
</tr>
</tbody>
</table>

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

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

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

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

<table>
<thead>
<tr>
<th>R/W</th>
<th>Port Address</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read Write</td>
<td>320</td>
<td>Read data (from controller to system unit). Write data (from system unit to controller).</td>
</tr>
<tr>
<td></td>
<td>320</td>
<td></td>
</tr>
<tr>
<td>Read Write</td>
<td>321</td>
<td>Read controller hardware status.</td>
</tr>
<tr>
<td></td>
<td>321</td>
<td>Controller reset.</td>
</tr>
<tr>
<td>Read Write</td>
<td>322</td>
<td>Reserved.</td>
</tr>
<tr>
<td></td>
<td>322</td>
<td>Generate controller-select pulse.</td>
</tr>
<tr>
<td>Read Write</td>
<td>323</td>
<td>Not used.</td>
</tr>
<tr>
<td></td>
<td>323</td>
<td>Write pattern to DMA and interrupt mask register.</td>
</tr>
</tbody>
</table>
System I/O Channel Interface

The following lines are used by the disk controller:

AO-A19  Positive true 20-bit address. The least-significant 10 bits contain the I/O address within the range of hex 320 to hex 323 when an I/O read or write is executed by the system unit. The full 20 bits are decoded to address the read-only storage (ROS) between the addresses of hex C8000 and C9FFF.

DO-D7  Positive 8-bit data bus over which data and status information is passed between the system board and the controller.

IOR  Negative true signal that is asserted when the system board reads status or data from the controller under either programmed I/O or DMA control.

IOW  Negative true signal that is asserted when the system board sends a command or data to the controller under either programmed I/O or DMA control.

AEN  Positive true signal that is asserted when the DMA in the system board is generating the I/O Read (IOR) or I/O Write (IOW) signals and has control of the address and data buses.

RESET  Positive true signal that forces the disk controller to its initial power-up condition.

IRQ 5  Positive true interrupt request signal that is asserted by the controller when enabled to interrupt the system board on the return ending status byte from the controller.

DRQ 3  Positive true DMA-request signal that is asserted by the controller when data is available for transfer to or from the controller under DMA control. This signal remains active until the system board's DMA channel activates the DMA-acknowledge signal (DACK 3) in response.

DACK 3  This signal is true when negative, and is generated by the system board DMA channel in response to a DMA request (DRQ 3).
### Fixed Disk Adapter Interface Specifications

#### Disk Drive Connector J1

<table>
<thead>
<tr>
<th>Signal</th>
<th>Pin Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ground - Odd Numbers</td>
<td>1-33</td>
</tr>
<tr>
<td>Reserved</td>
<td>4, 16, 30, 32</td>
</tr>
<tr>
<td>- Reduced Write Current</td>
<td>2</td>
</tr>
<tr>
<td>- Write Gate</td>
<td>6</td>
</tr>
<tr>
<td>- Seek Complete</td>
<td>8</td>
</tr>
<tr>
<td>- Track 00</td>
<td>10</td>
</tr>
<tr>
<td>- Write Fault</td>
<td>12</td>
</tr>
<tr>
<td>- Head Select 2&lt;sup&gt;0&lt;/sup&gt;</td>
<td>14</td>
</tr>
<tr>
<td>- Head Select 2&lt;sup&gt;1&lt;/sup&gt;</td>
<td>18</td>
</tr>
<tr>
<td>- Index</td>
<td>20</td>
</tr>
<tr>
<td>- Ready</td>
<td>22</td>
</tr>
<tr>
<td>- Step</td>
<td>24</td>
</tr>
<tr>
<td>- Drive Select 1</td>
<td>26</td>
</tr>
<tr>
<td>- Drive Select 2</td>
<td>28</td>
</tr>
<tr>
<td>- Direction In</td>
<td>34</td>
</tr>
</tbody>
</table>

#### Disk Drive Connector J2 or J3

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

---

*Hardware*
IBM 10MB Fixed Disk Drive

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

An impact-resistant enclosure provides mechanical and contamination protection for the heads, actuator, and disks. A self-contained recirculating system supplies clean air through a 0.3-micron filter. Thermal isolation of the stepper and spindle motor assemblies from the disk enclosure results in a very low temperature rise within the enclosure. This isolation provides a greater off-track margin and the ability to perform read and write operations immediately after power-up with no thermal stabilization delay.
<table>
<thead>
<tr>
<th>Media</th>
<th>Rigid media disk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Tracks</td>
<td>1224</td>
</tr>
<tr>
<td>Track Density</td>
<td>345 tracks per inch</td>
</tr>
<tr>
<td>Dimensions</td>
<td></td>
</tr>
<tr>
<td>Height</td>
<td>3.25 inches (82.55 mm)</td>
</tr>
<tr>
<td>Width</td>
<td>5.75 inches (146.05 mm)</td>
</tr>
<tr>
<td>Depth</td>
<td>8.0 inches (203.2 mm)</td>
</tr>
<tr>
<td>Weight</td>
<td>4.6 lb (2.08 kg)</td>
</tr>
<tr>
<td>Temperature</td>
<td></td>
</tr>
<tr>
<td>Operating</td>
<td>40°F to 122°F (4°C to 50°C)</td>
</tr>
<tr>
<td>Non operating</td>
<td>-40°F to 140°F (-40°C to 60°C)</td>
</tr>
<tr>
<td>Relative Humidity</td>
<td></td>
</tr>
<tr>
<td>Operating</td>
<td>8% to 80% (non condensing)</td>
</tr>
<tr>
<td>Maximum Wet Bulb</td>
<td>78°F (26°C)</td>
</tr>
<tr>
<td>Shock</td>
<td></td>
</tr>
<tr>
<td>Operating</td>
<td>10 Gs</td>
</tr>
<tr>
<td>Non operating</td>
<td>20 Gs</td>
</tr>
<tr>
<td>Access Time</td>
<td>3 ms track-to-track</td>
</tr>
<tr>
<td>Average Latency</td>
<td>8.33 ms</td>
</tr>
<tr>
<td>Error Rates</td>
<td></td>
</tr>
<tr>
<td>Soft Read Errors</td>
<td>1 per 10¹⁶ bits read</td>
</tr>
<tr>
<td>Hard Read Errors</td>
<td>1 per 10¹² bits read</td>
</tr>
<tr>
<td>Seek Errors</td>
<td>1 per 10⁶ seeks</td>
</tr>
<tr>
<td>Design Life</td>
<td>5 years (8,000 hours MTF)</td>
</tr>
<tr>
<td>Disk Speed</td>
<td>3600 rpm ±1%</td>
</tr>
<tr>
<td>Transfer Rate</td>
<td>5.0 M bits/sec</td>
</tr>
<tr>
<td>Recording Mode</td>
<td>MFM</td>
</tr>
<tr>
<td>Power</td>
<td>+12 Vdc ± 5% 1.8 A (4.5 A maximum)</td>
</tr>
<tr>
<td></td>
<td>+5 Vdc ± 5% 0.7 A (1.0 A maximum)</td>
</tr>
<tr>
<td>Maximum Ripple</td>
<td>1% with equivalent resistive load</td>
</tr>
</tbody>
</table>

**Mechanical and Electrical Specifications**

1-196 Fixed Disk Drive
Three memory expansion options and a memory module kit are available for the IBM Personal Computer XT. They are the 32KB, 64KB, and 64/256K Memory Expansion Options and the 64KB Memory Module Kit. The base system has a standard 128K of RAM on the system board. One or two memory module kits can be added, providing the system board with 192K or 256K of RAM. The base 64/256K option has a standard 64K of RAM. One, two, or three 64K memory module kits may be added, providing the 64/256K option with 128K, 192K, or 256K of RAM. A maximum of 256K or RAM can be installed on the system board as modules without using any of the system unit expansion slots or expansion options. The system board must be populated to the maximum 256K of RAM before any memory expansion options can be installed.

An expansion option must be configured to reside at a sequential 32K or 64K memory address boundary within the system address space. This is done by setting DIP switches on the option.

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

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

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

Dynamic-memory refresh timing and address generation are functions that are performed on the system board and made available in the I/O channel for all devices.
To allow the system to address 32K, 64K, or 64/256K memory expansion options, refer to "Appendix G: Switch Settings" for the proper memory expansion option switch settings.

**Operating Characteristics**

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

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

General specifications for memory used on all cards are:

<table>
<thead>
<tr>
<th></th>
<th>16K by 1 Bit</th>
<th>32K by 1 Bit</th>
<th>64K by 1 Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access</td>
<td>250 ns</td>
<td>250 ns</td>
<td>200 ns</td>
</tr>
<tr>
<td>Cycle</td>
<td>410 ns</td>
<td>410 ns</td>
<td>345 ns</td>
</tr>
</tbody>
</table>

**Memory Module Description**

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

The 64/256K option has four banks of 9 pluggable sockets. Each bank will accept a 64K memory module kit, consisting of 9 (64K by 1) modules. The kits must be installed sequentially into banks 1, 2, and 3. The base 64/256K option comes with modules installed in bank 0, providing 64K of memory. One, two, or three 64K bits may be added, upgrading the option to 128K, 192K, or 256K of memory.
The 16K by 1 and the 32K by 1 modules require three voltage levels: +5 Vdc, -5 Vdc, and +12 Vdc. The 64K by 1 modules require only one voltage level of +5 Vdc. All three memory modules require 128 refresh cycles every 2 ns. Absolute maximum access times are:

<table>
<thead>
<tr>
<th></th>
<th>16K by 1 Bit</th>
<th>32K by 1 Bit</th>
<th>64K by 1 Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>From RAS</td>
<td>250 ns</td>
<td>250 ns</td>
<td>200 ns</td>
</tr>
<tr>
<td>From CAS</td>
<td>165 ns</td>
<td>165 ns</td>
<td>115 ns</td>
</tr>
</tbody>
</table>

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

*16K by 1 and 64K by 1 bit modules have 16 pins.
**Data In and Data Out are tied together (three-state bus).
***Data In and Data Out are tied together on Data Bits 0-7 (three-state bus).

Memory Module Pin Configuration

Memory Expansion Options   1-199
Switch-Configurable Start Address

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

<table>
<thead>
<tr>
<th>Number</th>
<th>32K and 64K Options</th>
<th>64/256K Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ON: A19=0; OFF: A19=1</td>
<td>ON: A19=0; OFF: A19=1</td>
</tr>
<tr>
<td>2</td>
<td>ON: A18=0; OFF: A18=1</td>
<td>ON: A18=0; OFF: A18=1</td>
</tr>
<tr>
<td>3</td>
<td>ON: A17=0; OFF: A17=1</td>
<td>ON: A17=0; OFF: A17=1</td>
</tr>
<tr>
<td>4</td>
<td>ON: A16=0; OFF: A16=1</td>
<td>ON: A16=0; OFF: A16=1</td>
</tr>
<tr>
<td>5</td>
<td>ON: A15=0; OFF: A15=1*</td>
<td>ON: Select 64K</td>
</tr>
<tr>
<td>6</td>
<td>Not used</td>
<td>ON: Select 128K</td>
</tr>
<tr>
<td>7</td>
<td>Not used</td>
<td>ON: Select 192K</td>
</tr>
<tr>
<td>8</td>
<td>Used only in 64K RAM Card*</td>
<td>ON: Select 256K</td>
</tr>
</tbody>
</table>

*Switch 8 may be set on the 64K memory expansion option to use only half the memory on the card (that is, 32K). If switch 8 is on, all 64K is accessible. If switch 8 is off, address bit A15 (as set by switch 5) is used to determine which 32K are accessible, and the 64K option behaves as a 32K option.

DIP Module Start Address

Memory Option Switch Settings

Switch settings for all memory expansion options are located in “Appendix G: Switch Settings.”
The following method can be used to determine the switch settings for the 32K memory expansion option.

Starting Address = xxxK

\[
\begin{array}{c}
\text{32K} \\
\frac{xxxK}{= \text{Decimal value}}
\end{array}
\]

Convert decimal value to binary

Bit: \ldots 4 3 2 1 0
Bit value: \ldots 16 8 4 2 1

Switch

![Switch Diagram]

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

Starting Address = xxxK

\[
\begin{array}{c}
\text{64K} \\
\frac{xxxK}{= \text{Decimal value}}
\end{array}
\]

Convert decimal value to binary

Bit: \ldots 3 2 1 0
Bit value: \ldots 8 4 2 1

Switch

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

Starting Address = xxxK

= Decimal value

64K \[xxxK\]

Convert decimal value to binary

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

Switch

Amount of memory installed on option
256K
192K (on = logical 1)
128K
64K

bit
0
1
2 (off = logical 1)
3

1-202 Memory Expansion Options
IBM Game Control Adapter

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

Game Control Adapter Block Diagram
Functional Description

Address Decode

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

Data Bus Buffer/Driver

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

Trigger Buttons

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

Joy Stick Positions

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

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

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

A9-AO: Address lines 9 through 0 are used to address the game control adapter.

D7-DO: Data lines 7 through 0 are the data bus.

IOR, IOW: I/O read and I/O write are used when reading from or writing to an adapter (In, Out).

AEN: When active, the adapter must be inactive and the data bus driver inactive.

+5 Vdc: Power for the game control adapter.

GND: Common ground.

A9-A10: Unused.

MEMR, MEMW: Unused.

DACKO-DACK3: Unused.

IRQ7-IRQ2: Unused.

DRQ3-DRQ1: Unused.

ALE, T/C: Unused.

CLK, OSC: Unused.

I/O CHCK: Unused.

I/O CH RDY: Unused.

RESET DRV: Unused.

−5 Vdc, +12 Vdc, −12 Vdc: Unused.
Interface Description

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

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

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

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

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

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

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

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>B-#2 Button</td>
<td>B-#1 Button</td>
<td>A-#2 Button</td>
<td>A-#1 Button</td>
<td>B-Y Coordinate</td>
<td>B-X Coordinate</td>
<td>A-Y Coordinate</td>
<td>A-X Coordinate</td>
</tr>
</tbody>
</table>

1-206 Game Control Adapter
The game paddles will have a set of two (A and B) or four (A, B, C, and D) paddles. These will have one button each and one variable resistance each, with a range of 0 to 100 k-ohms. This should be attached to give the following input data:

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>C</td>
<td>B</td>
<td>A</td>
<td>D</td>
<td>C</td>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td>Button</td>
<td>Button</td>
<td>Button</td>
<td>Button</td>
<td>Coordinate</td>
<td>Coordinate</td>
<td>Coordinate</td>
<td>Coordinate</td>
</tr>
</tbody>
</table>

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

15-Pin Male D-Shell Connector

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

Joy Stick Schematic Diagram
At Standard TTL Levels

<table>
<thead>
<tr>
<th>Voltage</th>
<th>Adapter Pin No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>+5 Vdc</td>
<td>1</td>
</tr>
<tr>
<td>Button 4</td>
<td>2</td>
</tr>
<tr>
<td>Position 0</td>
<td>3</td>
</tr>
<tr>
<td>Ground</td>
<td>4</td>
</tr>
<tr>
<td>Ground</td>
<td>5</td>
</tr>
<tr>
<td>Position 1</td>
<td>6</td>
</tr>
<tr>
<td>Button 5</td>
<td>7</td>
</tr>
<tr>
<td>+5 Vdc</td>
<td>8</td>
</tr>
<tr>
<td>Button 6</td>
<td>9</td>
</tr>
<tr>
<td>Position 2</td>
<td>10</td>
</tr>
<tr>
<td>Ground</td>
<td>11</td>
</tr>
<tr>
<td>Position 3</td>
<td>12</td>
</tr>
<tr>
<td>Button 7</td>
<td>13</td>
</tr>
<tr>
<td>+5 Vdc</td>
<td>14</td>
</tr>
</tbody>
</table>

External Devices

Game Control Adapter

Connector Specifications

1-208 Game Control Adapter
IBM Prototype Card

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

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

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

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

1-210 Prototype Card
I/O Channel Interface

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

Prototype Card Layout

The component side has the ground bus [0.05 inch (1.27 millimeters) wide] screened on it and card-edge tabs that are labeled A1 through A31.
The component side also has a silk screen printed on it that is used as a component guide for the I/O interface.

Component Side

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

Pin Side

1-212 Prototype Card
Each card-edged tab is connected to a plated through-hole by a 0.012-inch (0.3-millimeter) land. There are three ground tabs connected to the ground bus by three 0.012-inch (0.3 millimeter) lands. Also, there are two +5 Vdc tabs connected to the voltage bus by two 0.012-inch (0.3 millimeter) lands.

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

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

**System Loading and Power Limitations**

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

Refer to the power supply information in this manual for the power limitations to be observed.
Prototype Card External Interface

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

<table>
<thead>
<tr>
<th>Connector Size</th>
<th>Part Number (Amp)</th>
</tr>
</thead>
<tbody>
<tr>
<td>9-pin D-shell (Male)</td>
<td>205865-1</td>
</tr>
<tr>
<td>9-pin D-shell (Female)</td>
<td>205866-1</td>
</tr>
<tr>
<td>15-pin D-shell (Male)</td>
<td>205867-1</td>
</tr>
<tr>
<td>15-pin D-shell (Female)</td>
<td>205868-1</td>
</tr>
<tr>
<td>25-pin D-shell (Male)</td>
<td>205857-1</td>
</tr>
<tr>
<td>25-pin D-shell (Female)</td>
<td>205858-1</td>
</tr>
<tr>
<td>37-pin D-shell (Male)</td>
<td>205859-1</td>
</tr>
<tr>
<td>37-pin D-shell (Female)</td>
<td>205860-1</td>
</tr>
</tbody>
</table>

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

![Diagram of a 15-pin D-shell female connector attached to a prototype card.](image)

Component Side

1-214  Prototype Card
IBM Asynchronous Communications Adapter

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

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

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

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

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

**Modes of Operation**

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

<table>
<thead>
<tr>
<th>Primary Adapter</th>
<th>Alternate Adapter</th>
<th>Register Selected</th>
<th>DLAB State</th>
</tr>
</thead>
<tbody>
<tr>
<td>3F8</td>
<td>2F8</td>
<td>TX Buffer</td>
<td>DLAB=0 (Write)</td>
</tr>
<tr>
<td>3F8</td>
<td>2F8</td>
<td>RX Buffer</td>
<td>DLAB=0 (Read)</td>
</tr>
<tr>
<td>3F8</td>
<td>2F8</td>
<td>Divisor Latch LSB</td>
<td>DLAB=1</td>
</tr>
<tr>
<td>3F9</td>
<td>2F9</td>
<td>Divisor Latch MSB</td>
<td>DLAB=1</td>
</tr>
<tr>
<td>3F9</td>
<td>2F9</td>
<td>Interrupt Enable Register</td>
<td>DLAB=1</td>
</tr>
<tr>
<td>3FA</td>
<td>3FA</td>
<td>Interrupt Identification Registers</td>
<td></td>
</tr>
<tr>
<td>3FB</td>
<td>2FB</td>
<td>Line Control Register</td>
<td></td>
</tr>
<tr>
<td>3FC</td>
<td>2FC</td>
<td>Modem Control Register</td>
<td></td>
</tr>
<tr>
<td>3FD</td>
<td>2FD</td>
<td>Line Status Register</td>
<td></td>
</tr>
<tr>
<td>3FE</td>
<td>2FE</td>
<td>Modem Status Register</td>
<td></td>
</tr>
</tbody>
</table>

### I/O Decodes

### Hex Address 3F8 to 3FF and 2F8 to 2FF

<table>
<thead>
<tr>
<th>A9</th>
<th>A8</th>
<th>A7</th>
<th>A6</th>
<th>A5</th>
<th>A4</th>
<th>A3</th>
<th>A2</th>
<th>A1</th>
<th>A0</th>
<th>DLAB</th>
<th>Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1/0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>0</td>
<td>Receive Buffer (read), Transmit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Holding Reg. (write)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>Interrupt Enable</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>1</td>
<td>Interrupt Identification</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>1</td>
<td>Line Control</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0</td>
<td>Modem Control</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0</td>
<td>Line Status</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>Modem Status</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>Divisor Latch (LSB)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>1</td>
<td>Divisor Latch (MSB)</td>
</tr>
</tbody>
</table>

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

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

### Address Bits
Interrupts

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

The data format will be as follows:

```
<table>
<thead>
<tr>
<th>Transmit Data Marking</th>
<th>Start Bit</th>
<th>D0</th>
<th>D1</th>
<th>D2</th>
<th>D3</th>
<th>D4</th>
<th>D5</th>
<th>D6</th>
<th>D7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Parity Bit</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Stop Bit</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

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

Interface Description

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

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

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

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

<table>
<thead>
<tr>
<th>Pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Transmitted Data</td>
</tr>
<tr>
<td>3</td>
<td>Received Data</td>
</tr>
<tr>
<td>4</td>
<td>Request to Send</td>
</tr>
<tr>
<td>5</td>
<td>Clear to Send</td>
</tr>
<tr>
<td>6</td>
<td>Data Set Ready</td>
</tr>
<tr>
<td>7</td>
<td>Signal Ground</td>
</tr>
<tr>
<td>8</td>
<td>Carrier Detect</td>
</tr>
<tr>
<td>20</td>
<td>Data Terminal Ready</td>
</tr>
<tr>
<td>22</td>
<td>Ring Indicator</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Interchange Voltage</th>
<th>Binary State</th>
<th>Signal Condition</th>
<th>Interface Control Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive Voltage</td>
<td>Binary (0)</td>
<td>= Spacing</td>
<td>= On</td>
</tr>
<tr>
<td>Negative Voltage</td>
<td>Binary (1)</td>
<td>= Marking</td>
<td>= Off</td>
</tr>
</tbody>
</table>

Invalid Levels

+15 Vdc
On Function

+3 Vdc
0 Vdc
Invalid Levels

-3 Vdc
Off Function

-15 Vdc
Invalid Levels

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

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

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

1-220 Asynchronous Adapter
INS8250 Functional Pin Description

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

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

Input Signals

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

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

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

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

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

Address Strobe (ADS), Pin 25: When low, provides latching for the register select (A0, A1, A2) and chip select (CS0, CS1, \(\overline{CS2}\)) signals.
Note: An active $\overline{ADS}$ input is required when the register select (A0, A1, A2) signals are not stable for the duration of a read or write operation. If not required, tie the $\overline{ADS}$ input permanently low.

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

<table>
<thead>
<tr>
<th>DLAB</th>
<th>A2</th>
<th>A1</th>
<th>A0</th>
<th>Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Receiver Buffer (Read), Transmitter Holding Register (Write)</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Interrupt Enable</td>
</tr>
<tr>
<td>X</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Interrupt Identification (Read Only)</td>
</tr>
<tr>
<td>X</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Line Control</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Modem Control</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Line Status</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Modem Status</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>None</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Divisor Latch (Least Significant Bit)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Divisor Latch (Most Significant Bit)</td>
</tr>
</tbody>
</table>

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

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

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

1-222 Asynchronous Adapter
Clear to Send (CTS), Pin 36: The CTS signal is a modem control function input whose condition can be tested by the processor by reading bit 4 (CTS) of the modem status register. Bit 0 (DCTS) of the modem status register indicates whether the CTS input has changed state since the previous reading of the modem status register.

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

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

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

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

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

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

VCC, Pin 40: +5 Vdc supply.

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

Output Signals

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

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

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

Output 2 (OUT 2), Pin 31: User-designated output that can be set to an active low by programming bit 3 (OUT 2) of the modem control register to a high level. The OUT 2 signal is set high upon a master reset operation.
Chip Select Out (CSOUT), Pin 24: When high, indicates that the chip has been selected by active CS0, CS1, and CS2 inputs. No data transfer can be initiated until the CSOUT signal is a logical 1.

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

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

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

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

Input/Output Signals

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

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

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

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

Asynchronous Communications Reset Functions

1-226 Asynchronous Adapter
Line-Control Register

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

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Word Length Select Bit 0 (WLS0)
- Word Length Select Bit 1 (WLS1)
- Number of Stop Bits (STB)
- Parity Enable (PEN)
- Even Parity Select (EPS)
- Stick Parity
- Set Break
- Divisor Latch Access Bit (DLAB)

Line-Control Register (LCR)

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

<table>
<thead>
<tr>
<th>Bit 1</th>
<th>Bit 0</th>
<th>Word Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>5 Bits</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>6 Bits</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>7 Bits</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>8 Bits</td>
</tr>
</tbody>
</table>

Asynchronous Adapter  1-227
**Bit 2:** This bit specifies the number of stop bits in each transmitted or received serial character. If bit 2 is a logical 0, one stop bit is generated or checked in the transmit or receive data, respectively. If bit 2 is logical 1 when a 5-bit word length is selected through bits 0 and 1, 1-1/2 stop bits are generated or checked. If bit 2 is logical 1 when either a 6-, 7-, or 8-bit word length is selected, two stop bits are generated or checked.

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

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

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

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

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

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

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

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

<table>
<thead>
<tr>
<th>Desired Baud Rate</th>
<th>Divisor Used to Generate 16x Clock</th>
<th>Percent Error Difference Between Desired and Actual</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>2304 (Decimal) 900 (Hex)</td>
<td>—</td>
</tr>
<tr>
<td>75</td>
<td>1536 (Decimal) 600 (Hex)</td>
<td>—</td>
</tr>
<tr>
<td>110</td>
<td>1047 (Decimal) 417 (Hex)</td>
<td>0.026</td>
</tr>
<tr>
<td>134.5</td>
<td>857 (Decimal) 359 (Hex)</td>
<td>0.058</td>
</tr>
<tr>
<td>150</td>
<td>768 (Decimal) 300 (Hex)</td>
<td>—</td>
</tr>
<tr>
<td>300</td>
<td>384 (Decimal) 180 (Hex)</td>
<td>—</td>
</tr>
<tr>
<td>600</td>
<td>192 (Decimal) 0C0 (Hex)</td>
<td>—</td>
</tr>
<tr>
<td>1200</td>
<td>96 (Decimal) 060 (Hex)</td>
<td>—</td>
</tr>
<tr>
<td>1800</td>
<td>64 (Decimal) 040 (Hex)</td>
<td>—</td>
</tr>
<tr>
<td>2000</td>
<td>58 (Decimal) 03A (Hex)</td>
<td>0.69</td>
</tr>
<tr>
<td>2400</td>
<td>48 (Decimal) 030 (Hex)</td>
<td>—</td>
</tr>
<tr>
<td>3600</td>
<td>32 (Decimal) 020 (Hex)</td>
<td>—</td>
</tr>
<tr>
<td>4800</td>
<td>24 (Decimal) 018 (Hex)</td>
<td>—</td>
</tr>
<tr>
<td>7200</td>
<td>16 (Decimal) 010 (Hex)</td>
<td>—</td>
</tr>
<tr>
<td>9600</td>
<td>12 (Decimal) 00C (Hex)</td>
<td>—</td>
</tr>
</tbody>
</table>

Baud Rate at 1.843 MHz

1-230 Asynchronous Adapter
Line Status Register

This 8-bit register provides status information on the processor concerning the data transfer. The contents of the line status register are indicated and described below:

<table>
<thead>
<tr>
<th>Hex Address 3FD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7 6 5 4 3 2 1 0</td>
</tr>
<tr>
<td>Data Ready (DR)</td>
</tr>
<tr>
<td>Overrun Error (OR)</td>
</tr>
<tr>
<td>Parity Error (PE)</td>
</tr>
<tr>
<td>Framing Error (FE)</td>
</tr>
<tr>
<td>Break Interrupt (BI)</td>
</tr>
<tr>
<td>Transmitter Holding Register Empty (THRE)</td>
</tr>
<tr>
<td>Tx Shift Register Empty (TSRE)</td>
</tr>
<tr>
<td>= 0</td>
</tr>
</tbody>
</table>

Line Status Register (LSR)

Bit 0: This bit is the receiver data ready (DR) indicator. Bit 0 is set to a logical 1 whenever a complete incoming character has been received and transferred into the receiver buffer register. Bit 0 may be reset to a logical 0 either by the processor reading the data in the receiver buffer register or by writing a logical 0 into it from the processor.

Bit 1: This bit is the overrun error (OE) indicator. Bit 1 indicates that data in the receiver buffer register was not read by the processor before the next character was transferred into the receiver buffer register, thereby destroying the previous character. The OE indicator is reset whenever the processor reads the contents of the line status register.

Bit 2: This bit is the parity error (PE) indicator. Bit 2 indicates that the received data character does not have the correct even or odd parity, as selected by the even parity-select bit. The PE bit is set to a logical 1 upon detection of a parity error and is reset to a logical 0 whenever the processor reads the contents of the line status register.
Bit 3: This bit is the framing error (FE) indicator. Bit 3 indicates that the received character did not have a valid stop bit. Bit 3 is set to a logical 1 whenever the stop bit following the last data bit or parity is detected as a zero bit (spacing level).

Bit 4: This bit is the break interrupt (BI) indicator. Bit 4 is set to a logical 1 whenever the received data input is held in the spacing (logical 0) state for longer than a full word transmission time (that is, the total time of start bit + data bits + parity + stop bits).

Note: Bits 1 through 4 are the error conditions that produce a receiver line status interrupt whenever any of the corresponding conditions are detected.

Bit 5: This bit is the transmitter holding register empty (THRE) indicator. Bit 5 indicates that the INS8250 is ready to accept a new character for transmission. In addition, this bit causes the INS8250 to issue an interrupt to the processor when the transmit holding register empty interrupt enable is set high. The THRE bit is set to a logical 1 when a character is transferred from the transmitter holding register into the transmitter shift register. The bit is reset to logical 0 concurrently with the loading of the transmitter holding register by the processor.

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

Bit 7: This bit is permanently set to logical 0.

Interrupt Identification Register

The INS8250 has an on-chip interrupt capability that allows for complete flexibility in interfacing to all the popular microprocessors presently available. In order to provide minimum software overhead during data character transfers, the INS8250 prioritizes interrupts into four levels: receiver line status (priority 1), received data ready (priority 2), transmitter holding register empty (priority 3), and modem status (priority 4).
Information indicating that a prioritized interrupt is pending and
the type of prioritized interrupt is stored in the interrupt
identification register. Refer to the “Interrupt Control
Functions” table. The interrupt identification register (IIR), when
addressed during chip-select time, freezes the highest priority
interrupt pending, and no other interrupts are acknowledged until
that particular interrupt is serviced by the processor. The contents
of the IIR are indicated and described below.

<table>
<thead>
<tr>
<th>Hex Address 3FA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7 6 5 4 3 2 1 0</td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>0 If Interrupt Pending</td>
</tr>
<tr>
<td>Interrupt ID Bit (0)</td>
</tr>
<tr>
<td>Interrupt ID Bit (1)</td>
</tr>
<tr>
<td>= 0</td>
</tr>
<tr>
<td>= 0</td>
</tr>
<tr>
<td>= 0</td>
</tr>
<tr>
<td>= 0</td>
</tr>
<tr>
<td>= 0</td>
</tr>
</tbody>
</table>

**Interrupt Identification Register (IIR)**

**Bit 0:** This bit can be used in either a hard-wired prioritized or
pooled environment to indicate whether an interrupt is pending and
the IIR contents may be used as a pointer to the appropriate
interrupt service routine When bit 0 is a logical 1, no interrupt is
pending and polling (if used) is continued.

**Bits 1 and 2:** These two bits of the IIR are used to identify the
highest priority interrupt pending as indicated in the “Interrupt
Control Functions” table.

**Bits 3 through 7:** These five bits of the IIR are always logical 0.
<table>
<thead>
<tr>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>Priority Level</th>
<th>Interrupt Type</th>
<th>Interrupt Source</th>
<th>Interrupt Reset Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>Highest</td>
<td>Receiver Line Status</td>
<td>Overrun Error or Parity Error or Framing Error or Break Interrupt</td>
<td>Reading the Line Status Register</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Second</td>
<td>Received Data Available</td>
<td>Receiver Data Available</td>
<td>Reading the Receiver Buffer Register</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Third</td>
<td>Transmitter Holding Register Empty</td>
<td>Transmitter Holding Register Empty</td>
<td>Reading the IIR Register (if source of interrupt) or Writing into the Transmitter Holding Register</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Fourth</td>
<td>Modem Status</td>
<td>Clear to Send or Data Set Ready or Ring Indicator or Received Line Signal Direct</td>
<td>Reading the Modem Status Register</td>
</tr>
</tbody>
</table>

**Interrupt Control Functions**

1-234 Asynchronous Adapter
Interrupt Enable Register

This eight-bit register enables the four types of interrupt of the INS8250 to separately activate the chip interrupt (INTRPT) output signal. It is possible to totally disable the interrupt system by resetting bits 0 through 3 of the interrupt enable register. Similarly, by setting the appropriate bits of this register to a logical 1, selected interrupts can be enabled. Disabling the interrupt system inhibits the interrupt identification register and the active (high) INTRPT output from the chip. All other system functions operate in their normal manner, including the setting of the line status and modem status registers. The contents of the interrupt enable register are indicated and described below:

<table>
<thead>
<tr>
<th>Hex Address 3F9</th>
<th>DLAB = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>7 6 5 4 3 2 1 0</td>
</tr>
<tr>
<td></td>
<td>1 = Enable Data Available Interrupt</td>
</tr>
<tr>
<td></td>
<td>1 = Enable Tx Holding Register Empty Interrupt</td>
</tr>
<tr>
<td></td>
<td>1 = Enable Receive Line Status Interrupt</td>
</tr>
<tr>
<td></td>
<td>1 = Enable Modem Status Interrupt</td>
</tr>
<tr>
<td></td>
<td>= 0</td>
</tr>
<tr>
<td></td>
<td>= 0</td>
</tr>
<tr>
<td></td>
<td>= 0</td>
</tr>
<tr>
<td></td>
<td>= 0</td>
</tr>
</tbody>
</table>

Interrupt Enable Register (IER)

Bit 0: This bit enables the received data available interrupt when set to logical 1.

Bit 1: This bit enables the transmitter holding register empty interrupt when set to logical 1.

Bit 2: This bit enables the receiver line status interrupt when set to logical 1.
Bit 3: This bit enables the modem status interrupt when set to logical 1.

Bits 4 through 7: These four bits are always logical 0.

Modem Control Register

This eight-bit register controls the interface with the modem or data set (or peripheral device emulating a modem). The contents of the modem control register are indicated and described below:

<table>
<thead>
<tr>
<th>Hex Address</th>
<th>3FC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>7   6  5  4  3  2  1  0</td>
</tr>
<tr>
<td></td>
<td>Data Terminal Ready (DTR)</td>
</tr>
<tr>
<td></td>
<td>Request to Send (RTS)</td>
</tr>
<tr>
<td></td>
<td>Out 1</td>
</tr>
<tr>
<td></td>
<td>Out 2</td>
</tr>
<tr>
<td></td>
<td>Loop</td>
</tr>
<tr>
<td></td>
<td>= 0</td>
</tr>
<tr>
<td></td>
<td>= 0</td>
</tr>
<tr>
<td></td>
<td>= 0</td>
</tr>
</tbody>
</table>

Modem Control Register (MCR)

Bit 0: This bit controls the data terminal ready (DTR) output. When bit 0 is set to logical 1, the DTR output is forced to a logical 0. When bit 0 is reset to a logical 0, the DTR output is forced to a logical 1.

Note: The DTR output of the INS8250 may be applied to an EIA inverting line driver (such as the DS1488) to obtain the proper polarity input at the succeeding modem or data set.

Bit 1: This bit controls the request to send (RTS) output. Bit 1 affects the RTS output in a manner identical to that described above for bit 0.
Bit 2: This bit controls the output 1 (\( \overline{\text{OUT1}} \)) signal, which is an auxiliary user-designated output. Bit 2 affects the \( \overline{\text{OUT1}} \) output in a manner identical to that described above for bit 0.

Bit 3: This bit controls the output 2 (\( \overline{\text{OUT2}} \)) signal, which is an auxiliary user-designated output. Bit 3 affects the \( \overline{\text{OUT2}} \) output in a manner identical to that described above for bit 0.

Bit 4: This bit provides a loopback feature for diagnostic testing of the INS8250. When bit 4 is set to logical 1, the following occurs: the transmitter serial output (SOUT) is set to the marking (logical 1) state; the receiver serial input (SIN) is disconnected; the output of the transmitter shift register is “looped back” into the receiver shift register input; the four modem control inputs (CTS, DRS, RLSD, and RI) are disconnected; and the four modem control outputs (DTR, RTS, OUT 1, and OUT 2) are internally connected to the four modem control inputs. In the diagnostic mode, data that is transmitted is immediately received. This feature allows the processor to verify the transmit- and receive-data paths of the INS8250.

In the diagnostic mode, the receiver and transmitter interrupts are fully operational. The modem control interrupts are also operational but the interrupts’ sources are now the lower four bits of the modem control register instead of the four modem control inputs. The interrupts are still controlled by the interrupt enable register.

The INS8250 interrupt system can be tested by writing into the lower four bits of the modem status register. Setting any of these bits to a logical 1 generates the appropriate interrupt (if enabled). The resetting of these interrupts is the same as in normal INS8250 operation. To return to normal operation, the registers must be reprogrammed for normal operation and then bit 4 of the modem control register must be reset to logical 0.

Bits 5 through 7: These bits are permanently set to logical 0.
Modem Status Register

This eight-bit register provides the current state of the control lines from the modem (or peripheral device) to the processor. In addition to this current-state information, four bits of the modem status register provide change information. These bits are set to a logical 1 whenever a control input from the modem changes state. They are reset to logical 0 whenever the processor reads the modem status register.

The content of the modem status register are indicated and described below:

<table>
<thead>
<tr>
<th>Hex Address 3FE</th>
<th>Bit 7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Delta Clear to Send (DCTS)</td>
<td>Delta Data Set Ready (DDSR)</td>
<td>Trailing Edge Ring Indicator (TERI)</td>
<td>Delta Rx Line Signal Detect (DRLSD)</td>
<td>Clear to Send (CTS)</td>
<td>Data Set Ready (DSR)</td>
<td>Ring Indicator (RI)</td>
<td>Receive Line Signal Detect (RLSD)</td>
</tr>
</tbody>
</table>

Modem Status Register (MSR)

1-238  Asynchronous Adapter
Bit 0: This bit is the delta clear to send (DCTS) indicator. Bit 0 indicates that the CTS input to the chip has changed state since the last time it was read by the processor.

Bit 1: This bit is the delta data set ready (DDSR) indicator. Bit 1 indicates that the DRS input to the chip has changed since the last time it was read by the processor.

Bit 2: This bit is the trailing edge of ring indicator (TERI) detector. Bit 2 indicates that the RI input to the chip has changed from an on (logical 1) to an off (logical 0) condition.

Bit 3: This bit is the delta received line signal detector (DRLSD) indicator. Bit 3 indicates that the RLSD input to the chip has changed state.

Note: Whenever bit 0, 1, 2, or 3 is set to a logical 1, a modem status interrupt is generated.

Bit 4: This bit is the complement of the clear to send (CTS) input. If bit 4 (LOOP) of the MCR is set to a logical 1, this is equivalent to RTS in the MCR.

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

Bit 6: This bit is the complement of the ring indicator (RI) input. If bit 4 of the MCR is set to a logical 1, this bit is equivalent to OUT 1 in the MCR.

Bit 7: This bit is the complement of the received line signal detect (RLSD) input. If bit 4 of the MCR is set to a logical 1, this bit is equivalent to OUT 2 of the MCR.
**Receiver Buffer Register**

The receiver buffer register contains the received character as defined below:

<table>
<thead>
<tr>
<th>Hex Address</th>
<th>DLAB = 0</th>
<th>Read Only</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>7 6 5 4 3 2 1 0</td>
<td></td>
</tr>
</tbody>
</table>

- Data Bit 0
- Data Bit 1
- Data Bit 2
- Data Bit 3
- Data Bit 4
- Data Bit 5
- Data Bit 6
- Data Bit 7

**Receiver Buffer Register (RBR)**

Bit 0 is the least significant bit and is the first bit serially received.
Transmitter Holding Register

The transmitter holding register contains the character to be serially transmitted and is defined below:

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Hex Address 3F8  DLAB = 0  Write Only

Data Bit 0
Data Bit 1
Data Bit 2
Data Bit 3
Data Bit 4
Data Bit 5
Data Bit 6
Data Bit 7

Transmitter Holding Register (THR)

Bit 0 is the least significant bit and is the first bit serially transmitted.
Selecting the Interface Format and Adapter Address

The voltage or current loop interface and adapter address are selected by plugging the programmed shunt modules with the locator dots up or down. See the figure below for the configurations.
### Hardware

**Rear Panel**

![Rear Panel Diagram]

At Standard RS-232C Levels (With Exception of Current Loops)

<table>
<thead>
<tr>
<th>Description</th>
<th>Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>NC</td>
<td>1</td>
</tr>
<tr>
<td>Transmitted Data</td>
<td>2</td>
</tr>
<tr>
<td>Received Data</td>
<td>3</td>
</tr>
<tr>
<td>Request to Send</td>
<td>4</td>
</tr>
<tr>
<td>Clear to Send</td>
<td>5</td>
</tr>
<tr>
<td>Data Set Ready</td>
<td>6</td>
</tr>
<tr>
<td>Signal Ground</td>
<td>7</td>
</tr>
<tr>
<td>Received Line Signal Detector</td>
<td>8</td>
</tr>
<tr>
<td>+Transmit Current Loop Data</td>
<td>9</td>
</tr>
<tr>
<td>NC</td>
<td>10</td>
</tr>
<tr>
<td>-Transmit Current Loop Data</td>
<td>11</td>
</tr>
<tr>
<td>NC</td>
<td>12</td>
</tr>
<tr>
<td>NC</td>
<td>13</td>
</tr>
<tr>
<td>NC</td>
<td>14</td>
</tr>
<tr>
<td>NC</td>
<td>15</td>
</tr>
<tr>
<td>NC</td>
<td>16</td>
</tr>
<tr>
<td>NC</td>
<td>17</td>
</tr>
<tr>
<td>+Receive Current Loop Data</td>
<td>18</td>
</tr>
<tr>
<td>NC</td>
<td>19</td>
</tr>
<tr>
<td>Data Terminal Ready</td>
<td>20</td>
</tr>
<tr>
<td>NC</td>
<td>21</td>
</tr>
<tr>
<td>Ring Indicator</td>
<td>22</td>
</tr>
<tr>
<td>NC</td>
<td>23</td>
</tr>
<tr>
<td>NC</td>
<td>24</td>
</tr>
<tr>
<td>-Receive Current Loop Return</td>
<td>25</td>
</tr>
</tbody>
</table>

**External Device**

Asynchronous Communications Adapter (RS-232C)

**Note:** To avoid inducing voltage surges on interchange circuits, signals from interchange circuits shall not be used to drive inductive devices, such as relay coils.

**Connector Specifications**
1-244  Asynchronous Adapter
The binary synchronous communications (BSC) adapter is a 4-inch high by 7.5-inch wide card that provides an RS232C-compatible communication interface for the IBM Personal Computer. All system control, voltage, and data signals are provided through a 2- by 31-position card-edge tab. External interface is in the form of EIA drivers and receivers connected to an RS232C, standard 25-pin, D-shell connector.

The adapter is programmed by communication software to operate in binary synchronous mode. Maximum transmission rate is 9600 bits per second (bps). The heart of the adapter is an Intel 8251A Universal Synchronous/Asynchronous Receiver/Transmitter (USART). An Intel 8255A-5 programmable peripheral interface (PPI) is also used for an expanded modem interface, and an Intel 8253-5 programmable interval timer provides time-outs and generates interrupts.

The following is a block diagram of the BSC adapter.
Functional Description

8251A Universal Synchronous/Asynchronous Receiver/Transmitter

The 8251A operational characteristics are programmed by the system unit’s software, and it can support virtually any form of synchronous data technique currently in use. In the configuration being described, the 8251A is used for IBM’s binary synchronous communications (BSC) protocol in half-duplex mode.

Operation of the 8251A is started by programming the communications format, then entering commands to tell the 8251A what operation is to be performed. In addition, the 8251A can pass device status to the system unit by doing a Status Read operation. The sequence of events to accomplish this are mode instruction, command instruction, and status read. Mode instruction must follow a master reset operation. Commands can be issued in the data block at any time during operation of the 8251A.

A block diagram of the 8251A follows:

8251A Block Diagram

1-246  BSC Adapter
Data Bus Buffer

The system unit's data bus interfaces the 8251A through the data bus buffer. Data is transferred or received by the buffer upon execution of input or output instructions from the system unit. Control words, command words, and status information are also transferred through the data bus buffer.

Read/Write Control Logic

The read/write control logic controls the transfer of information between the system unit and the 8251A. It consists of pins designated as RESET, CLK, WR, RD, C/D, and CS.

RESET: The Reset pin is gated by Port B, bit 4 of the 8255, and performs a master reset of the 8251A. The minimum reset pulse width is 6 clock cycles. Clock-cycle duration is determined by the oscillator speed of the processor.

CLK (Clock): The clock generates internal device timing. No external inputs or outputs are referenced to CLK. The input is the system board's bus clock of 4.77 MHz.

WR (Write): An input to WR informs the 8251A that the system unit is writing data or control words to it. The input is the WR signal from the system-unit bus.

RD (Read): An input to RD informs the 8251A that the processing unit is reading data or status information from it. The input is the RD signal from the system-unit bus.

C/D (Control/Data): An input on this pin, in conjunction with the WR and RD inputs, informs the 8251A that the word on the data bus is either a data character, a control word, or status information. The input is the low-order address bit from the system board's address bus.

CS (Chip Select): A low on the input selects the 8251A. No reading or writing will occur unless the device is selected. An input is decoded at the adapter from the address information on the system-unit bus.
Modem Control

The **8251A** has the following input and output control signals which are used to interface the transmission equipment selected by the user.

**DSR (Data Set Ready):** The DSR input port is a general-purpose, 1-bit, inverting input port. The **8251A** can test its condition with a Status Read operation.

**CTS (Clear to Send):** A low on this input enables the **8251A** to transfer serial data if the TxEnable bit in the command byte is set to 1. If either a TxEnable off or CTS off condition occurs while the transmitter is in operation, the transmitter will send all the data in the **USART** that was written prior to the **TxDisable** command, before shutting down.

**DTR (Data Terminal Ready):** The DTR output port is a general-purpose, 1-bit, inverting output port. It can be set low by programming the appropriate bit in the command instruction word.

**RTS (Request to Send):** The RTS output signal is a general-purpose, 1-bit, inverting output port. It can be set low by programming the appropriate bit in the Command Instruction word.

Transmitter Buffer

The transmitter buffer accepts parallel data from the data-bus buffer, converts it to a serial bit stream, and inserts the appropriate characters or bits for the BSC protocol. The output from the transmit buffer is a composite serial stream of data on the falling edge of Transmit Clock. The transmitter will begin transferring data upon being enabled, if CTS = 0 (active). The transmit data (**TxD**) line will be set in the marking state upon receipt of a master reset, or when transmit **enable/CTS** is off and the transmitter is empty (**TxEmpty**).
Transmitter Control

Transmitter Control manages all activities associated with the transfer of serial data. It accepts and issues the following signals, both externally and internally, to accomplish this function:

TxRDY (Transmitter Ready): This output signals the system unit that the transmitter is ready to accept a data character. The TxRDY output pin is used as an interrupt to the system unit (Level 4) and is masked by turning off Transmit Enable. TxRDY is automatically reset by the leading edge of a WR input signal when a data character is loaded from the system unit.

TxE (Transmitter Empty): This signal is used only as a status register input.

TxC (Transmit Clock): The Transmit Clock controls the rate at which the character is to be transmitted. In synchronous mode, the bit-per-second rate is equal to the TxC frequency. The falling edge of TxC shifts the serial data out of the 8251A.

Receiver Buffer

The receiver accepts serial data, converts it to parallel format, checks for bits or characters that are unique to the communication technique, and sends an "assembled" character to the system unit. Serial data input is received on the RxD (Receive Data) pin, and is clocked in on the rising edge of RxC (Receive Clock).

Receiver Control

This control manages all receiver-related activities. The parity-toggle and parity-error flip-flop circuits are used for parity-error detection, and set the corresponding status bit.
RxRDY (Receiver Ready): This output indicates that the 8251A has a character that is ready to be received by the system unit. RxRDY is connected to the interrupt structure of the system unit (Interrupt Level 3). With Receive Enable off, RxRDY is masked and held in the reset mode. To set RxRDY, the receiver must be enabled, and a character must finish assembly and be transferred to the data output register. Failure to read the received character from the RxRDY output register before the assembly of the next Rx Data character will set an overrun-condition error, and the previous character will be lost.

RxC (Receiver Clock): The receiver clock controls the rate at which the character is to be received. The bit rate is equal to the actual frequency of RxC.

SYNDET (Synchronization Detect): This pin is used for synchronization detection and may be used as either input or output, programmable through the control word. It is reset to output-mode-low upon reset. When used as an output (internal synchronization mode), the SYNDET pin will go to 1 to indicate that the 8251A has found the synchronization character in the receive mode. If the 8251A is programmed to use double synchronization characters (bisynchronization, as in this application), the SYNDET pin will go to 1 in the middle of the last bit of the second synchronization character. SYNDET is automatically reset for a Status Read operation.

8255A-5 Programmable Peripheral Interface

The 8255A-5 is used on the BSC adapter to provide an expanded modem interface and for internal gating and control functions. It has three 8-bit ports, which are defined by the system during initialization of the adapter. All levels are considered plus active unless otherwise indicated. A detailed description of the ports is in "Programming Considerations" in this section.

I-250 BSC Adapter
8253-5 Programmable Interval Timer

The 8253-5 is driven by a divided-by-two system-clock signal. Its outputs are used as clocking signals and to generate inactivity timeout interrupts. These level 4 interrupts occur when either of the timers reaches its programmed terminal counts. The 8253-5 has the following outputs:

Timer 0: Not used for synchronous-mode operation.

Timer 1: Connected to port A, bit 7 of the 8255 and Interrupt Level 4.

Timer 2: Connected to port A, bit 6 of the 8255 and Interrupt Level 4.

Operation

The complete functional definition of the BSC adapter is programmed by the system software. Initialization and control words are sent out by the system to initialize the adapter and program the communications format in which it operates. Once programmed, the BSC Adapter is ready to perform its communication functions.

Transmit

In synchronous transmission, the TxD output is continuously at a mark level until the system sends its first character, which is a synchronization character to the 8251A. When the CTS line goes on, the first character is serially transmitted. All bits are shifted out on the falling edge of TxC. When the 8251A is ready to receive another character from the system for transmission, it raises TxDY, which causes a level-4 interrupt.
Once transmission has started, the data stream at the TxD output must continue at the TxC rate. If the system does not provide the 8251A with a data character before the 8251A transmit buffers become empty, the synchronization characters will be automatically inserted in the TxD data stream. In this case, the TxE bit in the status register is raised high to signal that the 8251A is empty and that synchronization characters are being sent out. (Note that this TxE bit is in the status register, and is not the TxE pin on the 8251A). TxE does not go low when SYNC is being shifted out. The TxE status bit is internally reset by a data character being written to the 8251A.

Receive

In synchronous reception, the 8251A will achieve character synchronization, because the hardware design of the BSC adapter is intended for internal synchronization. Therefore, the SYNDET pin on the 8251A is not connected to the adapter circuits. For internal synchronization, the Enter Hunt command should be included in the first command instruction word written. Data on the RxD pin is then sampled in on the rising edge of RxC. The content of the RxD buffer is compared at every bit boundary with the first SYNC character until a match occurs. Because the 8251A has been programmed for two synchronization characters (bisynchronization), the next received character is also compared. When both SYNC characters have been detected, the 8251A ends the hunt mode and is in character synchronization. The SYNDET bit in the status register (not the SYNDET pin) is then set high, and is reset automatically by a Status Read.

Once synchronization has occurred, the 8251A begins to assemble received data bytes. When a character is assembled and ready to be transferred to memory from the 8251A, it raises RxRDY, causing an interrupt level 3 to the system.

If the system has not fetched a previous character by the time another received character is assembled (and an interrupt-level 3 issued by the adapter), the old character will be overwritten, and the overrun error flag will be raised. All error flags can be reset by an error reset operation.
Programming Considerations

Before starting data transmission or reception, the BSC adapter is programmed by the system unit to define control and gating ports, timer functions and counts, and the communication environment in which it is to operate.

Typical Programming Sequence

The 8255A-5 programmable peripheral interface (PPI) is initialized for the proper mode by selecting address hex 3A3 and writing the control word. This defines port A as an input, port B as an output for modem control and gating, and port C for 4-bit input and 4-bit output. The bit descriptions for the 8255A-5 are shown in the following figures. Using an output to port C, the adapter is then set to wrap mode, disallow interrupts, and gate external clocks (address=3A2H, data=ODH). The adapter is now isolated from the communication interface, and initialization continues.

Through bit 4 of 8255 Port B, the 8251A reset pin is brought high, held, then dropped. This resets the internal registers of the 8251A.
### 8255 Port A Assignments
**Input Port**

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscillating = Transmit Clock Active</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscillating = Receive Clock Active</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 = TxRDY Active</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 = Timer 2 Output Active</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 = Timer 1 Output Active</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Address**: hex 3A0 for BSC  
hex 380 for Alternate BSC

### 8255 Port B Assignments
**Output Port**

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 = Not Used</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 = Reset 8251A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 = Gate Timer 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 = Gate Timer 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 = Gate Timers 1 and 2 to Interrupt Level 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Address**: hex 3A1 for BSC  
hex 381 for Alternate BSC

### 8255 Port C Assignments

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 = Gate Internal Clock (Output Bit)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 = Gate External Clock (Output Bit)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 = Electronic Wrap (Output Bit)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 = Enable Timer 1 and 2, Interrupt 6 and Receive Interrupt 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscillating = Receive Data (Input Bit)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscillating = Timer 0 Output (Input Bit)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 = Test Indicate Active (Input Bit)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 = BSC Adapter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Address**: hex 3A2 for BSC  
hex 382 for Alternate BSC

The 8253-5 programmable interval timer is used in the synchronous mode to provide inactivity time-outs to interrupt the system unit after a preselected period of time has elapsed from the start of a communication operation. Counter 0 is not used for synchronous operation. Counters 1 and 2 are connected to interrupt-level 4, and are programmed to terminal-count values, which will provide the desired time delay before a level-4 interrupt is generated. These interrupts will indicate to the system software that a predetermined period of time has elapsed without a TxRDY (level 4) or RxRDY (level 3) interrupt being sent to the system unit.

1-254  BSC Adapter
The modes for each counter are programmed by selecting each timer-register address and writing the correct control word for counter operation to the adapter. The mode for counters 1 and 2 is set to 0. The terminal-count values are loaded using control-word bits D4 and D5 to select “load.” The 8253-5 Control Word format is shown in the following chart.

### Control Word Format

<table>
<thead>
<tr>
<th>Control Word Format</th>
<th>Address hex 3A7</th>
</tr>
</thead>
<tbody>
<tr>
<td>D7  D6  D5  D4  D3  D2  D1  D0</td>
<td>SC1  SC0  RL1  RL0  M2  M1  M0  BCD</td>
</tr>
</tbody>
</table>

### Definition of Control

#### SC – Select Counter:

<table>
<thead>
<tr>
<th>SC1</th>
<th>SC0</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Select Counter 0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Select Counter 1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Select Counter 2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Illegal</td>
</tr>
</tbody>
</table>

#### RL – Read/Load:

<table>
<thead>
<tr>
<th>RL1</th>
<th>RL0</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Counter Latching operation</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Read/Load most significant byte only</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Read/Load least significant byte only</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Read/Load least significant byte first, then most significant byte</td>
</tr>
</tbody>
</table>

#### M – Mode:

<table>
<thead>
<tr>
<th>M2</th>
<th>M1</th>
<th>M0</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Mode 0</td>
</tr>
</tbody>
</table>

#### BCD:

<p>| | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Binary Counter 16-bits</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Binary Coded Decimal (BCD) Counter (4 Decades)</td>
</tr>
</tbody>
</table>

---

8253-5 Control Word Format
8251A Programming Procedures

After the support devices on the BSC adapter are programmed, the 8251A is loaded with a set of control words that define the communication environment. The control words are split into two formats, mode instruction, and command instruction.

Both the mode and command instructions must conform to a specified sequence for proper device operation. The mode instruction must be inserted immediately after a reset operation, before using the 8251A for data communications. The required synchronization characters for the defined communication technique are next loaded into the 8251A (usually hex 32 for BSC). All control words written to the 8251A after the mode instruction will load the command instruction. Command instructions can be written to the 8251A at any time in the data block anytime during the operation of the 8251A. To return to the mode instruction format, the master reset bit in the command instruction word can be set to start an internal reset operation which automatically places the 8251A back into the mode instruction format. Command instructions must follow the mode instructions or synchronization characters.

The following diagram is a typical data block, showing the mode instruction and command instruction.

![Diagram of typical data block]

Typical Data Block

1-256 BSC Adapter
Mode Instruction Definition

The mode instruction defines the general operational characteristics of the 8251A. It follows a reset operation (internal or external). Once the mode instruction has been written to the 8251A by the system unit, synchronization characters or command instructions may be written to the device.

The following figure shows the format for the mode instruction.

| Mode Instruction Format | Address: Hex 3A9 for BSC
|                        | Hex 389 for Alternate BSC

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Not Used (Always 0)</td>
<td>00</td>
</tr>
<tr>
<td>6</td>
<td>Not Used (Always 0)</td>
<td>00</td>
</tr>
<tr>
<td>5</td>
<td>Character Length Bit</td>
<td>5 Bits</td>
</tr>
<tr>
<td>4</td>
<td>Character Length Bit</td>
<td>6 Bits</td>
</tr>
<tr>
<td>3</td>
<td>1 = Parity Enable</td>
<td>7 Bits</td>
</tr>
<tr>
<td>2</td>
<td>0 = Double SYNC Character</td>
<td>8 Bits</td>
</tr>
<tr>
<td>1</td>
<td>1 = Even Parity</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1 = SYNDET is an Input</td>
<td></td>
</tr>
</tbody>
</table>

Bit 0 Not used; always 0

Bit 1 Not used; always 0

Bit 2 and Bit 3 These two bits are used together to define the character length. With 0 and 1 as inputs on bits 2 and 3, character lengths of 5, 6, 7, and 8 bits can be established, as shown in the preceding figure.

Bit 4 In the synchronous mode, parity is enabled from this bit. A 1 on this bit sets parity enable.

Bit 5 The parity generation/check is set from this bit. For BSC, even parity is used by having bit 5 = 1.

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

Bit 7 This bit establishes the mode of character synchronization. A 0 is set on this bit to give double character synchronization.
Command-Instruction Format

The command-instruction format defines a status word that is used to control the actual operation of the 8251A. Once the mode instruction has been written to the 8251A, and SYNC characters loaded, all further “Control Writes” to I/O address hex 3A9 or hex 389 will load a command instruction.

Data is transferred by accessing two I/O ports on the 8251A, ports 3A8 and 388. A byte of data can be read from port 3A8 and can be written to port 388.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Transmit Enable</td>
</tr>
<tr>
<td>6</td>
<td>Data Terminal Ready</td>
</tr>
<tr>
<td>5</td>
<td>Receive Enable</td>
</tr>
<tr>
<td>4</td>
<td>Send Break Character</td>
</tr>
<tr>
<td>3</td>
<td>Error Reset</td>
</tr>
<tr>
<td>2</td>
<td>Request to Send</td>
</tr>
<tr>
<td>1</td>
<td>Internal Reset</td>
</tr>
<tr>
<td>0</td>
<td>Enter Hunt Mode</td>
</tr>
</tbody>
</table>

Command Instruction Format

Bit 0  The Transmit Enable bit sets the function of the 8251A to either enabled (1) or disabled (0).

Bit 1  The Data Terminal Ready bit, when set to 1 will force the data terminal output to 0. This is a one-bit inverting output port.

Bit 2  The Receive Enable bit sets the function to either enable the bit (1), or to disable the bit (0).

Bit 3  The Send Break Character bit is set to 0 for normal BSC operation.

Bit 4  The Error Reset bit is set to 1 to reset error flags from the command instruction.

Bit 5  A 1 on the Request to Send bit will set the output to 0. This is a one-bit inverting output port.

1-258  BSC Adapter
Bit 6 The Internal Reset bit when set to 1 returns the 8251A to mode-instruction format.

Bit 7 The Enter Hunt bit is set to 1 for BSC to enable a search for synchronization characters.

Status Read Definition

In telecommunication systems, the status of the active device must often be checked to determine if errors or other conditions have occurred that require the processor’s attention. The 8251A has a status read facility that allows the system software to read the status of the device at anytime during the functional operation. A normal read command is issued by the processor with I/O address hex 3A9 for BSC, and hex 389 for Alternate BSC to perform a status read operation.

The format for a status read word is shown in the figure below. Some of the bits in the status read format have the same meanings as external output pins so the 8251A can be used in a completely polled environment or in an interrupt-driven environment.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>TxDY (See Note Below)</td>
</tr>
<tr>
<td>1</td>
<td>RXRDY</td>
</tr>
<tr>
<td>2</td>
<td>TXEmpty</td>
</tr>
<tr>
<td>3</td>
<td>Parity Error (PE Flag On when a Parity Error Occurs)</td>
</tr>
<tr>
<td>4</td>
<td>Overrun Error (OE Flag On when Overrun Error Occurs)</td>
</tr>
<tr>
<td>5</td>
<td>Framing Error (Not Used for Synchronous Communications)</td>
</tr>
<tr>
<td>6</td>
<td>SYNDET</td>
</tr>
<tr>
<td>7</td>
<td>Data Set Ready (Indicates that DSR is at 0 Level)</td>
</tr>
</tbody>
</table>

Note: TxDY status bit does not have the same meaning as the 8251A TxDY output pin. The former is not conditioned by CTS and TxEnable. The latter is conditioned by both CTS and TxEnable.

Status Read Format
Bit 0   See the Note in the preceding figure.

Bit 1   An output on this bit means a character is ready to be received by the computer’s 8088 microprocessor.

Bit 2   A 1 on this bit indicates the 8251A has no characters to transmit.

Bit 3   The Parity Error bit sets a flag when errors are detected. It is reset by the error reset in the command instruction.

Bit 4   This bit sets a flag when the computers 8088 microprocessor does not read a character before another one is presented. The 8251A operation is not inhibited by this flag, but the overrun character will be lost.

Bit 5   Not used

Bit 6   SYNDET goes to 1 when the synchronization character is found in receive mode. For BSC, SYNDET goes high in the middle of the last bit of the second synchronization character.

Bit 7   The Data Set Ready bit is a one bit inverting input. It is used to check modem conditions, such as data-set ready.

**Interface Signal Information**

The BSC adapter conforms to interface signal levels standardized by the Electronics Industry Association (EIA) RS232C Standard. These levels are shown in the following figure.

Additional lines, not standardized by the EIA, are pins 11, 18, and 25 on the interface connector. These lines are designated as Select Standby, Test, and Test Indicate. Select Standby is used to support the switched network backup facility of a modem that provides this option. Test and Test Indicate support a modem wrap function on modems that are designated for business-machine, controlled-modem wraps.
### Interface Voltage Levels

#### EIA RS232C/CCITT V24-V28 Signal Levels

<table>
<thead>
<tr>
<th>Driver</th>
<th>EIA RS232C/CCITT V24-V28 Signal Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>+15 Vdc</td>
<td>Active/Data = 0</td>
</tr>
<tr>
<td>+5 Vdc</td>
<td></td>
</tr>
<tr>
<td>+5 Vdc</td>
<td></td>
</tr>
<tr>
<td>-5 Vdc</td>
<td>Invalid Level</td>
</tr>
<tr>
<td>-5 Vdc</td>
<td></td>
</tr>
<tr>
<td>-15 Vdc</td>
<td>Inactive/Data = 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Receiver</th>
<th>EIA RS232C/CCITT V24-V28 Signal Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>+25 Vdc</td>
<td>Active/Data = 0</td>
</tr>
<tr>
<td>+3 Vdc</td>
<td></td>
</tr>
<tr>
<td>+3 Vdc</td>
<td></td>
</tr>
<tr>
<td>-3 Vdc</td>
<td>Invalid Level</td>
</tr>
<tr>
<td>-3 Vdc</td>
<td></td>
</tr>
<tr>
<td>-25 Vdc</td>
<td>Inactive/Data = 1</td>
</tr>
</tbody>
</table>
Interrupt Information

Interrupt Level 4: Transmitter Ready
Counter 1
Counter 2

Interrupt Level 3: Receiver Ready

The following chart is a device address summary for the primary and alternate modes of the binary synchronous communications adapter.

<table>
<thead>
<tr>
<th>Hex Address</th>
<th>Device</th>
<th>Register Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary</td>
<td>Alternate</td>
<td>8255</td>
<td>Port A Data</td>
</tr>
<tr>
<td>3A0</td>
<td>380</td>
<td>8255</td>
<td>Port B Data</td>
</tr>
<tr>
<td>3A1</td>
<td>381</td>
<td>8255</td>
<td>Port C Data</td>
</tr>
<tr>
<td>3A2</td>
<td>382</td>
<td>8255</td>
<td>Mode Set</td>
</tr>
<tr>
<td>3A3</td>
<td>383</td>
<td>8253</td>
<td>Counter 0 LSB</td>
</tr>
<tr>
<td>3A4</td>
<td>384</td>
<td>8253</td>
<td>Counter 0 MSB</td>
</tr>
<tr>
<td>3A5</td>
<td>385</td>
<td>8253</td>
<td>Counter 1 LSB</td>
</tr>
<tr>
<td>3A6</td>
<td>386</td>
<td>8253</td>
<td>Counter 1 MSB</td>
</tr>
<tr>
<td>3A7</td>
<td>387</td>
<td>8253</td>
<td>Counter 2 LSB</td>
</tr>
<tr>
<td>3A8</td>
<td>388</td>
<td>8251</td>
<td>Counter 2 MSB</td>
</tr>
<tr>
<td>3A9</td>
<td>389</td>
<td>8251</td>
<td>Mode Register</td>
</tr>
</tbody>
</table>

Device Address Summary

Data
Mode/Command
USART Status

1-262  BSC Adapter
## External Device

<table>
<thead>
<tr>
<th>Signal Name — Description</th>
<th>Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Connection</td>
<td>1</td>
</tr>
<tr>
<td>Transmitted Data</td>
<td>2</td>
</tr>
<tr>
<td>Received Data</td>
<td>3</td>
</tr>
<tr>
<td>Request to Send</td>
<td>4</td>
</tr>
<tr>
<td>Clear to Send</td>
<td>5</td>
</tr>
<tr>
<td>Data Set Ready</td>
<td>6</td>
</tr>
<tr>
<td>Signal Ground</td>
<td>7</td>
</tr>
<tr>
<td>Received Line Signal Detector</td>
<td>8</td>
</tr>
<tr>
<td>No Connection</td>
<td>9</td>
</tr>
<tr>
<td>No Connection</td>
<td>10</td>
</tr>
<tr>
<td>Select Standby*</td>
<td>11</td>
</tr>
<tr>
<td>No Connection</td>
<td>12</td>
</tr>
<tr>
<td>No Connection</td>
<td>13</td>
</tr>
<tr>
<td>No Connection</td>
<td>14</td>
</tr>
<tr>
<td>Transmitter Signal Element Timing</td>
<td>15</td>
</tr>
<tr>
<td>No Connection</td>
<td>16</td>
</tr>
<tr>
<td>Receiver Signal Element Timing</td>
<td>17</td>
</tr>
<tr>
<td>Test (IBM Modems Only)*</td>
<td>18</td>
</tr>
<tr>
<td>No Connection</td>
<td>19</td>
</tr>
<tr>
<td>Data Terminal Ready</td>
<td>20</td>
</tr>
<tr>
<td>No Connection</td>
<td>21</td>
</tr>
<tr>
<td>Ring Indicator</td>
<td>22</td>
</tr>
<tr>
<td>Data Signal Rate Selector</td>
<td>23</td>
</tr>
<tr>
<td>No Connection</td>
<td>24</td>
</tr>
<tr>
<td>Test Indicate (IBM Modems Only)*</td>
<td>25</td>
</tr>
</tbody>
</table>

*Not standardized by EIA (Electronics Industry Association).

### Connector Specifications

**BSC Adapter**  1-263
Notes:

1-264  BSC Adapter
IBM Synchronous Data Link Control (SDLC) Communications Adapter

The SDLC communications adapter system control, voltage, and data signals are provided through a 2 by 31 position card edge tab. Modem interface is in the form of EIA drivers and receivers connecting to an RS232C standard 25-pin, D-shell, male connector.

The adapter is programmed by communications software to operate in a half-duplex synchronous mode. Maximum transmission rate is 9600 bits per second, as generated by the attached modem or other data communication equipment.

The SDLC adapter utilizes an Intel 8273 SDLC protocol controller and an Intel 8255A-5 programmable peripheral interface for an expanded external modem interface. An Intel 8253 programmable interval timer is also provided to generate timing and interrupt signals. Internal test loop capability is provided for diagnostic purposes.

The figure below is a block diagram of the SDLC communications adapter.
The **8273** SDLC protocol control module has the following key features:

- Automatic frame check sequence generation and checking.
- Automatic zero bit insertion and deletion.
- TTL compatibility.
- Dual internal processor architecture, allowing frame level command structure and control of data channel with minimal system processor intervention.

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

**Command**

Commands **and/or** parameters for the required operation are issued by the processor.

**Execution**

Executes the command, manages the data link, and may transfer data to or from memory utilizing direct memory access (DMA), thus freezing the processor except for minimal interruptions.

**Result**

Returns the outcome of the command by returning interrupt results.

Support of the controller operational phases is through internal registers and control blocks of the **8273** controller.
The 8273 module consists of two major interfaces: the processor interface and the modem interface. A block diagram of the 8273 protocol controller module follows.

8273 SDLC Protocol Control Block Diagram
Processor Interface

The processor interface consists of four major blocks: the control/read/write logic (C/R/W), internal registers, data transfer logic, and data bus buffers.

Control/Read/Write Logic

The control/read/write logic is used by the processor to issue commands to the 8273. Once the 8273 receives and executes a command, it returns the results using the C/R/W logic. The logic is supported by seven registers which are addressed by A0, A1, RD, and WR, in addition to CS. A0 and A1 are the two low-order bits of the adapter address-byte. RD and WR are the processor read and write signals present on the system control bus. CS is the chip select, also decoded by the adapter address logic. The table below shows the address of each register using the C/R/W logic.

<table>
<thead>
<tr>
<th>Address Inputs</th>
<th>Control Inputs</th>
<th>Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>A0 A1</td>
<td>CS WR RD</td>
<td></td>
</tr>
<tr>
<td>0 0</td>
<td>0 0 1</td>
<td>Command</td>
</tr>
<tr>
<td>0 0</td>
<td>0 1 0</td>
<td>Status</td>
</tr>
<tr>
<td>0 1</td>
<td>0 0 1</td>
<td>Parameter</td>
</tr>
<tr>
<td>0 1</td>
<td>0 1 0</td>
<td>Result</td>
</tr>
<tr>
<td>1 0</td>
<td>0 0 1</td>
<td>Reset</td>
</tr>
<tr>
<td>1 0</td>
<td>0 1 0</td>
<td>Txl/R</td>
</tr>
<tr>
<td>1 1</td>
<td>0 0 1</td>
<td>None</td>
</tr>
<tr>
<td>1 1</td>
<td>0 1 0</td>
<td>Rxl/R</td>
</tr>
</tbody>
</table>

8273 SDLC Protocol Controller Register Selection
### 8273 Control/Read/Write Registers

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>Operations are initialized by writing the appropriate command byte into this register.</td>
</tr>
<tr>
<td>Status</td>
<td>This register provides the general status of the 8273. The status register supplies the processor/adapter handshaking necessary during various phases of the 8273 operation.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Additional information that is required to process the command is written into this register. Some commands require more than one parameter.</td>
</tr>
<tr>
<td>Immediate Result (Result)</td>
<td>Commands that execute immediately produce a result byte in this register, to be read by the processor.</td>
</tr>
<tr>
<td>Transmit Interrupt Results (TxI/R)</td>
<td>Results of transmit operations are passed to the processor from this register. This result generates an interrupt to the processor when the result becomes available.</td>
</tr>
<tr>
<td>Receiver Interrupt Results (Rx/I/R)</td>
<td>Results of receive operations are passed to the processor from this register. This result generates an interrupt to the processor when the result becomes available.</td>
</tr>
<tr>
<td>Reset</td>
<td>This register provides a software reset function for the 8273.</td>
</tr>
</tbody>
</table>

The other elements of the C/R/W logic are the interrupt lines (RxINT and TxINT). Interrupt priorities are listed in the "Interrupt Information" table in this section. These lines signal the processor that either the transmitter or the receiver requires service (results should be read from the appropriate register), or a data transfer is required. The status of each interrupt line is also reflected by a bit in the status register, so non-interrupt driven operation is also possible by the communication software examining these bits periodically.
Data Interfaces

The 8273 supports two independent data interfaces through the data transfer logic: received data and transmitted data. These interfaces are programmable for either DMA or non-DMA data transfers. Speeds below 9600 bits-per-second may or may not require DMA, depending on the task load and interrupt response time of the processor. The processor DMA controller is used for management of DMA data transfer timing and addressing. The 8273 handles the transfer requests and actual counts of data-block lengths. DMA level 1 is used to transmit and receive data transfers. Dual DMA support is not provided.

Elements of Data Transfer Interface

**TxDRQ/RxDRQ**  
This line requests a DMA to or from memory and is asserted by the 8273.

**TxDACK/RxDACK**  
This line notifies the 8273 that a request has been granted and provides access to data regions. This line is returned by the DMA controller (DACK1 on the system unit control bus is connected to TxDACK/RxDACK on the 8273).

**RD (Read)**  
This line indicates data is to be read from the 8273 and placed in memory. It is controlled by the processor DMA controller.

**WR (Write)**  
This line indicates if data is to be written to the 8273 from memory and is controlled by the processor DMA controller.

To request a DMA transfer, the 8273 raises the DMA request line. Once the DMA controller obtains control of the system bus, it notifies the 8273 that the DRQ is granted by returning DACK, and WR or RD, for a transmit or receive operation, respectively. The DACK and WR or RD signals transfer data between the 8273 and memory, independent of the 8273 chip-select pin (CS). This "hard select" of data into the transmitter or out of the receiver alleviates the need for the normal transmit and receive data registers, addressed by a combination of address lines, CS, and WR or RD.

1-270 SDLC Adapter
Modem Interface

The modem interface of the 8273 consists of two major blocks: the modem control block and the serial data timing block.

Modem Control Block

The modem control block provides both dedicated and user-defined modem control function. EIA inverting drivers and receivers are used to convert TTL levels to EIA levels.

Port A is a modem control input port. Bits PA0 and PA1 have dedicated functions.

<table>
<thead>
<tr>
<th>8273 Port A (Modem Control Input Port)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit PA</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>PA0 Clear to Send</td>
</tr>
<tr>
<td>PA1 Carrier Detect</td>
</tr>
<tr>
<td>PA2 Data Set Ready</td>
</tr>
<tr>
<td>PA3 CTS Change</td>
</tr>
<tr>
<td>PA4 DSR Change</td>
</tr>
<tr>
<td>Not Used</td>
</tr>
</tbody>
</table>

Bit PA0

This bit reflects the logical state of the clear to send (CTS) pin. The 8273 waits until CTS is active before it starts transmitting a frame. If CTS goes inactive while transmitting, the frame is aborted and the processor is interrupted. A CTS failure will be indicated in the appropriate interrupt-result register.

Bit PA1

This bit reflects the logical state of the carrier detect pin (CD). CD must be active in sufficient time for reception of a frame’s address field. If CD is lost (goes inactive) while receiving a frame, an interrupt is generated with a CD failure result.

Bit PA2

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

Bit PA3

This bit is a sense bit to detect a change in CTS.
Bit PA4  This bit is a sense bit to detect a change in data set ready.

Bits PA5 to PA7  These bits are not used and each is read as a 1 for a read port A command.

Port B is a modem control output port. Bits PB0 and PB5 are dedicated function pins.

<table>
<thead>
<tr>
<th>8273 Port B (Modem Control Output Port)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit PB 7 6 5 4 3 2 1 0</td>
</tr>
<tr>
<td>PB0 - Request to Send</td>
</tr>
<tr>
<td>PB1 - Reserved</td>
</tr>
<tr>
<td>PB2 - Data Terminal Ready</td>
</tr>
<tr>
<td>PB3 - Reserved</td>
</tr>
<tr>
<td>PB4 - Reserved</td>
</tr>
<tr>
<td>PB5 - Flag Detect</td>
</tr>
<tr>
<td>PB6 - Not Used</td>
</tr>
<tr>
<td>PB7 - Not Used</td>
</tr>
</tbody>
</table>

Bit PB0  This bit represents the logical state of request to send (RTS). This function is handled automatically by the 8273.

Bit PB1  Reserved.

Bit PB2  Used for data terminal ready.

Bit PB3  Reserved.

Bit PB4  Reserved.

Bit PB5  This bit reflects the state of the flag detect pin. This pin is activated whenever an active receiver sees a flag character.

Bit PB6  Not used.

Bit PB7  Not used.

I-272  SDLC Adapter
Serial Data Timing Block

The serial data timing block is comprised of two sections: the serial data logic and the digital phase locked loop (DPLL).

Elements of the serial data logic section are the data pins TxD (transmitted data output) and RxD (received data input), and the respective clocks. The leading edge of TxC generates new transmitted data and the trailing edge of RxC is used to capture the received data. The figure below shows the timing for these signals.

![Diagram of serial data timing block]

8273 SDLC Protocol Controller Transmit/Receive Timing

The digital phase locked loop provided on the 8273 controller module is utilized to capture looped data in proper synchronization during wrap operations performed by diagnostics.
8255A-5 Programmable Peripheral Interface

The 8255A-5 contains three eight bit ports. Descriptions of each bit of these ports are as follows:

<table>
<thead>
<tr>
<th>8255A-5 Port A Assignments*</th>
<th>Hex Address 380</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td></td>
</tr>
<tr>
<td>7  6  5  4  3  2  1  0</td>
<td></td>
</tr>
<tr>
<td>0 = Ring Indicator is on from Interface</td>
<td></td>
</tr>
<tr>
<td>0 = Data Carrier Detect is on from Interface</td>
<td></td>
</tr>
<tr>
<td>Oscillating = Transmit Clock Active</td>
<td></td>
</tr>
<tr>
<td>0 = Clear to Send is on from Interface</td>
<td></td>
</tr>
<tr>
<td>Oscillating = Receive Clock Active</td>
<td></td>
</tr>
<tr>
<td>1 = Modem Status Changed</td>
<td></td>
</tr>
<tr>
<td>1 = Timer 2 Output Active</td>
<td></td>
</tr>
<tr>
<td>1 = Timer 1 Output Active</td>
<td></td>
</tr>
</tbody>
</table>

*Port A is defined as an input port

<table>
<thead>
<tr>
<th>8255A-5 Port B Assignments*</th>
<th>Hex Address 381</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td></td>
</tr>
<tr>
<td>7  6  5  4  3  2  1  0</td>
<td></td>
</tr>
<tr>
<td>0 = Turn On Data Signal Rate Select at Modem Interface</td>
<td></td>
</tr>
<tr>
<td>0 = Turn On Select Standby at Modem Interface</td>
<td></td>
</tr>
<tr>
<td>0 = Turn On Test</td>
<td></td>
</tr>
<tr>
<td>1 = Reset Modem Status Changed Logic</td>
<td></td>
</tr>
<tr>
<td>1 = Reset 8273</td>
<td></td>
</tr>
<tr>
<td>1 = Gate Timer 2</td>
<td></td>
</tr>
<tr>
<td>1 = Gate Timer 1</td>
<td></td>
</tr>
<tr>
<td>1 = Enable Level 4 Interrupt</td>
<td></td>
</tr>
</tbody>
</table>

*Port B is defined as an output port
8255A-5 Port C Assignments

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>= Gate Internal Clock (Output Bit)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>= Gate External Clock (Output Bit)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>= Electronic Wrap (Output Bit)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>= Gate Interrupts 3 and 4 (Output Bit)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscillating</td>
<td>= Receive Data (Input Bit)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Oscillating</td>
<td>= Timer 0 Output (Input bit)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>= Test Indicate Active (Input Bit)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Port C is defined for internal control and gating functions. It has three input and four output bits. The four output bits are defined during initialization, but only three are used.

8253-5 Programmable Interval Timer

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

- **Timer 0**: Programmed to generate a square wave signal, used as an input to timer 2. Also connected to 8253 port C, bit 5.
- **Timer 1**: Connected to 8255 port A, bit 7, and interrupt level 4.
- **Timer 2**: Connected to 8255 port A, bit 6, and interrupt level 4.

Programming Considerations

The software aspects of the 8273 involve the communication of both commands from the processor to the 8273 and the return of results of those commands from the 8273 to the processor. Due to the internal processor architecture of the 8273, this system unit/8273 communication is basically a form of interprocessor communication, and must be considered when programming for the SDLC communications adapter.
The protocol for this interprocessor communication is implemented through use of handshaking supplied in the 8273 status register. The bit definitions of this register are shown below.

<table>
<thead>
<tr>
<th>Bit</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RxIRA 1 = RxINT Result Available</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TxIRA 1 = TxINT Result Available</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TxINT 1 = Tx Interrupt</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>RxINT 1 = Rx Interrupt</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CRBF 1 = Command Result Buffer Full</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CPBF 1 = Command Parameter Buffer Full</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CBF 1 = Command Buffer Full</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CBSY 1 = Command Busy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Bit 0  This bit is the transmitter interrupt result available (TxIRA) bit. This bit is set when the 8273 places an interrupt-result byte in the TxI/R register, and reset when the processor reads the TxI/R register.

Bit 1  This bit is the receiver interrupt result available (RxIRA) bit. It is the corresponding result-available bit for the receiver. It is set when the 8273 places an interrupt-result byte in the RxI/R register and reset when the processor reads the register.

Bit 2  This bit is the transmitter interrupt (TxINT) bit and reflects the state of the TxINT pin. TxINT is set by the 8273 whenever the transmitter needs servicing, and reset when the processor reads the result or performs the data transfer.
Bit 3  This bit is the receiver interrupt (RxINT) bit and is identical to the TxINT, except action is initiated based on receiver interrupt-sources.

Bit 4  This bit is the command result buffer full (CRBF) bit. It is set when the 8273 places a result from an immediate-type command in the result register, and reset when the processor reads the result or performs the data transfer.

Bit 5  This bit is the command parameter buffer full (CPBF) bit and indicates that the parameter register contains a parameter. It is set when the processor deposits a parameter in the parameter register, and reset when the 8273 accepts the parameter.

Bit 6  This bit is the command buffer full (CBF) bit and, when set, it indicates that a byte is present in the command register. This bit is normally not used.

Bit 7  This bit is the command busy (CBSY) bit and indicates when the 8273 is in the command phase. It is set when the processor writes a command into the command register, starting the command phase. It is reset when the last parameter is deposited in the parameter register and accepted by the 8273, completing the command phase.
Initializing the Adapter (Typical Sequence)

Before initialization of the 8273 protocol controller, the support devices on the card must be initialized to the proper modes of operation.

Configuration of the 8255A-5 programmable peripheral interface is accomplished by selecting the mode-set address for the 8255 (see the "SDLC Communications Adapter Device Addresses" table later in this section) and writing the appropriate control word to the device (hex 98) to set ports A, B, and C to the modes described previously in this section.

Next, a bit pattern is output to port C which disallows interrupts, sets wrap mode on, and gates the external clock pins (address = hex 382, data = hex OD). The adapter is now isolated from the communications interface.

Using bit 4 of port B, the 8273 reset line is brought high, held and then dropped. This resets the internal registers of the 8273.

The 8253-5's counter 1 and 2 terminal-count values are now set to values which will provide the desired time delay before a level 4 interrupt is generated. These interrupts may be used to indicate to the communication software that a pre-determined period of time has elapsed without a result interrupt (interrupt level 3). The terminal count-values for these counters are set for any time delay which the programmer requires. Counter 0 is also set at this time to mode 3 (generates square wave signal, used to drive counter 2 input).

To setup the counter modes, the address for the 8253 counter mode register is selected (see the "SDLC Communications Adapter Device Addresses" table, later in this section), and the control word for each individual counter is written to the device separately. The control-word format and bit definitions for the 8253 are shown below. Note that the two most-significant bits of the control word select each individual counter, and each counter mode is defined separately.

Once the support devices have been initialized to the proper modes and the 8273 has been reset, the 8273 protocol controller is ready to be configured for the operating mode that defines the communications environment in which it will be used.

1-278 SDLC Adapter
## Control Word Format

<table>
<thead>
<tr>
<th>D_7</th>
<th>D_6</th>
<th>D_5</th>
<th>D_4</th>
<th>D_3</th>
<th>D_2</th>
<th>D_1</th>
<th>D_0</th>
</tr>
</thead>
<tbody>
<tr>
<td>SC1</td>
<td>SC0</td>
<td>RL1</td>
<td>RLO</td>
<td>M2</td>
<td>M1</td>
<td>M0</td>
<td>BCD</td>
</tr>
</tbody>
</table>

### Definitions of Control

#### SC - Select Counter:

<table>
<thead>
<tr>
<th>SC1</th>
<th>SC0</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Select Counter 0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Select Counter 1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Select Counter 2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Illegal</td>
</tr>
</tbody>
</table>

#### RL - Read/Load:

<table>
<thead>
<tr>
<th>RL1</th>
<th>RLO</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Counter Latching operation</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Read/Load most significant byte (MSB)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Read/Load least significant byte (LSB)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Read/Load least significant byte first, then most significant byte.</td>
</tr>
</tbody>
</table>

#### M - Mode:

<table>
<thead>
<tr>
<th>M2</th>
<th>M1</th>
<th>M0</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Mode 0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Mode 1</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
<td>0</td>
<td>Mode 2</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
<td>1</td>
<td>Mode 3</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Mode 4</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Mode 5</td>
</tr>
</tbody>
</table>

#### BCD:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Binary Counter 16-bits</td>
</tr>
<tr>
<td>1</td>
<td>Binary Coded Decimal (BCD) Counter (4 Decades)</td>
</tr>
</tbody>
</table>

---

**8253-5 Programmable Interval Timer Control Word**
Initialization/Configuration Commands

The initialization/configuration commands manipulate internal registers of the 8273, which define operating modes. After chip reset, the 8273 defaults to all 1’s in the mode registers. The initialization/configuration commands either set or reset specified bits in the registers depending on the type of command. One parameter is required with the commands. The parameter is actually the bit pattern (mask) used by the set or reset command to manipulate the register bits.

Set commands perform a logical OR operation of the parameter (mask) of the internal register. This mask contains 1’s where register bits are to be set. Zero (0’s) in the mask cause no change to the corresponding register bit.

Reset commands perform a logical AND operation of the parameter (mask) and internal register. The mask 0 is reset to register bit, and 1 to cause no change.

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

Operating Mode Register

<table>
<thead>
<tr>
<th>8273 Operating Mode Register Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>1 = Flag Stream Mode</td>
</tr>
<tr>
<td>1 = Two Preframe Sync Characters</td>
</tr>
<tr>
<td>1 = Buffered Mode</td>
</tr>
<tr>
<td>1 = Enable Early Tx Interrupt</td>
</tr>
<tr>
<td>1 = EOP Interrupt Enable</td>
</tr>
<tr>
<td>1 = HDLC Abort Enable</td>
</tr>
<tr>
<td>Not Used</td>
</tr>
<tr>
<td>Not Used</td>
</tr>
</tbody>
</table>
Bit 0  If bit 0 is set to a 1, flags are sent immediately if the transmitter was idle when the bit was set. If a transmit or transmit-transparent command was active, flags are sent immediately after transmit completion. This mode is ignored if loop transmit is active or the one-bitdelay mode register is set for one-bit delay. If bit 0 is reset (to 0), the transmitter sends idles on the next character boundary if idle or, after transmission is complete, if the transmitter was active at bit-0 reset time.

Bit 1  If bit 1 is set to a 1, the 8273 sends two characters before the first flag of a frame. These characters are hex 00 if NRZI is set or hex 55 if NRZI is not set. (See "Serial I/O Mode Register," for NRZI encoding mode format.)

Bit 2  If bit 2 is set to a 1, the 8273 buffers the first two bytes of a received frame (the bytes are not passed to memory). Resetting this bit (to 0) causes these bytes to be passed to and from memory.

Bit 3  This bit indicates to the 8273 when to generate an end-of-frame interrupt. If bit 3 is set, an early interrupt is generated when the last data character has been passed to the 8273. If the processor responds to the early interrupt with another transmit command before the final flag is sent, the final-flag interrupt will not be generated and a new frame will begin when the current frame is complete. Thus, frames may be sent separated by a single flag. A reset condition causes an interrupt to be generated only following a final flag.

Bit 4  This is the EOP-interrupt-mode function and is not used on the SDLC communications adapter. This bit should always be in the reset condition.

Bit 5  This bit is always reset for SDLC operation, which causes the 8273 protocol controller to recognize eight ones (0 1 1 1 1 1 1 1) as an abort character.
Serial I/O Mode Register

<table>
<thead>
<tr>
<th>8273 Serial I/O Mode Register Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

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

Bit 1  When bit 1 is set to 1, the transmit clock is internally routed to the receive-clock circuitry. It is normally used with the loopback bit (bit 2). The reset condition causes the transmit and receive clocks to be routed to their respective 8273 I/O pins.

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

Data Transfer Mode Register

<table>
<thead>
<tr>
<th>8273 Data Transfer Mode Register Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

1-282  SDLC Adapter
When the data transfer mode register is set, the 8273 protocol controller will interrupt when data bytes are required for transmission, or are available from a reception. If a transmit or receive interrupt occurs and the status register indicates that there is no transmit or receive interrupt result, the interrupt is a transmit or receive data request, respectively. Reset of this register causes DMA requests to be performed with no interrupts to the processor.

One-Bit Delay Mode Register

<table>
<thead>
<tr>
<th>8273 One-Bit Delay Mode Register Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7 6 5 4 3 2 1 0</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Not Used</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>1 = One-Bit Delay Enable</td>
</tr>
</tbody>
</table>

When one-bit delay is set, the 8273 retransmits the received data stream one-bit delayed. Reset of this bit stops the one-bit delay mode.

The table below is a summary of all set and reset commands associated with the 8273 mode registers. The set or reset mask used to define individual bits is treated as a single parameter. No result or interrupt is generated by the 8273 after execution of these commands.

<table>
<thead>
<tr>
<th>Register</th>
<th>Command</th>
<th>Hex Code</th>
<th>Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>One-Bit Delay Mode</td>
<td>Set</td>
<td>A4</td>
<td>Set Mask</td>
</tr>
<tr>
<td></td>
<td>Reset</td>
<td>64</td>
<td>Reset</td>
</tr>
<tr>
<td>Data Transfer Mode</td>
<td>Set</td>
<td>97</td>
<td>Set Mask</td>
</tr>
<tr>
<td></td>
<td>Reset</td>
<td>57</td>
<td>Reset Mask</td>
</tr>
<tr>
<td>Operating Mode</td>
<td>Set</td>
<td>91</td>
<td>Set Mask</td>
</tr>
<tr>
<td></td>
<td>Reset</td>
<td>51</td>
<td>Reset Mask</td>
</tr>
<tr>
<td>Serial I/O Mode</td>
<td>Set</td>
<td>A0</td>
<td>Set Mask</td>
</tr>
<tr>
<td></td>
<td>Reset</td>
<td>60</td>
<td>Reset Mask</td>
</tr>
</tbody>
</table>

8273 SDLC Protocol Controller Mode Register Commands
Command Phase

Although the $8273$ is a full duplex device, there is only one command register. Thus, the command register must be used for only one command sequence at a time and the transmitter and receiver may never be simultaneously in a command phase.

The system software starts the command phase by selecting the $8273$ command register address and writing a command byte into the register. The following table lists command and parameter information for the $8273$ protocol controller. If further information is required by the $8273$ prior to execution of the command, the system software must write this information into the parameter register.
<table>
<thead>
<tr>
<th>Command Description</th>
<th>Command (Hex)</th>
<th>Parameter</th>
<th>Results</th>
<th>Result Port</th>
<th>Completion Interrupt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set One-Bit Delay</td>
<td>A4</td>
<td>Set Mask</td>
<td>None</td>
<td>—</td>
<td>No</td>
</tr>
<tr>
<td>Reset One-Bit Delay</td>
<td>64</td>
<td>Reset Mask</td>
<td>None</td>
<td>—</td>
<td>No</td>
</tr>
<tr>
<td>Set Data Transfer Mode</td>
<td>97</td>
<td>Set Mask</td>
<td>None</td>
<td>—</td>
<td>No</td>
</tr>
<tr>
<td>Reset Data Transfer Mode</td>
<td>57</td>
<td>Reset Mask</td>
<td>None</td>
<td>—</td>
<td>No</td>
</tr>
<tr>
<td>Set Operating Mode</td>
<td>91</td>
<td>Set Mask</td>
<td>None</td>
<td>—</td>
<td>No</td>
</tr>
<tr>
<td>Reset Operating Mode</td>
<td>51</td>
<td>Reset Mask</td>
<td>None</td>
<td>—</td>
<td>No</td>
</tr>
<tr>
<td>Set Serial I/O Mode</td>
<td>A0</td>
<td>Set Mask</td>
<td>None</td>
<td>—</td>
<td>No</td>
</tr>
<tr>
<td>Reset Serial I/O Mode</td>
<td>60</td>
<td>Reset Mask</td>
<td>None</td>
<td>—</td>
<td>No</td>
</tr>
<tr>
<td>General Receive</td>
<td>C0</td>
<td>B0, B1</td>
<td>RIC, R0, R1, A, C</td>
<td>RXI/R</td>
<td>Yes</td>
</tr>
<tr>
<td>Selective Receive</td>
<td>C1</td>
<td>B0, B1, A1, A2</td>
<td>RIC, R0, R1, A, C</td>
<td>RXI/R</td>
<td>Yes</td>
</tr>
<tr>
<td>Receive Disable</td>
<td>C5</td>
<td>None</td>
<td>None</td>
<td>—</td>
<td>No</td>
</tr>
<tr>
<td>Transmit Frame</td>
<td>C8</td>
<td>L0, L1, A, C</td>
<td>TIC</td>
<td>TXI/R</td>
<td>Yes</td>
</tr>
<tr>
<td>Transmit Transparent</td>
<td>C9</td>
<td>L0, L1</td>
<td>TIC</td>
<td>TXI/R</td>
<td>Yes</td>
</tr>
<tr>
<td>Abort Transmit Frame</td>
<td>CC</td>
<td>None</td>
<td>TIC</td>
<td>TXI/R</td>
<td>Yes</td>
</tr>
<tr>
<td>Abort Transmit Transparent</td>
<td>CD</td>
<td>None</td>
<td>TIC</td>
<td>TXI/R</td>
<td>Yes</td>
</tr>
<tr>
<td>Read Port A</td>
<td>22</td>
<td>None</td>
<td>Port Value</td>
<td>Result</td>
<td>No</td>
</tr>
<tr>
<td>Read Port B</td>
<td>23</td>
<td>None</td>
<td>Port Value</td>
<td>Result</td>
<td>No</td>
</tr>
<tr>
<td>Set Port B Bit</td>
<td>A3</td>
<td>Set Mask</td>
<td>None</td>
<td>—</td>
<td>No</td>
</tr>
<tr>
<td>Reset Port B Bit</td>
<td>63</td>
<td>Reset Mask</td>
<td>None</td>
<td>—</td>
<td>No</td>
</tr>
</tbody>
</table>

8273 Command Summary Key

- **B0**: Least significant byte of the receiver buffer length.
- **B1**: Most significant byte of the receiver buffer length.
- **L0**: Least significant byte of the Tx frame length.
- **L1**: Most significant byte of the Tx frame length.
- **A1**: Receive frame address match field one.
- **A2**: Receive frame address match field two.
- **A**: Address field of received frame. If non-buffered mode is specified, this result is not provided.
- **C**: Control field of received frame. If non-buffered mode is specified, this result is not provided.
- **RXI/R**: Receive interrupt result register.
- **TXI/R**: Transmit interrupt result register.
- **R0**: Least significant byte of the length of the frame received.
- **R1**: Most significant byte of the length of the frame received.
- **RIC**: Receiver interrupt result code.
- **TIC**: Transmitter interrupt result code.
A flowchart of the command phase is shown below. Handshaking of the command and parameter bytes is accomplished by the CBSY and CPBF bits of the status register. A command may not be written if the 8273 is busy (CBSY = 1). The original command will be overwritten if a second command is issued while CBSY = 1. The flowchart also indicates a parameter buffer full check. The processor must wait until CPBF = 0 before writing a parameter to the parameter register. Previous parameters are overwritten and lost if a parameter is written while CPBF = 1.
Execution Phase

During the execution phase, the operation specified by the command phase is performed. If DMA is utilized for data transfers, no processor involvement is required.

For interrupt-driven transfers the 8273 raises the appropriate INT pin (TxINT or RxINT). When the processor responds to the interrupt, it must determine the cause by examining the status register and the associated IRA (interrupt result available) bit of the status register. If IRA = 0, the interrupt is a data transfer request. If IRA = 1, an operation is complete and the associated interrupt result register must be read to determine completion status.

Result Phase

During the result phase, the 8273 notifies the processor of the outcome of a command execution. This phase is initiated by either a successful completion or error detection during execution.

Some commands such as reading or writing the I/O ports provide immediate results. These results are made available to the processor in the 8273 result register. Presence of a valid immediate result is indicated by the CRBF (command result buffer full) bit of the status register.

Non-immediate results deal with the transmitter and receiver. These results are provided in the TxI/R (transmit interrupt result) or RxI/R (receiver interrupt result) registers, respectively. The 8273 notifies the processor that a result is available with the TxIRA and RxIRA bits of the status register. Results consist of one-byte result interrupt code indicating the condition for the interrupt and, if required, one or more bytes supplying additional information. The "Result Code Summary" table later in this section provides information on the format and decode of the transmitter and receiver results.

The following are typical frame transmit and receive sequences. These examples assume DMA is utilized for data transfer operations.
Transmit

Before a frame can be transmitted, the DMA controller is supplied, by the communication software, the starting address for the desired information field. The $8273$ is then commanded to transmit a frame (by issuing a transmit frame command).

After a command, but before transmission begins, the $8273$ needs some more information (parameters). Four parameters are required for the transmit frame command; the frame address field byte, the frame control field byte, and two bytes which are the least significant and most significant bytes of the information field byte length. Once all four parameters are loaded, the $8273$ makes RTS (request to send) active and waits for CTS (clear to send) to go active from the modem interface. Once CTS is active, the $8273$ starts the frame transmission. While the $8273$ is transmitting the opening flag, address field, and control field, it starts making transmitter DMA requests. These requests continue at character (byte) boundaries until the pre-loaded number of bytes of information field have been transmitted. At this point, the requests stop, the FCS (frame check sequence) and closing flag are transmitted, and the $\text{TxINT}$ line is raised, signaling the processor the frame transmission is complete and the result should be read. Note that after the initial command and parameter loading, no processor intervention was required (since DMA is used for data transfers) until the entire frame was transmitted.

General Receive

Receiver operation is very similar. Like the initial transmit sequence, the processor's DMA controller is loaded with a starting address for a receive data buffer and the $8273$ is commanded to receive. Unlike the transmitter, there are two different receive commands; a general receive, where all received frames are transferred to memory, and selective receive, where only frames having an address field matching one of two preprogrammed $8273$ address fields are transferred to memory.
(This example covers a general receive operation.) After the receive command, two parameters are required before the receiver becomes active; the least significant and most significant bytes of the receiver buffer length. Once these bytes are loaded, the receiver is active and the processor may return to other tasks. The next frame appearing at the receiver input is transferred to memory using receiver DMA requests. When the closing flag is received, the 8273 checks the FCS and raises its RxINT line. The processor can then read the results, which indicate if the frame was error-free or not. (If the received frame had been longer than the pre-loaded buffer length, the processor would have been notified of that occurrence earlier with a receiver error interrupt). Like the transmit example, after the initial command, the processor is free for other tasks until a frame is completely received.

Selective Receive

In selective receive, two parameters (A1 and A2) are required in addition to those for general receive. These parameters are two address match bytes. When commanded to selective receive, the 8273 passes to memory or the processor only those frames having an address field matching either A1 or A2. This command is usually used for secondary stations with A1 designating the secondary address and A2 being the "all parties" address. If only one match byte is needed, A1 and A2 should be equal. As in general receive, the 8273 counts the incoming data bytes and interrupts the processor if the received frame is larger than the preset receive buffer length.
## Result Code Summary

<table>
<thead>
<tr>
<th>Hex Code</th>
<th>Result</th>
<th>Status After Interrupt</th>
</tr>
</thead>
<tbody>
<tr>
<td>OC</td>
<td>Early Transmit Interrupt</td>
<td>Transmitter Active</td>
</tr>
<tr>
<td>OD</td>
<td>Frame Transmit Complete</td>
<td>Idle or Flags</td>
</tr>
<tr>
<td>OE</td>
<td>DMA Underrun</td>
<td>Abort</td>
</tr>
<tr>
<td>OF</td>
<td>Clear to Send Error</td>
<td>Abort</td>
</tr>
<tr>
<td>10</td>
<td>Abort Complete</td>
<td>Idle or Flags</td>
</tr>
<tr>
<td>X0</td>
<td>A1 Match or General Receive</td>
<td>Active</td>
</tr>
<tr>
<td>X1</td>
<td>A2 Match</td>
<td>Active</td>
</tr>
<tr>
<td>03</td>
<td>CRC Error</td>
<td>Active</td>
</tr>
<tr>
<td>04</td>
<td>Abort Detected</td>
<td>Active</td>
</tr>
<tr>
<td>05</td>
<td>Idle Detected</td>
<td>Disabled</td>
</tr>
<tr>
<td>06</td>
<td>EOP Detected</td>
<td>Disabled</td>
</tr>
<tr>
<td>07</td>
<td>Frame Less Than 32 Bits</td>
<td>Active</td>
</tr>
<tr>
<td>08</td>
<td>DMA Overrun</td>
<td>Disabled</td>
</tr>
<tr>
<td>09</td>
<td>Memory Buffer Overflow</td>
<td>Disabled</td>
</tr>
<tr>
<td>0A</td>
<td>Carrier Detect Failure</td>
<td>Disabled</td>
</tr>
<tr>
<td>0B</td>
<td>Receiver Interrupt Overrun</td>
<td>Disabled</td>
</tr>
</tbody>
</table>

**Note:** X decodes to number of bits in partial byte received.

The first two codes in the receive result code table result from the error free reception of a frame. Since SDLC allows frames of arbitrary length (>32 bits), the high order bits of the receive result report the number of valid received bits in the last received information field byte. The chart below shows the decode of this receive result bit.

<table>
<thead>
<tr>
<th>X</th>
<th>Bits Received in Last Byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>All Eight Bits of Last Byte</td>
</tr>
<tr>
<td>O</td>
<td>Bit0 Only</td>
</tr>
<tr>
<td>B</td>
<td>Bit1-Bit0</td>
</tr>
<tr>
<td>4</td>
<td>Bit2-Bit0</td>
</tr>
<tr>
<td>C</td>
<td>Bit3-Bit0</td>
</tr>
<tr>
<td>2</td>
<td>Bit4-Bit0</td>
</tr>
<tr>
<td>A</td>
<td>Bit5-Bit0</td>
</tr>
<tr>
<td>6</td>
<td>Bit6-Bit0</td>
</tr>
</tbody>
</table>

1-290 SDLC Adapter
Address and Interrupt Information

The following tables provide address and interrupt information for the SDLC adapter:

### SDLC Communications Adapter Device Addresses

<table>
<thead>
<tr>
<th>Hex Code</th>
<th>Device</th>
<th>Register Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>380</td>
<td>8255</td>
<td>Port A Data</td>
<td>Internal/External Sensing</td>
</tr>
<tr>
<td>381</td>
<td>8255</td>
<td>Port B Data</td>
<td>External Modem Interface</td>
</tr>
<tr>
<td>382</td>
<td>8255</td>
<td>Port C Data</td>
<td>Internal Control</td>
</tr>
<tr>
<td>383</td>
<td>8255</td>
<td>Mode Set</td>
<td>8255 Mode Initialization</td>
</tr>
<tr>
<td>384</td>
<td>8253</td>
<td>Counter 0 LSB</td>
<td>Square Wave Generator</td>
</tr>
<tr>
<td>384</td>
<td>8253</td>
<td>Counter 0 MSB</td>
<td>Square Wave Generator</td>
</tr>
<tr>
<td>385</td>
<td>8253</td>
<td>Counter 1 LSB</td>
<td>Inactivity Time-outs</td>
</tr>
<tr>
<td>385</td>
<td>8253</td>
<td>Counter 1 MSB</td>
<td>Inactivity Time-outs</td>
</tr>
<tr>
<td>386</td>
<td>8253</td>
<td>Counter 2 LSB</td>
<td>Inactivity Time-outs</td>
</tr>
<tr>
<td>386</td>
<td>8253</td>
<td>Counter 2 MSB</td>
<td>Inactivity Time-outs</td>
</tr>
<tr>
<td>387</td>
<td>8253</td>
<td>Mode Register</td>
<td>8253 Mode Set</td>
</tr>
<tr>
<td>388</td>
<td>8273</td>
<td>Command/Status</td>
<td>Out=Command In=Status</td>
</tr>
<tr>
<td>389</td>
<td>8273</td>
<td>Parameter/Result</td>
<td>Out=Parameter In=Status</td>
</tr>
<tr>
<td>38A</td>
<td>8273</td>
<td>Transmit INT Status</td>
<td>DMA/INT</td>
</tr>
<tr>
<td>38B</td>
<td>8273</td>
<td>Receive INT Status</td>
<td>DMA/INT</td>
</tr>
<tr>
<td>38C</td>
<td>8273</td>
<td>Data</td>
<td>DPC (Direct Program Control)</td>
</tr>
</tbody>
</table>

### Interrupt Information

<table>
<thead>
<tr>
<th>Interrupt Level</th>
<th>Transmit/Receive Interrupt</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Timer 1 Interrupt</td>
</tr>
<tr>
<td></td>
<td>Timer 2 Interrupt</td>
</tr>
<tr>
<td></td>
<td>Clear to Send Changed</td>
</tr>
<tr>
<td></td>
<td>Data Set Ready Changed</td>
</tr>
</tbody>
</table>

DMA Level One is used for Transmit and Receive
Interface Information

The SDLC communications adapter conforms to interface signal levels standardized by the Electronics Industries Association RC-232C Standard. These levels are shown in the figure below.

Additional lines used but not standardized by EIA are pins 11, 18, and 25. These lines are designated as select standby, test and test indicate, respectively. Select Standby is used to support the switched network backup facility of a modem providing this option. Test and test indicate support a modem wrap function on modems which are designed for business machine controlled modem wraps. Two jumpers on the adapter (P1 and P2) are used to connect test and test indicate to the interface, if required (see Appendix D for these jumpers).
<table>
<thead>
<tr>
<th>Signal Name — Description</th>
<th>Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Connection</td>
<td>1</td>
</tr>
<tr>
<td>Transmitted Data</td>
<td>2</td>
</tr>
<tr>
<td>Received Data</td>
<td>3</td>
</tr>
<tr>
<td>Request to Send</td>
<td>4</td>
</tr>
<tr>
<td>Clear to Send</td>
<td>5</td>
</tr>
<tr>
<td>Data Set Ready</td>
<td>6</td>
</tr>
<tr>
<td>Signal Ground</td>
<td>7</td>
</tr>
<tr>
<td>Received Line Signal Detector</td>
<td>8</td>
</tr>
<tr>
<td>No Connection</td>
<td>9</td>
</tr>
<tr>
<td>No Connection</td>
<td>10</td>
</tr>
<tr>
<td>Select Standby*</td>
<td>11</td>
</tr>
<tr>
<td>No Connection</td>
<td>12</td>
</tr>
<tr>
<td>No Connection</td>
<td>13</td>
</tr>
<tr>
<td>No Connection</td>
<td>14</td>
</tr>
<tr>
<td>Transmitter Signal Element Timing</td>
<td>15</td>
</tr>
<tr>
<td>No Connection</td>
<td>16</td>
</tr>
<tr>
<td>Receiver Signal Element Timing</td>
<td>17</td>
</tr>
<tr>
<td>Test (IBM Modems Only)*</td>
<td>18</td>
</tr>
<tr>
<td>No Connection</td>
<td>19</td>
</tr>
<tr>
<td>Data Terminal Ready</td>
<td>20</td>
</tr>
<tr>
<td>No Connection</td>
<td>21</td>
</tr>
<tr>
<td>Ring Indicator</td>
<td>22</td>
</tr>
<tr>
<td>Data Signal Rate Selector</td>
<td>23</td>
</tr>
<tr>
<td>No Connection</td>
<td>24</td>
</tr>
<tr>
<td>Test Indicate (IBM Modems Only)*</td>
<td>25</td>
</tr>
</tbody>
</table>

*Not standardized by EIA (Electronics Industry Association).

Connector Specifications
Notes:
IBM Communications Adapter Cable

The IBM Communications Adapter Cable is a ten foot cable for connection of an IBM communications adapter to a modem or other RC-232C DCE (data communications equipment). It is fully shielded and provides a high quality, low noise channel for interface between the communications adapter and DCE.

The connector ends are 25-pin D-shell connectors. All pin connections conform with the EIA RS-232C standard. In addition, connection is provided on pins 11, 18 and 25. These pins are designated as select standby, test and test indicate, respectively, on some modems. Select standby is used to support the switched network backup facility, if applicable. Test and test indicate support a modem wrap function on modems designed for business machine controlled modem wraps.
The IBM Communications Adapter Cable connects the following pins on the 25-pin D-shell connectors.

![Diagram of IBM Communications Adapter Cable](image)

<table>
<thead>
<tr>
<th>Communications Adapter Connector Pin #</th>
<th>Name</th>
<th>Modem Connector Pin #</th>
</tr>
</thead>
<tbody>
<tr>
<td>NC</td>
<td>Outer Cable Shield</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>Transmitted Data</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Received Data</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>Request to Send</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>Clear to Send</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>Data Set Ready</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>Signal Ground (Inner Lead Shields)</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>Received Line Signal Detector</td>
<td>8</td>
</tr>
<tr>
<td>NC</td>
<td></td>
<td>NC</td>
</tr>
<tr>
<td>NC</td>
<td></td>
<td>NC</td>
</tr>
<tr>
<td>NC</td>
<td></td>
<td>NC</td>
</tr>
<tr>
<td>11</td>
<td>Select Standby</td>
<td>11</td>
</tr>
<tr>
<td>NC</td>
<td></td>
<td>NC</td>
</tr>
<tr>
<td>NC</td>
<td></td>
<td>NC</td>
</tr>
<tr>
<td>NC</td>
<td></td>
<td>NC</td>
</tr>
<tr>
<td>15</td>
<td>Transmitter Signal Element Timing</td>
<td>15</td>
</tr>
<tr>
<td>NC</td>
<td>Receiver Signal Element Timing</td>
<td>NC</td>
</tr>
<tr>
<td>17</td>
<td>Test</td>
<td>17</td>
</tr>
<tr>
<td>NC</td>
<td>Data Terminal Ready</td>
<td>18</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>NC</td>
</tr>
<tr>
<td>NC</td>
<td>Ring Indicator</td>
<td>22</td>
</tr>
<tr>
<td>22</td>
<td>Data Signal Rate Selector</td>
<td>23</td>
</tr>
<tr>
<td>NC</td>
<td></td>
<td>NC</td>
</tr>
<tr>
<td>25</td>
<td>Test Indicate</td>
<td>25</td>
</tr>
</tbody>
</table>

**Connector Specifications**

1-296 Communications Cable
SECTION 2: ROM BIOS AND SYSTEM USAGE

ROM BIOS ........................................ 2-2
Keyboard Encoding and Usage ...................... 2-11
The basic input/output system (BIOS) resides in ROM on the system board and provides device level control for the major I/O devices in the system. Additional ROM modules may be located on option adapters to provide device level control for that option adapter. BIOS routines enable the assembly language programmer to perform block (disk and diskette) or character-level I/O operations without concern for device address and operating characteristics. System services, such as time-of-day and memory size determination, are provided by the BIOS.

The goal is to provide an operational interface to the system and relieve the programmer of the concern about the characteristics of hardware devices. The BIOS interface insulates the user from the hardware, thus allowing new devices to be added to the system, yet retaining the BIOS level interface to the device. In this manner, user programs become transparent to hardware modifications and enhancements.

The IBM Personal Computer MACRO Assembler manual and the IBM Personal Computer Disk Operating System (DOS) manual provide useful programming information related to this section. A complete listing of the BIOS is given in Appendix A.

**Use of BIOS**

Access to BIOS is through the 8088 software interrupts. Each BIOS entry point is available through its own interrupt, which can be found in the "8088 Software Interrupt Listing."

The software interrupts, hex 10 through hex 1A, each access a different BIOS routine. For example, to determine the amount of memory available in the system,

```
INT 12H
```

will invoke the BIOS routine for determining memory size and will return the value to the caller.
Parameter Passing

All parameters passed to and from the BIOS routines go through the 8088 registers. The prolog of each BIOS function indicates the registers used on the call and the return. For the memory size example, no parameters are passed. The memory size, in 1K byte increments, is returned in the AX register.

If a BIOS function has several possible operations, the AH register is used at input to indicate the desired operation. For example, to set the time of day, the following code is required:

```
MOV AH,1 ;function is to set time of day.
MOV CX,HIGH_COUNT ;establish the current time.
MOV DX,LOW_COUNT
INT 1AH :set the time.
```

To read the time of day:

```
MOV AH,0 ;function is to read time of day.
INT 1AH ;read the timer.
```

Generally, the BIOS routines save all registers except for AX and the flags. Other registers are modified on return only if they are returning a value to the caller. The exact register usage can be seen in the prolog of each BIOS function.
<table>
<thead>
<tr>
<th>Address (Hex)</th>
<th>Interrupt Number</th>
<th>Name</th>
<th>BIOS Entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-3</td>
<td>0</td>
<td>Divide by Zero</td>
<td>D11</td>
</tr>
<tr>
<td>4-7</td>
<td>1</td>
<td>Single Step</td>
<td>D11</td>
</tr>
<tr>
<td>8-B</td>
<td>2</td>
<td>Nonmaskable</td>
<td>NMI_INT</td>
</tr>
<tr>
<td>C-F</td>
<td>3</td>
<td>Breakpoint</td>
<td>D11</td>
</tr>
<tr>
<td>10-13</td>
<td>4</td>
<td>Overflow</td>
<td>D11</td>
</tr>
<tr>
<td>14-17</td>
<td>5</td>
<td>Print Screen</td>
<td>PRINT_SCREEN</td>
</tr>
<tr>
<td>18-1B</td>
<td>6</td>
<td>Reserved</td>
<td>D11</td>
</tr>
<tr>
<td>1D-1F</td>
<td>7</td>
<td>Reserved</td>
<td>D11</td>
</tr>
<tr>
<td>20-23</td>
<td>8</td>
<td>Time of Day</td>
<td>TIMER_INT</td>
</tr>
<tr>
<td>24-27</td>
<td>9</td>
<td>Keyboard</td>
<td>KB_INT</td>
</tr>
<tr>
<td>28-2B</td>
<td>A</td>
<td>Reserved</td>
<td>D11</td>
</tr>
<tr>
<td>2C-2F</td>
<td>B</td>
<td>Communications</td>
<td>D11</td>
</tr>
<tr>
<td>30-33</td>
<td>C</td>
<td>Communications</td>
<td>D11</td>
</tr>
<tr>
<td>34-37</td>
<td>D</td>
<td>Disk</td>
<td>D11</td>
</tr>
<tr>
<td>38-3B</td>
<td>E</td>
<td>Diskette</td>
<td>DISK_INT</td>
</tr>
<tr>
<td>3C-3F</td>
<td>F</td>
<td>Printer</td>
<td>D11</td>
</tr>
<tr>
<td>40-43</td>
<td>10</td>
<td>Video</td>
<td>VIDEO_IO</td>
</tr>
<tr>
<td>44-47</td>
<td>11</td>
<td>Equipment Check</td>
<td>EQUIPMENT</td>
</tr>
<tr>
<td>48-4B</td>
<td>12</td>
<td>Memory</td>
<td>MEMORY_SIZE_DETERMINE</td>
</tr>
<tr>
<td>4C-4F</td>
<td>13</td>
<td>Diskette/Disk</td>
<td>DISKETTE_IO</td>
</tr>
<tr>
<td>50-53</td>
<td>14</td>
<td>Communications</td>
<td>RS232_IO</td>
</tr>
<tr>
<td>54-57</td>
<td>15</td>
<td>Cassette</td>
<td>CASSETTE_IO</td>
</tr>
<tr>
<td>58-5B</td>
<td>16</td>
<td>Keyboard</td>
<td>KEYBOARD_IO</td>
</tr>
<tr>
<td>5C-5F</td>
<td>17</td>
<td>Printer</td>
<td>PRINTER_IO</td>
</tr>
<tr>
<td>60-63</td>
<td>18</td>
<td>Resident BASIC</td>
<td>F600:0000</td>
</tr>
<tr>
<td>64-67</td>
<td>19</td>
<td>Bootstrap</td>
<td>BOOT Strap</td>
</tr>
<tr>
<td>68-6B</td>
<td>1A</td>
<td>Time of Day</td>
<td>TIME_OF_DAY</td>
</tr>
<tr>
<td>6C-6F</td>
<td>1B</td>
<td>Keyboard Break</td>
<td>DUMMY_RETURN</td>
</tr>
<tr>
<td>70-73</td>
<td>1C</td>
<td>Timer Tick</td>
<td>DUMMY_RETURN</td>
</tr>
<tr>
<td>74-77</td>
<td>1D</td>
<td>Video Initialization</td>
<td>VIDEO_PARMs</td>
</tr>
<tr>
<td>78-7B</td>
<td>1E</td>
<td>Diskette Parameters</td>
<td>DISK_BASE</td>
</tr>
<tr>
<td>7C-7F</td>
<td>1F</td>
<td>Video Graphics Chars</td>
<td>0</td>
</tr>
</tbody>
</table>

**8088 Software Interrupt Listing**

**2-4 ROM BIOS**
Vectors with Special Meanings

Interrupt Hex 1B – Keyboard Break Address

This vector points to the code to be exercised when the Ctrl and Break keys are pressed on the keyboard. The vector is invoked while responding to the keyboard interrupt, and control should be returned through an IRET instruction. The power-on routines initialize this vector to point to an IRET instruction, so that nothing will occur when the Ctrl and Break keys are pressed unless the application program sets a different value.

Control may be retained by this routine, with the following problems. The Break may have occurred during interrupt processing, so that one or more End of Interrupt commands must be sent to the 8259 controller. Also, all I/O devices should be reset in case an operation was underway at that time.

Interrupt Hex 1C – Timer Tick

This vector points to the code to be executed on every system-clock tick. This vector is invoked while responding to the timer interrupt, and control should be returned through an IRET instruction. The power-on routines initialize this vector to point to an IRET instruction, so that nothing will occur unless the application modifies the pointer. It is the responsibility of the application to save and restore all registers that will be modified.

Interrupt Hex 1D – Video Parameters

This vector points to a data region containing the parameters required for the initialization of the 6845 on the video card. Note that there are four separate tables, and all four must be reproduced if all modes of operation are to be supported. The power-on routines initialize this vector to point to the parameters contained in the ROM video routines.

ROM BIOS 2-5
Interrupt Hex 1E – Diskette Parameters

This vector points to a data region containing the parameters required for the diskette drive. The power-on routines initialize the vector to point to the parameters contained in the ROM diskette routine. These default parameters represent the specified values for any IBM drives attached to the machine. Changing this parameter block may be necessary to reflect the specifications of the other drives attached.

Interrupt Hex 1F – Graphics Character Extensions

When operating in the graphics modes of the IBM Color/Graphics Monitor Adapter (320 by 200 or 640 by 200), the read/write character interface will form the character from the ASCII code point, using a set of dot patterns. The dot patterns for the first 128 code points are contained in ROM. To access the second 128 code points, this vector must be established to point at a table of up to 1K bytes, where each code point is represented by eight bytes of graphic information. At power-on, this vector is initialized to 000:0, and it is the responsibility of the user to change this vector if the additional code points are required.

Interrupt Hex 40 – Reserved

When an IBM Fixed Disk Drive Adapter is installed, the BIOS routines use interrupt hex 40 to revector the diskette pointer.

Interrupt Hex 41 – Fixed Disk Parameters

This vector points to a data region containing the parameters required for the fixed disk drive. The power-on routines initialize the vector to point to the parameters contained in the ROM disk routine. These default parameters represent the specified values for any IBM Fixed Disk Drives attached to the machine. Changing this parameter block may be necessary to reflect the specifications of the other fixed disk drives attached.
Other Read/Write Memory Usage

The IBM BIOS routines use 256 bytes of memory starting at absolute hex 400 to hex 4FF. Locations hex 400 to 407 contain the base addresses of any RS-232C cards attached to the system. Locations hex 408 to 40F contain the base addresses of the printer adapter.

Memory locations hex 300 to 3FF are used as a stack area during the power-on initialization, and bootstrap, when control is passed to it from power-on. If the user desires the stack in a different area, the area must be set by the application.

<table>
<thead>
<tr>
<th>Address (Hex)</th>
<th>Interrupt (Hex)</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>80-83</td>
<td>20</td>
<td>DOS Program Terminate</td>
</tr>
<tr>
<td>84-87</td>
<td>21</td>
<td>DOS Function Call</td>
</tr>
<tr>
<td>88-88</td>
<td>22</td>
<td>DOS Terminate Address</td>
</tr>
<tr>
<td>8C-8F</td>
<td>23</td>
<td>DOS Ctrl Break Exit Address</td>
</tr>
<tr>
<td>90-93</td>
<td>24</td>
<td>DOS Fatal Error Vector</td>
</tr>
<tr>
<td>94-97</td>
<td>25</td>
<td>DOS Absolute Disk Read</td>
</tr>
<tr>
<td>98-9B</td>
<td>26</td>
<td>DOS Absolute Disk Write</td>
</tr>
<tr>
<td>9C-9F</td>
<td>27</td>
<td>DOS Terminate, Fix In Storage</td>
</tr>
<tr>
<td>100-17F</td>
<td>28-3F</td>
<td>Reserved for DOS</td>
</tr>
<tr>
<td>180-19F</td>
<td>40-5F</td>
<td>Reserved</td>
</tr>
<tr>
<td>1A0-1FF</td>
<td>60-67</td>
<td>Reserved for User Software Interrupts</td>
</tr>
<tr>
<td>200-217</td>
<td>68-7F</td>
<td>Not Used</td>
</tr>
<tr>
<td>218-3C3</td>
<td>80-85</td>
<td>Reserved by BASIC</td>
</tr>
<tr>
<td>3C4-3FF</td>
<td>F1-FF</td>
<td>Not Used</td>
</tr>
</tbody>
</table>

BASIC and DOS Reserved Interrupts
<table>
<thead>
<tr>
<th>Address (Hex)</th>
<th>Mode</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>400-48F</td>
<td>ROM BIOS</td>
<td>See BIOS Listing</td>
</tr>
<tr>
<td>490-4EF</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>4F0-4FF</td>
<td>Reserved as Intra-Application Communication Area for any application</td>
<td></td>
</tr>
<tr>
<td>500-5FF</td>
<td>DOS</td>
<td>Print Screen Status Flag Store</td>
</tr>
<tr>
<td>500</td>
<td>DOS</td>
<td>0-Print Screen Not Active or Successful</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Print Screen Operation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1-Print Screen In Progress</td>
</tr>
<tr>
<td></td>
<td></td>
<td>255-Error Encountered during Print Screen Operation</td>
</tr>
<tr>
<td>504</td>
<td>DOS</td>
<td>Single Drive Mode Status Byte</td>
</tr>
<tr>
<td>510-511</td>
<td>BASIC</td>
<td>BASIC's Segment Address Store</td>
</tr>
<tr>
<td>512-515</td>
<td>BASIC</td>
<td>Clock Interrupt Vector Segment: Offset Store</td>
</tr>
<tr>
<td>516-519</td>
<td>BASIC</td>
<td>Break Key Interrupt Vector Segment: Offset Store</td>
</tr>
<tr>
<td>51A-51D</td>
<td>BASIC</td>
<td>Disk Error Interrupt Vector Segment: Offset Store</td>
</tr>
</tbody>
</table>

Reserved Memory Locations

If you do DEF SEG (Default workspace segment):

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

Example

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

```
    | L  | H  |
100 Hex 64 Hex 00
```

BASIC Workspace Variables

2-8 ROM BIOS
**BIOS Programming Hints**

The BIOS code is invoked through software interrupts. The programmer should not "hard code" BIOS addresses into applications. The internal workings and absolute addresses within BIOS are subject to change without notice.

If an error is reported by the disk or diskette code, you should reset the drive adapter and retry the operation. A specified number of retries should be required on diskette reads to ensure the problem is not due to motor start-up.

When altering I/O port bit values, the programmer should change only those bits which are necessary to the current task. Upon completion, the programmer should restore the original environment. Failure to adhere to this practice may be incompatible with present and future applications.
Adapter Cards with System-Accessible ROM Modules

The ROM BIOS provides a facility to integrate adapter cards with on board ROM code into the system. During the POST, interrupt vectors are established for the BIOS calls. After the default vectors are in place, a scan for additional ROM modules takes place. At this point, a ROM routine on the adapter card may gain control. The routine may establish or intercept interrupt vectors to hook themselves into the system.

The absolute addresses hex C8000 through hex F4000 are scanned in 2K blocks in search of a valid adapter card ROM. A valid ROM is defined as follows:

Byte 0:   Hex 55
Byte 1:   Hex AA
Byte 2:   A length indicator representing the number of 512 byte blocks in the ROM (length/512). A checksum is also done to test the integrity of the ROM module. Each byte in the defined ROM is summed modulo hex 100. This sum must be 0 for the module to be deemed valid.

When the POST identifies a valid ROM, it does a far call to byte 3 of the ROM (which should be executable code). The adapter card may now perform its power-on initialization tasks. The feature ROM should return control to the BIOS routines by executing a far return.
Keyboard Encoding and Usage

Encoding

The keyboard routine provided by IBM in the ROM BIOS is responsible for converting the keyboard scan codes into what will be termed “Extended ASCII.”

Extended ASCII encompasses one-byte character codes with possible values of 0 to 255, an extended code for certain extended keyboard functions, and functions handled within the keyboard routine or through interrupts.

Character Codes

The following character codes are passed through the BIOS keyboard routine to the system or application program. A “−1” means the combination is suppressed in the keyboard routine. The codes are returned in AL. See Appendix C for the exact codes. Also, see “Keyboard Scan Code Diagram” in Section 1.

<table>
<thead>
<tr>
<th>Key Number</th>
<th>Base Case</th>
<th>Upper Case</th>
<th>Ctrl</th>
<th>Alt</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Esc</td>
<td>Esc</td>
<td>Esc</td>
<td>-1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>!</td>
<td>-1</td>
<td>Note 1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>@</td>
<td>Nul (000) Note 1</td>
<td>Note 1</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>#</td>
<td>-1</td>
<td>Note 1</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>$</td>
<td>-1</td>
<td>Note 1</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>%</td>
<td>-1</td>
<td>Note 1</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>^</td>
<td>RS(030)</td>
<td>Note 1</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>&amp;</td>
<td>-1</td>
<td>Note 1</td>
</tr>
<tr>
<td>9</td>
<td>8</td>
<td>*</td>
<td>-1</td>
<td>Note 1</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>(</td>
<td>-1</td>
<td>Note 1</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>)</td>
<td>-1</td>
<td>Note 1</td>
</tr>
<tr>
<td>12</td>
<td>-</td>
<td>-</td>
<td>US(031)</td>
<td>Note 1</td>
</tr>
<tr>
<td>13</td>
<td>=</td>
<td>+</td>
<td>-1</td>
<td>Note 1</td>
</tr>
<tr>
<td>14</td>
<td>Backspace (008)</td>
<td>Backspace (008)</td>
<td>Del (127)</td>
<td>-1</td>
</tr>
<tr>
<td>15</td>
<td>←(009)</td>
<td>←(Note 1)</td>
<td>-1</td>
<td>Note 1</td>
</tr>
<tr>
<td>16</td>
<td>q</td>
<td>Q</td>
<td>DC1 (017)</td>
<td>Note 1</td>
</tr>
<tr>
<td>17</td>
<td>w</td>
<td>W</td>
<td>ETB (023)</td>
<td>Note 1</td>
</tr>
</tbody>
</table>

Character Codes (Part 1 of 3)
<table>
<thead>
<tr>
<th>Key Number</th>
<th>Base Case</th>
<th>Upper Case</th>
<th>Ctrl</th>
<th>Alt</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>e</td>
<td>E</td>
<td>ENQ (005)</td>
<td>Note 1</td>
</tr>
<tr>
<td>19</td>
<td>r</td>
<td>R</td>
<td>DC2 (018)</td>
<td>Note 1</td>
</tr>
<tr>
<td>20</td>
<td>t</td>
<td>T</td>
<td>DC4 (020)</td>
<td>Note 1</td>
</tr>
<tr>
<td>21</td>
<td>y</td>
<td>Y</td>
<td>EM (025)</td>
<td>Note 1</td>
</tr>
<tr>
<td>22</td>
<td>u</td>
<td>U</td>
<td>NAK (021)</td>
<td>Note 1</td>
</tr>
<tr>
<td>23</td>
<td>i</td>
<td>I</td>
<td>HT (009)</td>
<td>Note 1</td>
</tr>
<tr>
<td>24</td>
<td>o</td>
<td>O</td>
<td>SI (015)</td>
<td>Note 1</td>
</tr>
<tr>
<td>25</td>
<td>p</td>
<td>P</td>
<td>DLE (016)</td>
<td>Note 1</td>
</tr>
<tr>
<td>26</td>
<td>[</td>
<td>{</td>
<td>Esc (027)</td>
<td>-1</td>
</tr>
<tr>
<td>27</td>
<td>]</td>
<td>}</td>
<td>GS (029)</td>
<td>-1</td>
</tr>
<tr>
<td>28</td>
<td>CR</td>
<td>CR</td>
<td>LF (010)</td>
<td>-1</td>
</tr>
<tr>
<td>29 Ctrl</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>30</td>
<td>a</td>
<td>A</td>
<td>SOH (001)</td>
<td>Note 1</td>
</tr>
<tr>
<td>31</td>
<td>s</td>
<td>S</td>
<td>DC3 (019)</td>
<td>Note 1</td>
</tr>
<tr>
<td>32</td>
<td>d</td>
<td>D</td>
<td>EOT (004)</td>
<td>Note 1</td>
</tr>
<tr>
<td>33</td>
<td>f</td>
<td>F</td>
<td>ACK (006)</td>
<td>Note 1</td>
</tr>
<tr>
<td>34</td>
<td>g</td>
<td>G</td>
<td>BEL (007)</td>
<td>Note 1</td>
</tr>
<tr>
<td>35</td>
<td>h</td>
<td>H</td>
<td>BS (008)</td>
<td>Note 1</td>
</tr>
<tr>
<td>36</td>
<td>j</td>
<td>J</td>
<td>LF (010)</td>
<td>Note 1</td>
</tr>
<tr>
<td>37</td>
<td>k</td>
<td>K</td>
<td>VT (011)</td>
<td>Note 1</td>
</tr>
<tr>
<td>38</td>
<td>l</td>
<td>L</td>
<td>FF (012)</td>
<td>Note 1</td>
</tr>
<tr>
<td>39</td>
<td>;</td>
<td>,</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>40</td>
<td>'</td>
<td>&quot;</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>41</td>
<td>~</td>
<td>~</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>42 Shift</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>43</td>
<td>\</td>
<td>1</td>
<td>FS (028)</td>
<td>-1</td>
</tr>
<tr>
<td>44</td>
<td>z</td>
<td>Z</td>
<td>SUB (026)</td>
<td>Note 1</td>
</tr>
<tr>
<td>45</td>
<td>x</td>
<td>X</td>
<td>CAN (024)</td>
<td>Note 1</td>
</tr>
<tr>
<td>46</td>
<td>c</td>
<td>C</td>
<td>ETX (003)</td>
<td>Note 1</td>
</tr>
<tr>
<td>47</td>
<td>v</td>
<td>V</td>
<td>SYN (022)</td>
<td>Note 1</td>
</tr>
<tr>
<td>48</td>
<td>b</td>
<td>B</td>
<td>STX (002)</td>
<td>Note 1</td>
</tr>
<tr>
<td>49</td>
<td>n</td>
<td>N</td>
<td>SO (014)</td>
<td>Note 1</td>
</tr>
<tr>
<td>50</td>
<td>m</td>
<td>M</td>
<td>CR (013)</td>
<td>Note 1</td>
</tr>
<tr>
<td>51</td>
<td>&lt;</td>
<td></td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>52</td>
<td>&gt;</td>
<td></td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>53</td>
<td>?</td>
<td></td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>54 Shift</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>55</td>
<td>*</td>
<td>(Note 2)</td>
<td>(Note 1)</td>
<td>-1</td>
</tr>
<tr>
<td>56 Alt</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>57</td>
<td>SP</td>
<td>SP</td>
<td>SP</td>
<td>SP</td>
</tr>
<tr>
<td>58</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Caps Lock</th>
<th>Nul (Note 1)</th>
<th>Nul (Note 1)</th>
<th>Nul (Note 1)</th>
<th>Nul (Note 1)</th>
</tr>
</thead>
</table>

Character Codes (Part 2 of 3)

2-12 Keyboard Encoding
Character Codes (Part 3 of 3)

Keys 71 to 83 have meaning only in base case, in Num Lock (or shifted) states, or in Ctrl state. It should be noted that the shift key temporarily reverses the current Num Lock state.

---

Keyboard Encoding 2-13
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.

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

Keyboard Extended Functions

2-14 Keyboard Encoding
Shift States

Most shift states are handled within the keyboard routine, transparent to the system or application program. In any case, the current set of active shift states are available by calling an entry point in the ROM keyboard routine. The following keys result in altered shift states:

Shift

This key temporarily shifts keys 2-13, 15-27, 30-41, 43-53, 55, and 59-68 to upper case (base case if in Caps Lock state). Also, the Shift key temporarily reverses the Num Lock or non-Num-Lock state of keys 71-73, 75, 77, and 79-83.

Ctrl

This key temporarily shifts keys 3, 7, 12, 14, 16-28, 30-38, 43-50, 55, 59-71, 73, 75, 77, 79, and 81 to the Ctrl state. Also, the Ctrl key is used with the Alt and Del keys to cause the "system reset" function, with the Scroll Lock key to cause the "break" function, and with the Num Lock key to cause the "pause" function. The system reset, break, and pause functions are described in "Special Handling" on the following pages.

Alt

This key temporarily shifts keys 2-13, 16-25, 30-38, 44-50, and 59-68 to the Alt state. Also, the Alt key is used with the Ctrl and Del keys to cause the "system reset" function described in "Special Handling" on the following pages.

The Alt key has another use. This key allows the user to enter any character code from 0 to 255 into the system from the keyboard. The user holds down the Alt key and types the decimal value of the characters desired using the numeric keypad (keys 71-73, 75-77, and 79-82). The Alt key is then released. If more than three digits are typed, a modulo-256 result is created. These three digits are interpreted as a character code and are transmitted through the keyboard routine to the system or application program. Alt is handled internal to the keyboard routine.
Caps Lock

This key shifts keys 16-25, 30-38, and 44-50 to upper case. A second depression of the Caps Lock key reverses the action. Caps Lock is handled internal to the keyboard routine.

Scroll Lock

This key is interpreted by appropriate application programs as indicating use of the cursor-control keys should cause windowing over the text rather than cursor movement. A second depression of the Scroll Lock key reverses the action. The keyboard routine simply records the current shift state of the Scroll Lock key. It is the responsibility of the system or application program to perform the function.

Shift Key Priorities and Combinations

If combinations of the Alt, Ctrl, and Shift keys are pressed and only one is valid, the precedence is as follows: the Alt key is first, the Ctrl key is second, and the Shift key is third. The only valid combination is Alt and Ctrl, which is used in the "system reset" function.

Special Handling

System Reset

The combination of the Alt, Ctrl, and Del keys will result in the keyboard routine initiating the equivalent of a "system reset" or "reboot." System reset is handled internal to the keyboard.

Break

The combination of the Ctrl and Break keys will result in the keyboard routine signaling interrupt hex 1A. Also, the extended characters (AL = hex 00, AH = hex 00) will be returned.
Pause

The combination of the Ctrl and Num Lock keys will cause the keyboard interrupt routine to loop, waiting for any key except the Num Lock key to be pressed. This provides a system- or application-transparent method of temporarily suspending list, print, and so on, and then resuming the operation. The "unpause" key is thrown away. Pause is handled internal to the keyboard routine.

Print Screen

The combination of the Shift and PrtSc (key 55) keys will result in an interrupt invoking the print screen routine. This routine works in the alphanumeric graphics mode, with unrecognizable characters printing as blanks.

Other Characteristics

The keyboard routine does its own buffering. The keyboard buffer is large enough to support a fast typist. However, if a key is entered when the buffer is full, the key will be ignored and the "bell" will be sounded.

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

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

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

**Keyboard - Commonly Used Functions (Part 1 of 2)**

2-18  Keyboard Encoding
<table>
<thead>
<tr>
<th>Function</th>
<th>Key(s)</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exit insert mode</td>
<td>Ins</td>
<td>Line editor</td>
</tr>
<tr>
<td>Cancel current line</td>
<td>Esc</td>
<td>Command entry, text entry</td>
</tr>
<tr>
<td>Suspend system (pause)</td>
<td>Ctrl + Num Lock</td>
<td>Stop list, stop program, and so on Resumes on any key</td>
</tr>
<tr>
<td>Break interrupt</td>
<td>Ctrl Break</td>
<td>Interrupt current process</td>
</tr>
<tr>
<td>System reset</td>
<td>Alt Ctrl + Del</td>
<td>Reboot</td>
</tr>
<tr>
<td>Top of document and home cursor</td>
<td>Ctrl PgUp</td>
<td>Editors, word processors</td>
</tr>
<tr>
<td>Standard function keys</td>
<td>F1-F10</td>
<td>Primary function keys</td>
</tr>
<tr>
<td>Secondary function keys</td>
<td>Shift F1-F10, Ctrl F1-F10, Alt F1-F10</td>
<td>Extra function keys if 10 are not sufficient</td>
</tr>
<tr>
<td>Extra function keys</td>
<td>Alt Keys 2-13 (1-9,0,-,=)</td>
<td>Used when templates are put along top of keyboard</td>
</tr>
<tr>
<td>Extra function keys</td>
<td>Alt A-Z</td>
<td>Used when function starts with same letter as one of the alpha keys</td>
</tr>
</tbody>
</table>

Keyboard - Commonly Used Functions (Part 2 of 2)
### Function
<table>
<thead>
<tr>
<th>Function</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carriage return</td>
<td>Ctrl ←</td>
</tr>
<tr>
<td>Line feed</td>
<td></td>
</tr>
<tr>
<td>Bell</td>
<td>Ctrl G</td>
</tr>
<tr>
<td>Home</td>
<td>Home</td>
</tr>
<tr>
<td>Cursor up</td>
<td>↑</td>
</tr>
<tr>
<td>Cursor down</td>
<td>↓</td>
</tr>
<tr>
<td>Cursor left</td>
<td>←</td>
</tr>
<tr>
<td>Cursor right</td>
<td>→</td>
</tr>
<tr>
<td>Advance one word</td>
<td>Ctrl →</td>
</tr>
<tr>
<td>Reverse one word</td>
<td>Ctrl ←</td>
</tr>
<tr>
<td>Insert</td>
<td>Ins</td>
</tr>
<tr>
<td>Delete</td>
<td>Del</td>
</tr>
<tr>
<td>Clear screen</td>
<td>Ctrl Home</td>
</tr>
<tr>
<td>Freeze output</td>
<td>Ctrl Num Lock</td>
</tr>
<tr>
<td>Tab advance</td>
<td>→</td>
</tr>
<tr>
<td>Stop execution (break)</td>
<td>Ctrl Break</td>
</tr>
<tr>
<td>Delete current line</td>
<td>Esc</td>
</tr>
<tr>
<td>Delete to end of line</td>
<td>Ctrl End</td>
</tr>
<tr>
<td>Position cursor to end of line</td>
<td>End</td>
</tr>
</tbody>
</table>

**DOS Special Functions**

<table>
<thead>
<tr>
<th>Function</th>
<th>Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>Suspend</td>
<td>Ctrl Num Lock</td>
</tr>
<tr>
<td>Echo to printer</td>
<td>Ctrl PrtSc</td>
</tr>
<tr>
<td>Stop echo to printer</td>
<td>Ctrl PrtSc</td>
</tr>
<tr>
<td>Exit current function (break)</td>
<td>Ctrl Break</td>
</tr>
<tr>
<td>Backspace</td>
<td>← Key 14</td>
</tr>
<tr>
<td>Line feed</td>
<td>Ctrl ←</td>
</tr>
<tr>
<td>Cancel line</td>
<td>Esc</td>
</tr>
<tr>
<td>Copy character</td>
<td>F1 or ←</td>
</tr>
<tr>
<td>Copy until match</td>
<td>F2</td>
</tr>
<tr>
<td>Copy remaining</td>
<td>F3</td>
</tr>
<tr>
<td>Skip character</td>
<td>Del</td>
</tr>
<tr>
<td>Skip until match</td>
<td>F4</td>
</tr>
<tr>
<td>Enter insert mode</td>
<td>Ins</td>
</tr>
<tr>
<td>Exit insert mode</td>
<td>Ins</td>
</tr>
<tr>
<td>Make new line the template</td>
<td>F5</td>
</tr>
<tr>
<td>String separator in REPLACE</td>
<td>F6</td>
</tr>
<tr>
<td>End of file in keyboard input</td>
<td>F6</td>
</tr>
</tbody>
</table>

**BASIC Screen Editor Special Functions**

2-20  Keyboard Encoding
# APPENDIX A: ROM BIOS LISTINGS

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

| Fixed Disk ROM BIOS                                   |      |             |
| Fixed Disk I/O Interface                              | A-84 | 1           |
| Boot Strap Loader                                     | A-89 | 399         |
$TITLE(BIOS FOR The IBM PERSONAL COMPUTER XT)

11 EQUATES
12
0040 PORT_A EQU 60H | 0255 PORT A ADDR
0041 PORT_B EQU 61H | 0255 PORT B ADDR
0042 PORT_C EQU 62H | 0255 PORT C ADDR
0043 COM1_PORT EQU 63H
0044 INTA00 EQU 20H | 0259 PORT
0045 INTA01 EQU 21H | 0259 PORT
0046 EOI EQU 20H
0047 TIMER EQU 40H | 0253 TIMER CONTROL PORT ADDR
0048 TIOCD EQU 41H | 0253 TIMER/COUNTER 0 PORT ADDR
0049 TKINT EQU 01H | 0253 TIMER/INTR RECORD MASK
0050 DMA0B EQU 00H | 0253 DMA STATUS REG PORT ADDR
0051 DMA EQU 00H | 0253 DMA CH.0 ADDR. REG PORT ADDR
0052 MAX_PERIOD EQU 540H
0053 MINU_PERIOD EQU 410H
0054 KBD_IN EQU 60H | KEYBOARD DATA IN ADDR PORT
0055 KBD_INT EQU 02 | KEYBOARD INTERRUPT MASK
0056 KBD_DATA EQU 60H | KEYBOARD SCAN CODE PORT
0057 KR_CTL EQU 61H | CONTROL BITS FOR KEYBOARD SENSE DATA

35
36 | 35 8080 INTERRUPT LOCATIONS
37
38
39 ABS0 SEGMENT AT 0
0058 ORG 24H
0059 HMI_PTR LABEL WORD
0060 ORG 54H
0061 INTS_PTR LABEL WORD
0062 ORG 04H
0063 INT_ADDR LABEL WORD
0064 ORG 1004H
0065 INT_PTR LABEL DWORD
0066 ORG 1004H
0067 VIDEO_INT LABEL WORD
0068 ORG 1004H
0069 PARM_PTR LABEL DWORD | POINTERS TO VIDEO PARTS
006A ORG 1000H
006B BASIC_PTR LABEL WORD | ENTRY POINT FOR CASSETTE BASIC
006C ORG 01EH4H | INTERRUPT 1EH
006D DSK_POINTER LABEL DWORD | LOCATION OF POINTERS
006E ORG 01FH4H
006F EXT_PTR LABEL DWORD | POINTER TO EXTENSION
0070 ORG 400H
0071 DATA_AREA LABEL BYTE | ABSOLUTE LOCATION OF DATA SEGMENT
0072 DATA_ADDR LABEL WORD
0073 DATA_ADDR LABEL WORD
0074 ORG 0500H
0075 HSB_TEXT_PTR LABEL FAR
0076 ORG 7CDH
0077 BOOT_LOC0H LABEL FAR
0078 ABS0 END

67
68 | 68 STACK -- USED DURING INITIALIZATION ONLY
69
70
71 STACK SEGMENT AT 30H
007C 128
72 DW 128 DUP(?)

73 TOS LABEL WORD
74 STACK ENDS

77 | 77 1ROM BIOS DATA AREAS
LOC OBJ

DATA SEGMENT AT 40H

RS32E Daten | 4 DUP(?) | ADDRESSES OF RS32E ADAPTERS

PRINTD_BASE | 4 DUP(?) | ADDRESSES OF PRINTERS

EQUIP_FLAG | 4 | Installed HARDWARE

MS_TST | 4 | INITIALIZATION FLAG

MEMORY_SIZE | 4 | MEMORY SIZE IN K BYTES

MS_ERR_FLAG | 4 | SCRATCHPAD FOR MANUFACTURING

DB | 4 | ERROR CODES

### KEYBOARD DATA AREAS

KB_FLAG | DB | 

--- SHIFT FLAG ENCODES WITHIN KB_FLAG

INSSTATE | EQU | 00H | INSERT STATE IS ACTIVE

CPSSTATE | EQU | 04H | CAPS LOCK STATE HAS BEEN TOGGLED

NMSTATE | EQU | 08H | NUM LOCK STATE HAS BEEN TOGGLED

SCROLLSTATE | EQU | 09H | SCROLL LOCK STATE HAS BEEN TOGGLED

ALT_SHIFT | EQU | 0B0H | ALTERNATE SHIFT KEY DEPRESSED

CTRL_SHIFT | EQU | 0B4H | CONTROL SHIFT KEY DEPRESSED

LEFT_SHIFT | EQU | 0B2H | LEFT SHIFT KEY DEPRESSED

RIGHT_SHIFT | EQU | 0B1H | RIGHT SHIFT KEY DEPRESSED

KB_FLAG | DB | SECOND BYTE OF KEYBOARD STATUS

INS_SHIFT | EQU | 00H | INSERT KEY IS DEPRESSED

CAPS_SHIFT | EQU | 04H | CAPS LOCK KEY IS DEPRESSED

NUM_SHIFT | EQU | 08H | NUM LOCK KEY IS DEPRESSED

SCROLL_SHIFT | EQU | 09H | SCROLL LOCK KEY IS DEPRESSED

HOLD_STATE | EQU | 00H | SUSPEND KEY HAS BEEN Toggled

ALT_INPUT | DB | STORAGE FOR ALTERNATE KEYPAD ENTRY

BUFFER_HEAD | DB | POINTER TO HEAD OF KEYBOARD BUFFER

BUFFER_TAIL | DB | POINTER TO TAIL OF KEYBOARD BUFFER

KB_BUFFER | DB | 16 DUP(?) | ROOM FOR 16 ENTRIES

KB_BUFFER_END | Label | WORD

--- HEAD = TAIL INDICATES THAT THE BUFFER IS EMPTY

NUM_KEY | EQU | 69 | SCAN CODE FOR NUMBER LOCK

SCROLL_KEY | EQU | 7B | SCAN LOCK KEY

ALT_KEY | EQU | 86 | ALTERNATE SHIFT KEY SCAN CODE

CTRL_KEY | EQU | 29 | SCAN CODE FOR CONTROL KEY

CAPS_KEY | EQU | 56 | SCAN CODE FOR SHIFT LOCK

LEFT_KEY | EQU | 42 | SCAN CODE FOR LEFT SHIFT

RIGHT_KEY | EQU | 54 | SCAN CODE FOR RIGHT SHIFT

INS_KEY | EQU | 02 | SCAN CODE FOR INSERT KEY

DEL_KEY | EQU | 03 | SCAN CODE FOR DELETE KEY

--- DISKETTE DATA AREAS

SEEK_STATUS | DB | DRIVE RECALIBRATION STATUS

BIT | 3-8 = DRIVE 3-8 NEEDS RECAL

BIT | BEORE NEXT SEEK IF BIT IS 0

INT_FLAG | EQU | 00H | INTERRUPT OCCURRENCE FLAG

MOTOR_STATUS | DB | MOTOR STATUS

BIT | 3-6 = DRIVE 3-6 IS CURRENTLY

BIT | BUSY

BIT | CURRENT OPERATION IS A WRITE

REQUIRES DELAY

MOTOR_COUNT | DB | TIME OUT COUNTER FOR DRIVE TURN OFF

MOTOR_REM | DB | 2 SECS OF COUNTS FOR MOTOR TURN OFF
0041 ?? 169 DISKETTE_STATUS DB ? | RETURN CODE STATUS BYTE
0040 150 TIME_OUT EQU $00H | ATTACHMENT FAILED TO RESPO
003E 151 BAD_SEEK EQU $40H | SEEK OPERATION FAILED
0032 152 BAD_NEC EQU $2OH | NEC CONTROLLER HAS FAILED
001A 153 BAD_CRC EQU $10H | BAD CRC ON DISKETTE READ
0009 154 DMH_BOUNDARY EQU $09H | ATTACHMENT TO DMA ACROSS 64K BOUNDARY
0008 155 BAD_DMA EQU $08H | DMA OVERRUN ON OPERATION
0006 156 RECOUNT_MTB EQU $06H | 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
0000 160 HNCH_STATUS DB ? DUP(?) | STATUS BYTES FROM HNC

0049 ?? 162 STACK $0000H
0048 ?? 163 VIDEO DISPLAY DATA AREA
0047 ?? 164 CRT_MODE DB ? | CURRENT CRT MODE
0046 ?? 165 CRT_COLS DW ? | NUMBER OF COLUMNS ON SCREEN
0045 ?? 166 CRT_LINES DW ? | LENGTH OF REGN IN BYTES
0044 ?? 167 CRT_START DW ? | STARTING ADDRESS IN REGN BUFFER
0043 (0) 168 CURSOR_POSN DW $80H | CURSOR FOR EACH OF UP TO 8 PAGES

0042 ?? 169 CURSOR_MODE DW ? | CURRENT CURSOR MODE SETTING
0041 ?? 172 ACTIVE_PAGE DB ? | CURRENT PAGE BEING DISPLAYED
0040 ?? 173 ADDR_64K DW ? | BASE ADDRESS FOR ACTIV DISPLAY CARD
003F ?? 174 CRT_MODE_SET DB ? | CURRENT SETTING OF THE 3XS REGISTER
003E ?? 175 CRT_PALETTE DB ? | CURRENT PALETTE SETTING COLOR CARD

0047 ?? 176 POST DATA AREA
0046 ?? 177 TIMER DATA AREA
0045 ?? 178 SYSTEM DATA AREA

0041 ?? 190 BIOS_BREAK DB ? | BIT 7;1 IF BREAK KEY HAS BEEN HIT
0040 ?? 191 RESET_FLAG DB ? | WORD=1234H IF KEYBOARD RESET UNDERWAY
003F ?? 192 FIXED_DISK_DATA AREAS
003E ?? 193 PRINT,_TIME_OUT DB 4 DUP(?)
003D ?? 194 DATA ENDS
003C ?? 195 ADDITIONAL KEYBOARD DATA AREA

003A ?? 208 DATA ENDS
0039 ?? 209 EXTRA DATA AREA

A-4 System BIOS
----- 219 XDATA SEGMENT AT 0BH
0000 ?? 220 STATUS_BYTE DB ??
----- 211 XDATA ENDS
222 --------------------
223 | VIDEO DISPLAY BUFFER |
224 |-----------------------|
----- 225 VIDEO_RAM SEGMENT AT DB00H
0000 226 RESIDN LABEL BYTE
0000 227 RESIDN LABEL WORD
0000 (16504) 228 DB 16504 DUP(?)
?? ??

) 219 VIDEO_RAM ENDS
230 -------------------------------------
231 | ROM RESIDENT CODE |
232 |-------------------|
----- 233 CODE SEGMENT AT 0F000H
0000 (57344) 234 DB 57344 DUP(?)
????

) 235
E000 313503135132 236 DB '1501512 COMM. 1961' | COPYRIGHT NOTICE
94942020319950
32

237
230
239
240 | INITIAL RELIABILITY TESTS -- PHASE 1 |
241 -------------------------------------
242 | ASSUME CS:CODE,DS:CODE,ES:ABS0,DS:DATA|
243 | DATA DEFINITIONS |
244 |------------------|
245 |------------------|
246 |------------------|
247 |------------------|
248 E016 D7E0 249 C1 DN C11 | RETURN ADDRESS
E018 7E11 250 C2 DN C26 | RETURN ADDRESS FOR CP510 STACK
251 E01A 204842404F4B 252 F3B DB ' KB OK!'13 | KB FOR MEMORY SIZE
E020 0D
253
254 |------------------|
255 | LOAD A BLOCK OF TEST CODE THROUGH THE KEYBOARD PORT |
256 | FOR MANUFACTURING TESTING |
257 | THIS ROUTINE WILL LOAD A TEST (MAX LENGTH=FAPFH) THROUGH |
258 | THE KEYBOARD PORT. CODE WILL BE LOADED AT LOCATION |
259 | 0000:0580. AFTER LOADING, CONTROL WILL BE TRANSFERRED |
260 | TO LOCATION 0000:5500. STACK WILL BE LOCATED JUST BELOW |
261 | THE TEST CODE. THIS ROUTINE ASSUMES THAT THE FIRST 2 |
262 | BYTES TRANSFERRED CONTAIN THE COUNT OF BYTES TO BE LOADED |
263 | (BYTE = COUNT LOW, BYTE = COUNT HI.) |
264 |------------------|
265 |------------------|
266 | FIRST, GET THE COUNT |
267 E021
268 |---------------|
E021 E0131A 269 CALL SP_TEST | GET COUNT LOW
E022 0AFB 270 MOV BH,DL | SAVE IT
E026 E021JA 271 CALL SP_TEST | GET COUNT HI
E029 5AEB 272 MOV CH,BL |
E02B BACF 273 MOV CL,BH | CK NOW HAS COUNT
E02C FC 274 CLD | SET DIR. FLAG TO INCREMENT
E02E FA 275 CLS |
E02F BF0008 276 MOV DI,0500H | SET TARGET OFFSET (+000008)
E030 8B0D 277 MOV AL,0DH | UNMASK K/B INTERRUPT
E034 E621 278 OUT INT41,AL |
E035 E60A 279 MOV AL,0AH |
E033 E620 280 OUT INT41,AL |
E03A 8A10D 281 MOV DX,63H | SET UP PORT B ADDRESS
E03D BCC4C 282 MOV BX,444CH | CONTROL BITS FOR PORT B
E043 D402 283 MOV AH,02H | K/B REQUEST PENDING MASK
E047 TST: 284
E042 0AC3 285 MOV AL,BL |
E046 EE 286 OUT DX,AL | TOGGLE K/B CLOCK
A-6 System BIOS
LOC OBJ

364  365  366  367  368
369  370  371  372  373
374  375  376  377  378
379  380  381  382  383
384  385  386  387  388
390  391  392  393  394
395  396  397  398  399
400  401  402  403  404
405  406  407  408  409
410  411  412  413  414
415  416  417  418  419
420  421  422  423  424
425  426  427  428  429
430  431  432  433  434
436  437  438  439  440

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>DESCRIPTION</th>
<th>DESCRIPTION</th>
<th>DESCRIPTION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>A CHECKSUM IS DONE FOR THE 8K</td>
<td>A CHECKSUM IS DONE FOR THE 8K</td>
<td>A CHECKSUM IS DONE FOR THE 8K</td>
<td>A CHECKSUM IS DONE FOR THE 8K</td>
<td>A CHECKSUM IS DONE FOR THE 8K</td>
</tr>
<tr>
<td>LOC OBJ</td>
<td>LOC OBJ</td>
<td>LOC OBJ</td>
<td>LOC OBJ</td>
<td>LOC OBJ</td>
</tr>
</tbody>
</table>

| EDAE | EDAE | EDAE | EDAE | EDAE |
| C10: | C10: | C10: | C10: | C10: |
| OUT 0AH,AL | OUT 09H,AL | MOV DX,320H | OUT DX,AL | MOV DL,08H |
| | | | | |

| EDB2 | EDB2 | EDB2 | EDB2 | EDB2 |
| B400 | B403 | B403 | B403 | B403 |
| MOV DX,320H | OUT DX,AL | MOV AL,08H | MOV AL,09H | MOV AL,01010101B |
| | | | | |

| EDFA | EDFA | EDFA | EDFA | EDFA |
| FE0 | FE0 | FE0 | FE0 | FE0 |
| INC AL | MOV DL,08H | MOV AL,09H | MOV AL,PORT,AL | MOV AL,10101010B |
| | | | | |

| EDCA | EDCA | EDCA | EDCA | EDCA |
| C10: | C10: | C10: | C10: | C10: |
| OUT PORT_B,AL | OUT PORT_A,AL | MOV AL,08H | MOV AX,CX | MOV DS,AX |
| | | | | |

| EDCC | EDCC | EDCC | EDCC | EDCC |
| C10: | C10: | C10: | C10: | C10: |
| MOV AL,09H | MOV AL,09H | MOV AX,CX | MOV SS,AX | MOV DS,AX |
| | | | | |

| EDCF | EDCF | EDCF | EDCF | EDCF |
| CLD | ASSUME SS:CODE | MOV BX,000000H | MOV SP,OFFSET CI | JMP BIOS_CHECKSUM |
| | | | | |

| EEPROM | EEPROM | EEPROM | EEPROM | EEPROM |
| D002 | D002 | D002 | D002 | D002 |
| MOV AL,02H | MOV AL,02H | MOV AL,04H | MOV AL,04H | MOV AL,04H |
| | | | | |

| EEPROM | EEPROM | EEPROM | EEPROM | EEPROM |
| 426 | 426 | 426 | 426 | 426 |
| MOV AL,04H | MOV AL,04H | MOV CL,0 | MOV BL,AL | MOV BYTE PTR ES:[DI],AL |
| | | | | |

| EEPF | EEPF | EEPF | EEPF | EEPF |
| 408 | 408 | 408 | 408 | 408 |
| MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV CL,0 | MOV CL,0 |
| | | | | |

| EEPF | EEPF | EEPF | EEPF | EEPF |
| 424 | 424 | 424 | 424 | 424 |
| MOV CL,0 | MOV CL,0 | MOV CL,0 | MOV CL,0 | MOV CL,0 |
| | | | | |

| EEPROM | EEPROM | EEPROM | EEPROM | EEPROM |
| 407 | 407 | 407 | 407 | 407 |
| MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H |
| | | | | |

| EEPROM | EEPROM | EEPROM | EEPROM | EEPROM |
| 426 | 426 | 426 | 426 | 426 |
| MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H |
| | | | | |

| EEPROM | EEPROM | EEPROM | EEPROM | EEPROM |
| 436 | 436 | 436 | 436 | 436 |
| MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H |
| | | | | |

| EEPROM | EEPROM | EEPROM | EEPROM | EEPROM |
| 424 | 424 | 424 | 424 | 424 |
| MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H |
| | | | | |

| EEPROM | EEPROM | EEPROM | EEPROM | EEPROM |
| 429 | 429 | 429 | 429 | 429 |
| MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H |
| | | | | |

| EEPROM | EEPROM | EEPROM | EEPROM | EEPROM |
| 435 | 435 | 435 | 435 | 435 |
| MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H |
| | | | | |

| EEPROM | EEPROM | EEPROM | EEPROM | EEPROM |
| 436 | 436 | 436 | 436 | 436 |
| MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H |
| | | | | |

| EEPROM | EEPROM | EEPROM | EEPROM | EEPROM |
| 437 | 437 | 437 | 437 | 437 |
| MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H |
| | | | | |

| EEPROM | EEPROM | EEPROM | EEPROM | EEPROM |
| 438 | 438 | 438 | 438 | 438 |
| MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H |
| | | | | |

| EEPROM | EEPROM | EEPROM | EEPROM | EEPROM |
| 439 | 439 | 439 | 439 | 439 |
| MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H |
| | | | | |

| EEPROM | EEPROM | EEPROM | EEPROM | EEPROM |
| 440 | 440 | 440 | 440 | 440 |
| MOV AL,04H | MOV AL,04H | MOV AL,04H | MOV AL,04H | MOVE AX,0 |
| | | | | |

System BIOS  A-7
LOC OBJ  LINE  SOURCE

A-8 System BIOS
LOC OBJ   LINE   SOURCE

E16F 090020   318   MOV  CX,2000H           1 SET FOR 16K WORDS
E172 01FB3412   319   CMP  DX,1234H          1  MARY START!
E176 7416   320   JE   CLR_STS
E17B 8C1880   321   MOV  SP,OFFSET C2
E17E 7412   322   JNP  STGST_CNT
E180 090056   323   CMP  AL,0EH          1 STORAGE OK, DETERMINE SIZE
E182 0006   324   MOV  BL,AL          1 SAVE FAILING BIT PATTERN
E184 E660   325   MOV  AL,DL.OH          1 <<<<<CHECKPOINT 00H>>>>
E186 0B09   326   C24A: OUT  PORT.A,AL         1 BASE RAM FAILURE - HANG
E188 0E2E   327   SUB  CX,CX          1 FLIPPING BETWEEN 04 AND
E18A 0600   328   C24B: LOOP  C24G            1 FAILING BIT PATTERN
E18C 0F06   329   XOR  BL,AL          1 FAILING BIT PATTERN
E18E 0C0   330   JMP  C24A
E190 0C0   331   CLR_STS:
E192 0C0   332   SUB  AX,AX          1 MAKE AX=0000
E194 0F3   333   REP  STOSW          1 STORE 8K WORDS OF 0000
E196 0F3   334   MOV  DATA_WORDSOFFSET_RESET_FLAG,0X     1 RESTORE RESET FLAG
E196 0A0040   335   MOV  DX,0040H       1 SET POINTER TO JUST>16KB
E199 801000   336   MOV  BX,16         1 BASIC COUNT OF 16K
E19C 8EC2   337   FILL_LOOP:
E19E 2BF   338   MOV  ES,0X          1 SET SEG. REG.
E1A0 865AA   339   SUB  DI,0I          1 TEST PATTERN
E1A2 4BC0   340   MOV  AX,0AA5H      1 SAVE PATTERN
E1A5 260005   341   MOV  CX,AX          1 SAVE PATTERN
E1A8 00F   342   MOV  AL,0FH          1 SEND PATTERN TO REM.
E1AA 260005   343   MOV  AX,ES:DI.OX    1 PUT SOMETHING IN AL
E1AC 3DC1   344   XOR  AX,AC          1 COMPARE PATTERNS
E1AE 7515   345   JNZ  MOV_SEG_END   1 GO END IF NO COMPARE
E1B0 900020   346   MOV  CX,2000H       1 SET COUNT FOR 8K WORDS
E1B4 0F3   347   REP  STOSW          1 FILL 8K WORDS
E1B8 0B6   348   ADD  CX,40H         1 POINT TO NEXT 16KB BLOCK
E1C0 3C510   349   ADD  BX,16        1 BUMP COUNT BY 16KB
E1C0 00FED   350   CHS  DH,0AH       1 TOP OF RAM AREA YET? (00000)
E1C2 750A   351   JNZ  FILL_LOOP        1
E1C2 91E1304   352   MOV  DATA_WORDSOFFSET_MEMORY_SIZE,0X  1 SAVE MEMORY SIZE
E1C2 91E1304   353   MOV  DATA_WORDSOFFSET_MEMORY_SIZE,0X  1 SAVE MEMORY SIZE
E1C6 0B3000   354   MOV  AX,STACK       1 SET STACK VALUE
E1C8 3C00   355   MOV  SS,AX       1 SET THE STACK UP
E1C8 B0041   356   MOV  SP,OFFSET TOS      1 STACK IS READY TO GO
E1C8 013   357   MOV  AL,13H          1 ICHI - EDGE, SHNL, ICH4
E1C8 2620   358   OUT  INTAddr.0L      1
E1C8 0000   359   MOV  AL,8          1 SETUP ICME - INT TYPE 0 (6-F)
E1C8 681   360   MOV  OUT  INTAddr.0L      1
E1C8 600F   361   MOV  AL,9          1 SETUP ICNM - BUFFER,0006 MODE
E1C8 6021   362   OUT  INTAddr.0L      1
E1C8 80FF   363   MOV  AL,OFFH       1 MARK ALL INTS. OFF
E1C8 621   364   OUT  INTAddr.0L      1 (VIDEO ROUTINE ENABLES INTS.1
E1D0 313   365   MOV  AX,STACK       1 SETUP THE INTERRUPT CONTROLLER CHIP:
E1D0 3C00   366   MOV  SS,AX       1 SET THE STACK UP
E1D0 B0041   367   MOV  SP,OFFSET TOS      1 STACK IS READY TO GO
E1D0 6220   368   MOV  AL,8          1 SETUP ICME - INT TYPE 0 (6-F)
E1D2 0000   369   MOV  AL,9          1 SETUP ICNM - BUFFER,0006 MODE
E1D2 621   370   OUT  INTAddr.0L      1
E1D4 80FF   371   MOV  AL,OFFH       1 MARK ALL INTS. OFF
E1D4 621   372   OUT  INTAddr.0L      1 (VIDEO ROUTINE ENABLES INTS.1
E1D8 0F1E   373   MOV  AX,CS          1 SETUP THE INTERRUPT VECTORS TO TEMP INTERRUPT
E1D8 0F1E   374   MOV  AX,CS          1 SETUP THE INTERRUPT VECTORS TO TEMP INTERRUPT
E1E0 9D2000   375   MOV  CX,32       1 FILL ALL 32 INTERRUPTS
E1E2 28F    376   MOV  ES,DI          1 FIRST INTERRUPT LOCATION
E1E4 8C77   377   MOV  ES:DI          1 SET ES=0000 ALSO
E1E6 8033F   378   MOV  AX,OFFSET D11       1 MOVE ADDR OF INTR PROC TO TBL
E1E9 0B   379   STOSB
E1EA 0CC6   380   MOV  AX,CB          1 GET ADDR OF INTR PROC SEG
E1EB 407   381   STOSB
E1ED 2F7   382   LOOP  D3          1 VECTORS
E1F6 603F00   383   MOV  AX,OFFSET VECTOR_TABLE+16   1 START WITH VIDEO ENTRY
E1F6 603F00   384   MOV  AX,OFFSET VECTOR_TABLE+16   1 START WITH VIDEO ENTRY

System BIOS A-9
E1FA 891000 593 MOV CX,16
E1FD A5 594 MOV AX,0
E1FE 47 595 INC DI
E1FF 47 596 INC DI
E200 E9FB 597 LOOP D9A
E200 598

[----------]
E202 599 I DETERMINE CONFIGURATION AND MFG. MODE: I

[----------]
E204 600

E202 1F 601 POP BX
E205 1E 602 PUSH DS
E204 E462 603 MOV DS,AX
E206 20DF 604 MOV AL,AL.0000111B
E206 8A40 605 MOV AH,AL
E206 B0A0 606 MOV AL,10101010B
E20C E661 607 MOV PORT_B,AL
E20E 90 608 HOP
E20F E462 610 MOV AL,PORT_C
E211 8104 611 MOV CL,4
E213 2E00 612 MOV AL,CL
E215 2E00 613 AND AL,1111111DB
E217 04C0 614 OR AL,AL
E219 2A46 615 SUB AL,AH
E21B A31004 616 MOV DATA_WD[OFFSET EQUIP_FLAG1].AX I SAVE SWITCH INFO
E21E 80DF 617 MOV AL,00H
E220 E661 618 OUT CXD_PORT.AL
E222 E051B 619 CALL KBD_RESET
E225 808AA 620 CMP BL,0AH
E225 7418 621 JE 66
E22A 80868 622 CMP BL,06H
E22D 705C 623 JNE 6B
E22F 00FFD 624 JMP MFG_BOOT
E232 B036 625 MOV AL,08H
E234 8661 626 OUT PORT_B,AL
E236 90 627 NOP
E237 90 628 NOP
E238 E460 629 IN AL,PORT_A
E23A 20F7 630 AND AL,0BFH
E23C 7504 631 JNZ 66
E23E F0012004 632 MOV DATA_AREA[OFFSET MFG_TEST] I SET MANUFACTURING TEST FLAG
E242 633

[----------]
E242 634 I INITIALIZE AND START CRT CONTROLLER (6045) I
E244 635 I TEST VIDEO READ/WRITE STORAGE.
E246 636 I DESCRIPTION:
E248 637 I RESET THE VIDEO ENABLE SIGNAL.
E24A 638 I SELECT ALPHANUMERIC MODE, 40 X 25, B & W.
E24C 639 I READ/WRITE DATA PATTERNS TO STG. CHECK STG.
E24E 640 I ADDRESSEABLE.
E250 641 I ERROR; \# 1 LONG AND 2 SHORT BEEPS

[----------]
E242 642

E244 A11004 646 MOV AX,DATA_WD[OFFSET EQUIP_FLAG1] I SET SEND SWITCH INFO
E246 80 647 PUSH AX
E248 5B30 648 MOV AL,00H
E24A A31004 649 MOV DATA_WD[OFFSET EQUIP_FLAG1].AX
E24B 2A64 650 MOV AH,0AH
E24D CD10 651 INT 10H I SEND INIT TO B/W CARD
E24F 8020 652 MOV AL,00H
E251 A31004 653 MOV DATA_WD[OFFSET EQUIP_FLAG1].AX
E253 2A64 654 MOV AH,0AH
E256 CD10 655 INT 10H
E258 80 656 MOV AX,DATA_WD[OFFSET EQUIP_FLAG1].AX I RECOVER REAL SWITCH INFO
E25A 5B30 657 MOV AL,00H
E25C 70A4 658 AND AL,00H
E25E 5B30 659 MOV AX,DATA_WD[OFFSET EQUIP_FLAG1].AX
E260 8F0040 660 MOV DI,OFFSET VIDEO_INT
E262 7C50B0F7 661 MOV [DI],OFFSET DUMMY RETURN I RETURN IF NO VIDEO CARD
E264 E9AD00 662 JMP E18 I I Bypass Video Test
E266 643 E7: 663 MOV AX,DATA_WD[OFFSET EQUIP_FLAG1].AX I SAVE IT
E268 80 664 MOV AL,00H
E26A 7408 665 JE 66
E26C 4CA4 666 INC AH
E26E 8020 667 MOV AL,00H
E270 8020 668 MOV AL,00H
E272 7502 669 JMP E18 I I Bypass Video Test
E274 8040 66A MOV AH,0

A-10 System BIOS
LOC OBJ  

LINE SOURCE

E776 6AE0 670 68: XCHG AH,AL ; SET_MODE:
E77B 5A 671 69: PUSH AX ; SAVE VIDEO MODE ON STACK
E779 2AE4 672 70: SUB AH,AL ; INITIALIZE TO ALPHANUMERIC MD
E77B CD10 673 71: INT 10H ; CALL VIDEO_ID
E77D 58 674 72: POP AX ; RESTORE VIDEO SENSE SMS IN AN
E77E 59 675 73: PUSH AX ; SAVE VALUE
E77F 800000 676 74: MOV BX,08000H ; REG VIDEO RAM ADDR B/W CD
E802 BAB003 677 75: MOV DX,30FH ; MODE REG FOR B/W
E805 B90000 678 76: MOV CX,2048 ; RAM MODE CRT FOR B/W CD
E808 8001 679 77: MOV AL,1 ; SET MODE FOR B/W CARD
E86A 00FC30 680 78: CMP AL,30H ; B/W VIDEO CARD ATTACHED?
E86D 7409 681 79: JE E9 ; YES - GO TEST VIDEO STG
E86F D700 682 80: MOV BX,0000H ; REG VIDEO RAM ADDR COLOR CD
E891 BADD03 683 81: MOV DX,30FH ; MODE REG FOR COLOR CD
E894 0020 684 82: MOV CH,20H ; RAM MODE CRT FOR COLOR CD
E89F F0C8 685 83: DEC AL ; SET MODE TO 0 FOR COLOR CD
E89F EE 686 84: OUT DX,AL ; TEST_VIDEO_STG:
E89F A13F2C43412 687 85: CMP DATA_WORDOFFSET_RESET_FLAGS1,1234H ; POP INIT BY IOB RESET?
E89F 86C3 688 86: MOV ES,BX ; POINT ES TO VIDEO RAM STG
E89A 7407 689 87: JE E10 ; YES - SKIP VIDEO RAM TEST
E89A D2DB 690 88: MOV DS,BX ; POINT DS TO VIDEO RAM STG
E89A EEC703 691 89: ASSUME DS:NOTHING,ES:NOTHING
E89A 7546 692 90: CALL STGST_CNT ; GO TEST VIDEO B/W STG
E89B 7E46 693 91: JNE E17 ; R/W STG FAILURE - BEEP SHK

| 670 | SETUP_VIDEO_DATA_ON_SCREN_FOR_VIDEO | 704 |
| 671 | LINE_TEST. | 705 |
| 672 | ENABLE_VIDEO_SIGNAL_AND_SET_MODE. | 706 |
| 673 | DISPLAY_A_HORIZONTAL_BAR_ON_SCREEN. | 707 |

| 670 | SETUP_VIDEO_DATA_ON_SCREN_FOR_VIDEO | 704 |
| 671 | LINE_TEST. | 705 |
| 672 | ENABLE_VIDEO_SIGNAL_AND_SET_MODE. | 706 |
| 673 | DISPLAY_A_HORIZONTAL_BAR_ON_SCREEN. | 707 |

| 711 | SETUP.STARTING_LOC | 716 |
| 712 | NO. OF BLANKS TO DISPLAY | 717 |
| 713 | WRITE_VIDEO_STORAGE | 718 |

| 720 | Interface_LINES_Test, | 721 |
| 722 | SENSE_ON-OFF_TRANSITION_OF_THE | 723 |
| 724 | SYNCHRONIZATION_LINES. | 725 |

| 726 | GET_VIDEO_SENSE_SM_INFO | 727 |
| 728 | B/W CARD ATTACHED? | 729 |
| 730 | YES - GO TEST LINES | 731 |
| 732 | LINE_TEST. | 733 |
| 734 | O/F』LOOPOSENT | 735 |

| 735 | GET_VIDEO_SENSE_SM_INFO | 736 |
| 737 | READ_RTC_STATUS_PORT | 738 |
| 739 | CHECK_VIDEO/NOIRE_LINE | 740 |
| 741 | LOOP_TILL_ON OR TIMEOUT | 742 |
| 743 | GO_PRINT_ERROR_MSG | 744 |

Appendix A

System BIOS A-11
E22A E204 746  AND  AL,AH  ; CHECK VIDEO/HORZ LINE
E22C 7611 747  JZ  E16  ; IT'S ON - CHECK NEXT LINE
E22E E2F9 748  LOOP  E15  ; LOOP IF OFF TILL IT GOES ON
E2F9 749  E17:  ; CRT_ERR:
E2FD 755  POP  DS
E2FJ 751  PUSH DS
E2F2 C66150006 752  MOV DS:MSR_ERR_FLAG.OFH  ; <<<<<<CRT_ERR CHKPT, 06<<<<
E2FE B0D0 753  MOV DX,102H
E2FA E0816 754  CALL  ERR_BEEP  ; GO BEEP SPEAKER
E2FD E0B6 755  JMP  SHORT  E1B
E2FF 756  E16:  ; HOT_LINE:
E2FF B103 757  MOV  CL,3  ; GET NEXT BIT TO CHECK
E301 D2EC 758  SHR  AH,CL  ; GET NEXT BIT TO CHECK
E303 7507 759  JNZ  E12  ; GET ORDER NUMBER
E305 766  E16:  ; DISPLAY_CURSOR:
E309 50 760  POP  AX  ; GET VIDEO SENSE SMS (AH)
E30A A400 761  MOV  AH,0  ; GET MODE AND DISPLAY CURSOR
E30C CD10 762  INT  10H  ; CALL VIDEO I/O PROCEDURE
E30E 763  E1B_1:  ; MOV  DX,OC000H
E310 B0CC 764  MOV  DX,OC000H
E312 765  E1B:  ; MOV  DX,OC000H
E313 86DA 766  MOV  DS,DX
E315 767  SUB  BX,BX
E317 8007 768  MOV  AX,[BX+1]
E319 53 769  PUSH BX  ; GET FIRST 2 LOCATIONS
E31B 770  POP BX  ; GET FIRST 2 LOCATIONS
E31D 3555AA 771  CMP AX,0A55H  ; LET BUS SETTLE
E31E 772  JNZ  E1B0  ; PRESENT?
E31F 773  CALL  ROM_CHECK  ; NO GO LOOK FOR OTHER MODULES
E320 EBD4 774  JMP  SHORT  E1B3
E31F 775  E1B0:  ; ADD  DX,OC000H  ; POINT TO NEXT 2K BLOCK
E322 776  E1B1:  ; MOV  DX,OC000H
E323 777  E1C:  ; MOV  DX,OC000H
E324 BFA00C8 778  ADD [DX],CL  ; TOP OF VIDEO ROM AREA YET?
E327 779  EI8A  ; GO SCAN FOR ANOTHER MODULE
E327 780  :  ; IOSINTERRUPT CONTROLLER TEST
E329 781  CJNE  RX,090H  ; DESCRIPTION
E32F 782  E000  ; READ/WRITE THE INTERRUPT MASK REGISTER (IMR)  ; WITH ALL ONES AND ZEROS, ENABLE SYSTEM
E331 6461 783  OUT INTAD1AL  ; INTERRUPTS, MASK DEVICE INTERRUPTS OFF, CHECK  ; FOR NOT INTERRUPTS (UNEXPECTED)
E333 6462 784  IN AL,INTAD1AL
E335 6460 785  OR AL,AL  ; FOR NOT INTERRUPTS (UNEXPECTED)
E337 7518 786  JNZ D6  ; FOR NOT INTERRUPTS (UNEXPECTED)
E339 60FF 787  MOV AL,OFFH  ; DISMANAGE DEVICE INTERRUPTS
E33B 6A01 788  OUT INTAD1AL  ; WRITE TO IMR
E33D 9421 789  JZ  D6  ; READ IMR
E33F 0401 790  ADD AL,1  ; READ IMR
E341 791  D6  ; ALL IMR BIT ON?
E343 792  JNZ D6  ; NO - GO TO ERR ROUTINE
E345 793  :  ; CHECK FOR NOT INTERRUPTS
E347 794  E1B:  ; INTERRUPTS ARE MASKED OFF. CHECK THAT NO INTERRUPTS OCCUR.
E349 795  E000  ; MOV  DATA_AREA[OFFSET INTAD_FLAG].OH
E34F 796  MOVE  AL,0  ; SET IMR TO ZERO
E351 6461 797  OUT INTAD1AL  ; READ IMR
E353 6A40 798  JZ  D6  ; DAMAGED DEVICE INTERRUPTS
E355 7518 799  JNO D6  ; READ IMR
E357 60FF 800  MOV AL,OFFH  ; DISABLE DEVICE INTERRUPTS
E359 6A01 801  OUT INTAD1AL  ; WRITE TO IMR
E35B 9421 802  JZ  D6  ; READ IMR
E35D 0401 803  ADD AL,1  ; READ IMR
E35F 0401 804  D6  ; ALL IMR BIT ON?
E345 E1B:  ; NO - GO TO ERR ROUTINE
E347 795  E000  ; MOV  DATA_AREA[OFFSET INTAD_FLAG].AL
E34F 796  MOVE  AL,0  ; CLEAR INTERRUPT FLAG
E347 6F8 806  MOV  DS,DX
E34F 807  D5:  ; LOOP D4  ; MIGHT OCCUR
E348 92E 808  LOOP D5  ; MIGHT OCCUR
E34B 809  LOOP D5  ; MIGHT OCCUR
E34D 80A 80A  CMP DATA_AREA[OFFSET INTAD_FLAG].OH  ; DID ANY INTERRUPTS OCCUR?
E34F 80B 80B  JZ  07  ; NO - GO TO NEXT TEST
E351 793 793
E353 61FF8000 80C  MOV  DX,OFFSET EO  ; DISPLAY 101 ERROR

A-12  System BIOS
A-14  System BIOS
<table>
<thead>
<tr>
<th>LOC OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>E443 EC</td>
<td>977</td>
<td>IN</td>
</tr>
<tr>
<td>E444 3AC3</td>
<td>978</td>
<td>CMP</td>
</tr>
<tr>
<td>E446 7510</td>
<td>979</td>
<td>JNE</td>
</tr>
<tr>
<td>E448 4A</td>
<td>980</td>
<td>DEC</td>
</tr>
<tr>
<td>E449 D1E3</td>
<td>981</td>
<td>SHL</td>
</tr>
<tr>
<td>E44B E2E6</td>
<td>982</td>
<td>LOOP</td>
</tr>
<tr>
<td></td>
<td>983</td>
<td>i------</td>
</tr>
<tr>
<td>E450 B9000</td>
<td>984</td>
<td>MOV</td>
</tr>
<tr>
<td>E450 B001</td>
<td>985</td>
<td>MOV</td>
</tr>
<tr>
<td>E452 4A</td>
<td>986</td>
<td>DEC</td>
</tr>
<tr>
<td>E453</td>
<td>987</td>
<td>EXP_EER:</td>
</tr>
<tr>
<td>E453 8AE0</td>
<td>988</td>
<td>MOV</td>
</tr>
<tr>
<td>E455 EE</td>
<td>989</td>
<td>OUT</td>
</tr>
<tr>
<td>E456 0091</td>
<td>990</td>
<td>MOV</td>
</tr>
<tr>
<td>E458 4C</td>
<td>991</td>
<td>IN</td>
</tr>
<tr>
<td>E459 3AC4</td>
<td>992</td>
<td>CMP</td>
</tr>
<tr>
<td>E45B 7506</td>
<td>993</td>
<td>JNE</td>
</tr>
<tr>
<td>E45D 40E0</td>
<td>994</td>
<td>SHL</td>
</tr>
<tr>
<td>E45F 8F2</td>
<td>995</td>
<td>LOOP</td>
</tr>
<tr>
<td></td>
<td>996</td>
<td>JMP</td>
</tr>
<tr>
<td>E460</td>
<td>997</td>
<td>MOV</td>
</tr>
<tr>
<td>E467 103F15</td>
<td>998</td>
<td>CALL</td>
</tr>
<tr>
<td></td>
<td>999</td>
<td>i------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1002</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1003</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1004</td>
</tr>
<tr>
<td>E464</td>
<td>1005</td>
<td>E19:</td>
</tr>
<tr>
<td>E464 E8EC15</td>
<td>1006</td>
<td>PUSH</td>
</tr>
<tr>
<td>E464 2E</td>
<td>1007</td>
<td>E20:</td>
</tr>
<tr>
<td>E464 B3E72003412</td>
<td>1008</td>
<td>JMP</td>
</tr>
<tr>
<td>E464 7503</td>
<td>1009</td>
<td>JMP</td>
</tr>
<tr>
<td>E465 E9F00</td>
<td>1010</td>
<td>E20A:</td>
</tr>
<tr>
<td>E467 80100</td>
<td>1011</td>
<td>JMP</td>
</tr>
<tr>
<td>E46C 8B</td>
<td>1012</td>
<td>E20B:</td>
</tr>
<tr>
<td>E46E 808010</td>
<td>1013</td>
<td>SUB</td>
</tr>
<tr>
<td>E46F 8014</td>
<td>1014</td>
<td>MOV</td>
</tr>
<tr>
<td>E46F 83EB</td>
<td>1015</td>
<td>SHR</td>
</tr>
<tr>
<td>E46F 86CB</td>
<td>1016</td>
<td>MOV</td>
</tr>
<tr>
<td>E46F B08004</td>
<td>1017</td>
<td>MOV</td>
</tr>
<tr>
<td>E471</td>
<td>1010</td>
<td>E21:</td>
</tr>
<tr>
<td>E472 80EC3</td>
<td>1019</td>
<td>MOV</td>
</tr>
<tr>
<td>E473 81430004</td>
<td>1020</td>
<td>ADD</td>
</tr>
<tr>
<td>E473 80</td>
<td>1021</td>
<td>PUSH</td>
</tr>
<tr>
<td>E474 81</td>
<td>1022</td>
<td>PUSH</td>
</tr>
<tr>
<td>E475 85</td>
<td>1023</td>
<td>PUSH</td>
</tr>
<tr>
<td>E476 83</td>
<td>1024</td>
<td>PUSH</td>
</tr>
<tr>
<td>E477 80</td>
<td>1025</td>
<td>MOV</td>
</tr>
<tr>
<td>E478 808002</td>
<td>1026</td>
<td>CALL</td>
</tr>
<tr>
<td>E479 8E8F01</td>
<td>1027</td>
<td>JNZ</td>
</tr>
<tr>
<td>E47A 754C</td>
<td>1028</td>
<td>POP</td>
</tr>
<tr>
<td>E47B 80100</td>
<td>1029</td>
<td>ADD</td>
</tr>
<tr>
<td>E47C 80</td>
<td>1030</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E47D 80</td>
<td>1031</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E47E 80</td>
<td>1032</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E47F 80</td>
<td>1033</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E480 80</td>
<td>1034</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E481 80</td>
<td>1035</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E482 80</td>
<td>1036</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E483 80</td>
<td>1037</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E484 80</td>
<td>1038</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E485 80</td>
<td>1039</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E486 80</td>
<td>1040</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E487 80</td>
<td>1041</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E488 80</td>
<td>1042</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E489 80</td>
<td>1043</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E48A 80</td>
<td>1044</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E48B 80</td>
<td>1045</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E48C 80</td>
<td>1046</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E48D 80</td>
<td>1047</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E48E 80</td>
<td>1048</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E48F 80</td>
<td>1049</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E490 80</td>
<td>1050</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E491 80</td>
<td>1051</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E492 80</td>
<td>1052</td>
<td>PRT_SIZ:</td>
</tr>
<tr>
<td>E493 80</td>
<td>1053</td>
<td>PRT_SIZ:</td>
</tr>
</tbody>
</table>

System BIOS  A-15
A-16  System BIOS
<table>
<thead>
<tr>
<th>LOC OBJ</th>
<th>OBJ</th>
<th>LINE</th>
<th>E4</th>
<th>SUB BX,BX</th>
<th>1 SETUP STARTING MEM ADDR</th>
</tr>
</thead>
<tbody>
<tr>
<td>E93D</td>
<td>E93D</td>
<td>280B</td>
<td>E4</td>
<td>MOV DS:DX</td>
<td>0 CHECK BIOS</td>
</tr>
<tr>
<td>E56F</td>
<td>E56F</td>
<td>624A</td>
<td>E4</td>
<td>CALL BIOS</td>
<td>0 CONTINUE IF OK</td>
</tr>
<tr>
<td>E5A1</td>
<td>E5A1</td>
<td>E8AE13</td>
<td>E4</td>
<td>CALL BIOS</td>
<td>0 POST ERROR</td>
</tr>
<tr>
<td>E554</td>
<td>E554</td>
<td>7905</td>
<td>E4</td>
<td>ADD BX,200H</td>
<td>0 POINT TO NEXT 8K MODULE</td>
</tr>
<tr>
<td>E556</td>
<td>E556</td>
<td>E80001</td>
<td>E4</td>
<td>DEC AH</td>
<td>0 ANY MORE TO DO?</td>
</tr>
<tr>
<td>E56F</td>
<td>E56F</td>
<td>790C</td>
<td>E4</td>
<td>JNZ E4</td>
<td>0 YES - CONTINUE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E561</td>
<td>E561</td>
<td>1F</td>
<td>F9</td>
<td>POP DS</td>
<td>0 DISKETTE ATTACH TEST</td>
</tr>
<tr>
<td>E552</td>
<td>E552</td>
<td>A61000</td>
<td>F9</td>
<td>MOV DSPTR,256</td>
<td>0 DISKETTE PRESENT</td>
</tr>
<tr>
<td>E555</td>
<td>E555</td>
<td>2401</td>
<td>F9</td>
<td>MOV AL,01H</td>
<td>0 NO - BYPASS DISKETTE TEST</td>
</tr>
<tr>
<td>E557</td>
<td>E557</td>
<td>740F</td>
<td>F9</td>
<td>JZ F15</td>
<td>0 DISK_TEST:</td>
</tr>
<tr>
<td>E559</td>
<td>E559</td>
<td>7401</td>
<td>F10</td>
<td>IN AL,INT001</td>
<td>0 ENABLE DISKETTE INTERRUPTS</td>
</tr>
<tr>
<td>E55B</td>
<td>E55B</td>
<td>200F</td>
<td>F10</td>
<td>AND AL,0FFH</td>
<td>0 ENABLE INT001, AL</td>
</tr>
<tr>
<td>E560</td>
<td>E560</td>
<td>E621</td>
<td>F10</td>
<td>OUT INT001,AL</td>
<td>0 ENABLE INT001, AL</td>
</tr>
<tr>
<td>E56F</td>
<td>E56F</td>
<td>D000</td>
<td>F10</td>
<td>MOV AH,0</td>
<td>0 RESET NEC FDC</td>
</tr>
<tr>
<td>E561</td>
<td>E561</td>
<td>D004</td>
<td>F10</td>
<td>MOV DL,AL</td>
<td>0 SET FOR DRIVE 0</td>
</tr>
<tr>
<td>E565</td>
<td>E565</td>
<td>0013</td>
<td>F10</td>
<td>INT 13H</td>
<td>0 VERIFY STATUS AFTER RESET</td>
</tr>
<tr>
<td>E565</td>
<td>E565</td>
<td>F6C4FF</td>
<td>F10</td>
<td>TEST AH,0FFH</td>
<td>0 STATUS OK?</td>
</tr>
<tr>
<td>E566</td>
<td>E566</td>
<td>7520</td>
<td>F10</td>
<td>JNZ F13</td>
<td>0 NO - FDC FAILED</td>
</tr>
<tr>
<td>E55A</td>
<td>E55A</td>
<td>EAF203</td>
<td>F10</td>
<td>MOV DX,03FH</td>
<td>0 GET ADDR OF FDC CARD</td>
</tr>
<tr>
<td>E56D</td>
<td>E56D</td>
<td>001C</td>
<td>F10</td>
<td>MOV AL,CH</td>
<td>0 GROUP ON, IN DM/INT</td>
</tr>
<tr>
<td>E56F</td>
<td>E56F</td>
<td>EE</td>
<td>F10</td>
<td>OUT DX,AL</td>
<td>0 WRITE FDC CONTROL REG</td>
</tr>
<tr>
<td>E570</td>
<td>E570</td>
<td>0FCF</td>
<td>F11</td>
<td>SUB CX,00C</td>
<td>0 DISK 0 - MODE</td>
</tr>
<tr>
<td>E572</td>
<td>E572</td>
<td>0EFE</td>
<td>F11</td>
<td>LOOP F11</td>
<td>0 WAIT FOR 1 SECOND</td>
</tr>
<tr>
<td>E574</td>
<td>E574</td>
<td>EE</td>
<td>F12</td>
<td>LOOP F12</td>
<td>0 GROUP 0 MODE</td>
</tr>
<tr>
<td>E576</td>
<td>E576</td>
<td>3200</td>
<td>F12</td>
<td>XOR DX,DX</td>
<td>0 SELECT DRIVE 0</td>
</tr>
<tr>
<td>E57B</td>
<td>E57B</td>
<td>0501</td>
<td>F12</td>
<td>MOV CL,1</td>
<td>0 SELECT TRACK 1</td>
</tr>
<tr>
<td>E57A</td>
<td>E57A</td>
<td>80163C00</td>
<td>F12</td>
<td>MOV SEEK_STATUS,DL</td>
<td>0 SELECT TRACK 1</td>
</tr>
<tr>
<td>E57E</td>
<td>E57E</td>
<td>E0FC00</td>
<td>F12</td>
<td>CALL SEEK</td>
<td>0 RECALIBRATE DISKETTE</td>
</tr>
<tr>
<td>E561</td>
<td>E561</td>
<td>7007</td>
<td>F12</td>
<td>JC F13</td>
<td>0 GO TO ERR SUBROUTINE IF ERR</td>
</tr>
<tr>
<td>E563</td>
<td>E563</td>
<td>002C</td>
<td>F12</td>
<td>MOV CH,34</td>
<td>0 SELECT TRACK 34</td>
</tr>
<tr>
<td>E565</td>
<td>E565</td>
<td>E8F008</td>
<td>F12</td>
<td>CALL SEEK</td>
<td>0 SELECT TRACK 34</td>
</tr>
<tr>
<td>E566</td>
<td>E566</td>
<td>7007</td>
<td>F12</td>
<td>JNC F14</td>
<td>0 OK, FDC OFF</td>
</tr>
<tr>
<td>E56A</td>
<td>E56A</td>
<td>E552E00</td>
<td>F12</td>
<td>MOV DX,OFFSET F3</td>
<td>0 GET ADDR OF MSG</td>
</tr>
<tr>
<td>E5BE</td>
<td>E5BE</td>
<td>E01A14</td>
<td>F12</td>
<td>CALL E_MSG</td>
<td>0 GET PRINT ERR MSG</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E91</td>
<td>E91</td>
<td>1F</td>
<td>F12</td>
<td>POP DS</td>
<td>0 DISK OFF</td>
</tr>
<tr>
<td>E591</td>
<td>E591</td>
<td>D000</td>
<td>F12</td>
<td>MOV AL,0CH</td>
<td>0 GROUP 0 FDC OFF</td>
</tr>
<tr>
<td>E595</td>
<td>E595</td>
<td>EAF203</td>
<td>F12</td>
<td>MOV DX,03FH</td>
<td>0 FDC CTRL ADDRESS</td>
</tr>
<tr>
<td>E596</td>
<td>E596</td>
<td>EE</td>
<td>F12</td>
<td>OUT DX,AL</td>
<td>0 GROUP 0 MODE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>E97</td>
<td>E97</td>
<td>FI6</td>
<td>F12</td>
<td>MOV INTR_FLAG,00H</td>
<td>0 SETUP PRINTER BASE ADDRESSES IF DEVICE ATTACHED</td>
</tr>
<tr>
<td>E957</td>
<td>E957</td>
<td>C936680000</td>
<td>F12</td>
<td>MOV INTR_FLAG,00H</td>
<td>0 SETUP PRINTER BASE ADDRESSES IF DEVICE ATTACHED</td>
</tr>
<tr>
<td>E95C</td>
<td>E95C</td>
<td>BE1000</td>
<td>F12</td>
<td>MOV DS_OFFSET,10H</td>
<td>0 SETUP INTERRUPT BASE ADDRESSES IF DEVICE ATTACHED</td>
</tr>
<tr>
<td>E57F</td>
<td>E57F</td>
<td>09361A00</td>
<td>F12</td>
<td>MOV BUFFER_START,10H</td>
<td>0 SETUP PRINTER BASE ADDRESSES IF DEVICE ATTACHED</td>
</tr>
<tr>
<td>E5A3</td>
<td>E5A3</td>
<td>09361C00</td>
<td>F12</td>
<td>MOV BUFFER_START,10H</td>
<td>0 SETUP PRINTER BASE ADDRESSES IF DEVICE ATTACHED</td>
</tr>
<tr>
<td>E5A7</td>
<td>E5A7</td>
<td>09366008</td>
<td>F12</td>
<td>MOV BUFFER_END,10H</td>
<td>0 SETUP PRINTER BASE ADDRESSES IF DEVICE ATTACHED</td>
</tr>
<tr>
<td>E592</td>
<td>E592</td>
<td>E0C020</td>
<td>F12</td>
<td>MOV BUFFER_END,10H</td>
<td>0 SETUP PRINTER BASE ADDRESSES IF DEVICE ATTACHED</td>
</tr>
<tr>
<td>E59E</td>
<td>E59E</td>
<td>09366000</td>
<td>F12</td>
<td>MOV BUFFER_END,10H</td>
<td>0 SETUP PRINTER BASE ADDRESSES IF DEVICE ATTACHED</td>
</tr>
<tr>
<td>E59E</td>
<td>E59E</td>
<td>E7B000</td>
<td>F12</td>
<td>MOV BUFFER_END,10H</td>
<td>0 SETUP PRINTER BASE ADDRESSES IF DEVICE ATTACHED</td>
</tr>
<tr>
<td>E588</td>
<td>E588</td>
<td>EE</td>
<td>F12</td>
<td>PUSH DS</td>
<td>0 FDC CTRL ADDRESS</td>
</tr>
<tr>
<td>E596</td>
<td>E596</td>
<td>07</td>
<td>F12</td>
<td>POP ES</td>
<td>0 FDC CTRL ADDRESS</td>
</tr>
</tbody>
</table>
E5D0 0014H 1208 MOV AX,1414H 1 DEFAULT=20
E5D0 AB 1209 STOSW 1
E5D0 AB 1210 STOSW 1
E5D0 0010H 1211 MOV AX,8010H 1 RS232 DEFAULT=01
E5D0 AB 1212 STOSW 1
E5D0 AB 1213 STOSW 1
E5D1 4401 1214 IN AL,INTA01 1 ENABLE TIMER AND KB INTS
E5D2 492CH 1215 AND AL,0FH 1
E5D2 0011H 1216 OUT INTA01,AL 1
E5C7 03F00 1217 CMP BP,0000H 1 CHECK FOR BP={NON-ZERO
E5C7 0010H 1218 CMP IP,0000H 1 (ERROR HAPPENED)
E5C7 4920 1219 MOV DX,E 1 CONTINUE IF NO ERROR
E5C7 4814 1220 MOV AX,DX 1 2 SHORT BEEPS (ERROR)
E5C7 0014H 1221 CALL ERR.BEEP 1
E5C2 00E9H 1222 MOV SI,OFFSET F3D 1 LOAD ERROR MSG
E5C2 0013H 1223 CALL P_MSG 1
E5C9 009 1224 ERR.WAIT: 1
E5C9 0008H 1225 MOV AH,00 1 WAIT FOR 'F1' KEY
E5C9 1001H 1226 INT 10H 1
E5C9 10FCH 1227 CMP AH,38H 1
E5C3 0097H 1228 JNE ERR.WAIT 1
E5C3 00E9H 1229 JMP FISA 1 BYPASS ERROR
E5E5 0005H 1230 FISA: 1
E5E5 0005H 1231 CMP MFG.TST.1 1 MFG MODE
E5EA 0046H 1232 JE FISA 1 BYPASS BEEP
E5EC 0010H 1233 MOV DX,1 1 1 SHORT BEEP (NO ERRORS)
E5ED 0012H 1234 CALL ERR.BEEP 1
E5F2 0001H 1235 FISA: MOV AL,BYTE PTR EQUIP_FLAGS 1 GET SWITCHES
E5F5 2401H 1236 MOV AL,00000000ID 1 'LOOPS POST' SWITCH ON
E5F7 7005H 1237 JNZ FISA 1 CONTINUE WITH BRING-UP
E5F9 0057H 1238 JMP START 1
E5FC 02AEH 1239 FISB: SUB AL,40H 1
E5F2 0040H 1240 MOV AL,BRT_MODE 1
E601 0010H 1241 INT 10H 1 CLEAR SCREEN
E603 0042H 1242 FISC: MOV BP,OFFSET F4 1 PRM.SPC_TBL
E607 2000H 1243 MOV SI,8 1
E60A 0010H 1244 FISI: MOV DX,CS:[BP] 1 PRINT BASE
E60A 005640H 1245 MOV DX,CS:[BP] 1 PRINT BASE
E60E 004AH 1246 MOV DX,AL 1 WRITE DATA TO PORT A
E610 0046H 1247 MOV AL,4AH 1 WRITE DATA TO PORT A
E611 001E 1248 OUT DX,AL 1
E613 0049H 1249 PUSH DS 1 BUS SETTLING
E618 0010H 1250 IN AL,DX 1 READ PORT A
E619 0015H 1251 POP DS 1
E619 002AH 1252 MOV AL,0AH 1 DATA PATTERN SAME
E619 0055H 1253 JNE FISI 1
E619 0095H 1254 MOV AL,0AH 1 DATA PATTERN SAME
E61C 0015H 1255 INC SI 1 INCREMENT TO NEXT WORD
E61D 0010H 1256 INC SI 1
E61D 0045H 1257 FISI: INC BP 1 POINT TO NEXT BASE ADDR
E61D 0045H 1258 INC BP 1 POINT TO NEXT BASE ADDR
E61D 0045H 1259 INC BP 1 POINT TO NEXT BASE ADDR
E61F 01F440H 1260 MOV BP,OFFSET F4E 1 ALL POSSIBLE ADDRS CHECKED?
E623 70BEH 1261 JNE FISI 1 PRM_BASE
E625 0000H 1262 MOV BX,0 1 POINTER TO RS232 TABLE
E62B 00A3H 1263 MOV DX,3F9H 1 CHECK IF RS232 CD & ATTCH?
E62D 0014H 1264 IN AL,DX 1 READ INTR 3D REG
E62F 0010H 1265 TEST AL,0F8H 1
E630 0076H 1266 JNE FIB 1
E630 0070H 1267 MOV RS232_BASE[BX],3F9H 1 SETUP RS232 CD #1 ADDR
E634 0043H 1268 INC BX 1
E634 0043H 1269 INC BX 1
E634 0043H 1270 FIB: MOV DX,3F9H 1 CHECK IF RS232 CD & ATTCH
E637 0043H 1271 IN AL,DX 1 READ INTR 3D REG
E63A 0010H 1272 TEST AL,0F8H 1
E63C 7056H 1273 JNE F10 1 BASE.END
E63E 0070H 1274 INC BX 1
E63E 0070H 1275 MOV RS232_BASE[BX],2F0H 1 SETUP RS232 CD #2
E642 0043H 1276 INC BX 1
E642 0043H 1277 INC BX 1
E642 0043H 1278 INC BX 1
E642 0043H 1279 INC BX 1
E644 0010H 1279 INC BX 1
E644 0010H 1280 INC BX 1
E644 0010H 1281 INC BX 1
E648 0010H 1282 INC BX 1
E648 0010H 1283 INC BX 1
E648 0010H 1284 INC BX 1

A-18 System BIOS
LOC OBJ | LINE | SOURCE
--- | --- | ---
E64A 0AC5 | 1285 | OR AL,BL
E64C A21100 | 1286 | MOV BYTE PTR EQUIP_FLAGS+1,AL
E64F B0102 | 1287 | MOV DX,201H
E652 EC | 1288 | IN AL,DX
E653 90 | 1289 | NOP
E654 90 | 1290 | NOP
E655 90 | 1291 | NOP
E656 8D0F | 1292 | TEST AL,0FH
E656 7506 | 1293 | JNZ F20
E65A 00810010 | 1294 | OR BYTE PTR EQUF_FLAGS+1,16
E65F | 1295 | F20:
E65F | 1296 | JNL SAME_CARD:
E65F | 1297 | i----- ENABLE NMI INTERRUPTS
E66F E441 | 1299 | IN AL,PORT_B
E661 0C30 | 1300 | OR AL,0H
E663 E661 | 1301 | OUT PORT_B,AL
E665 24CF | 1302 | AND AL,0DFH
E667 E661 | 1303 | OUT PORT_B,AL
E669 8000 | 1304 | MOV AL,00H
E66B E640 | 1305 | OUT BASH,AL
E66D | 1306 | F21:
E66D C019 | 1307 | INT 10H
E66E 39A | 1308 | GO TO THE BOOLOADER
E66F 39A | 1309 | 
E66F 39A | 1310 | i-------------------
E66F 39A | 1311 | ; THIS SUBROUTINE PERFORMS A READ/WRITE STORAGE TEST ON A BLOCK:
E66F 39A | 1312 | ; OF STORAGE:
E66F 39A | 1313 | ; ENTRY REQUIREMENTS:
E66F 39A | 1314 | ; CS = ADDRESS OF STORAGE SEGMENT BEING TESTED
E66F 39A | 1315 | ; DS = ADDRESS OF STORAGE SEGMENT BEING TESTED
E66F 39A | 1316 | ; CX = WORD COUNT OF STORAGE BLOCK TO BE TESTED
E66F 39A | 1317 | ; EXIT PARAMETERS:
E66F 39A | 1318 | ; ZERO FLAG = 0 IF STORAGE ERROR (DATA COMPARE OR PARITY)
E66F 39A | 1319 | ; CHECK, ALSO DENOTES A PARITY CHECK. ELSE ALHIGHTED
E66F 39A | 1320 | ; BIT PATTERN OF THE EXPECTED DATA PATTERN VS THE ACTUAL:
E66F 39A | 1321 | ; DATA READ.
E66F 39A | 1322 | ; AX,BX,CX,DX,SI,DI, AND SI ARE ALL DESTROYED.
E66F 39A | 1323 | ;-------------------------------
E66F 39A | 1324 | PROC NEAR
E66F 39A | 1325 | SETDIR_CNT
E66F 39A | 1326 | SUB DI,CI
E66F 39A | 1327 | SUB AL,AX
E66F 39A | 1328 | SETUP FOR 0-FF PATTERN TEST
E66F 39A | 1329 | MOV (DI),AL
E66F 39A | 1330 | ON FIRST BYTE
E66F 39A | 1331 | MOV AL,(DI)
E66F 39A | 1332 | D.K.? OR AL,
E66F 39A | 1333 | JNZ C7
E66F 39A | 1334 | INHIBIT AH
E66F 39A | 1335 | MOV AL,AH
E66F 39A | 1336 | LOOP TILL WRAP THROUGH FF
E66F 39A | 1337 | MOV BX,CX
E66F 39A | 1338 | SAVETHEDS COUNT OF BLOCK TO TEST
E66F 39A | 1339 | MOV BX,BX
E66F 39A | 1340 | INITIALIZE DATA PATTERN TO WRITE
E66F 39A | 1341 | MOV DX,OFFSET SELECTORS
E66F 39A | 1342 | ADD BASE LOCS IN BLOCK
E66F 39A | 1343 | LOOP IMMEDIATE PKT
E66F 39A | 1344 | SET DIR FLAG TO INCREMENT
E66F 39A | 1345 | MOV BX,AX
E66F 39A | 1346 | ADD AL,00H
E66F 39A | 1347 | OR AL,00H
E66F 39A | 1348 | MOV AL,00H
E66F 39A | 1349 | MOV AL,00H
E66F 39A | 1350 | MOV AX,0000H
E66F 39A | 1351 | MOV CX,AP
E66F 39A | 1352 | MOV CX,AX
E66F 39A | 1353 | INITIATE DESTINATION POINTERS
E66F 39A | 1354 | MOV BX,AX
E66F 39A | 1355 | SETUP BYTES FOR LOOP
E66F 39A | 1356 | LOOP IMMEDIATE PKT
E66F 39A | 1357 | READ OLD TEST BYTE FROM STORAGE [BITB0K 32
E66F 39A | 1358 | DATA READ AS EXPECTED?
E66F 39A | 1359 | MOV AL,DL
E66F 39A | 1360 | GET NEXT DATA PATTERN TO WRITE
E66F 39A | 1361 | MOV AL,BL
E66F 39A | 1362 | WRITE INTO LOC JUST READ [DI]+1
E66F 39A | 1363 | MOV CL,0
E66F 39A | 1364 | DECREMENT BYTE COUNT AND LOOP CX
E66F 39A | 1365 | MOV AH,0
E66F 39A | 1366 | ENDING ZERO PATTERN WITH ZERO PRESERVED?
E66F 39A | 1367 | MOV AH,0
E66F 39A | 1368 | YES - RETURN TO CALLER WITH AL=0

System BIOS  A-19
A-20  System BIOS
E710 8ECE 1440  MOV ES:DX  
E712 B0007C 1441  MOV BX,OFFSET BOOT_LOCK 
E715 B90110 1442  MOV CX,1  
E716 CD13 1443  INT 13H  
E71A 0D 1444  DISKETTE_ID 
E71A B9 1445  POP CX  
E716 73D4 1446  JMP H4  
E71D E2E5 1447  LOOP H1 
E71D 1448  I----- UNABLE TO IPL FROM THE DISKETTE 
E71F 1449  H3:  
E71F 0100 1450  INT 16H  
E71F 1451  I----- IPL WAS SUCCESSFUL 
E721 1452  H4:  
E721 EA007C8003 1453  JMP BOOT_LOCK 
E721 1454  B0007D 1455  ENDP  
E721 1456  I----- INT 14------------------------------------------------------- 
E721 1457  7  6  5  4  3  2  1  0  
E721 1458  ---- BAUD RATE ---- -PARITY- STOPBIT --WORD LENGTH--  
E721 1459  000 - 110 X0 - NONE 0 - 1 10 - 7 BITS  
E721 1460  001 - 150 01 - ODD 1 - 2 11 - 8 BITS  
E721 1461  010 - 300 11 - EVEN  
E721 1462  011 - 600  
E721 1463  100 - 1200  
E721 1464  101 - 2400  
E721 1465  110 - 4800  
E721 1466  111 - 9600  
E721 1467  ON RETURN, CONDITIONS SET AS IN CALL TO COMPO STATUS (AH=3) 
E721 1468  (AH=1) SEND THE CHARACTER IN (AL) OVER THE COMM LINE 
E721 1469  (AH=4) REGISTER IS PRESERVED 
E721 1470  ON EXIT, BIT 7 OF AH IS SET IF THE ROUTINE WAS UNABLE 
E721 1471  TO TRANSMIT THE BYTE OF DATA OVER THE LINE.  
E721 1472  IF BIT 7 OF AH IS NOT SET, THE REMAINDER OF AH 
E721 1473  IS SET AS IN A STATUS REQUEST, REFLECTING THE 
E721 1474  CURRENT STATUS OF THE LINE.  
E721 1475  (AH=2) RECEIVE A CHARACTER IN (AL) FROM COMM LINE BEFORE 
E721 1476  RETURNING TO CALLER 
E721 1477  ON EXIT, AH HAS THE CURRENT LINE STATUS, AS SET BY THE 
E721 1478  THE STATUS ROUTINE, EXCEPT THAT THE ONLY BITS 
E721 1479  LEFT ON ARE THE ERROR BITS (7,4,3,2,1)  
E721 1480  IF AH HAS BIT 7 ON, TIME OUT, THE REMAINING 
E721 1481  BITS ARE NOT PREDICTABLE.  
E721 1482  THUS, AH IS NOW ZERO ONLY WHEN AN ERROR 
E721 1483  OCCURRED. 
E721 1484  (AH=3) RETURN THE COMM PORT STATUS IN (AX) 
E721 1485  AH CONTAINS THE LINE STATUS 
E721 1486  BIT 7 = TIME OUT  
E721 1487  BIT 6 = TRANSMIT REGISTER EMPTY 
E721 1488  BIT 5 = TRANSMIT HOLDING REGISTER EMPTY 
E721 1489  BIT 4 = BREAK DETECT 
E721 1490  BIT 3 = FRAMING ERROR 
E721 1491  BIT 2 = PARITY ERROR 
E721 1492  BIT 1 = OVERRUN ERROR 
E721 1493  BIT 0 = DATA READY 
E721 1494  AL CONTAINS THE MODEM STATUS 
E721 1495  BIT 7 = RECEIVED LINE SIGNAL DETECT 
E721 1496  BIT 6 = RING INDICATOR 
E721 1497  BIT 5 = DATA SET READY 
E721 1498  BIT 4 = CLEAR TO SEND 
E721 1499  BIT 3 = DELTA RECEIVE LINE SIGNAL DETECT 
E721 1499  BIT 2 = TRAILING EDGE RING DETECTOR 
E721 1495  BIT 1 = DELTA DATA SET READY 
E721 1494  BIT 0 = DELTA CLEAR TO SEND 
E721 1493  (DX) = PARAMETER INDICATING WHICH BYTE CARD (0,1 ALIGNED)  

APPENDIX A

SYSTEM BIOS A-21
LOC OBJ

1517 1
1518 1 DATA AREA R5232_BASE CONTAINS THE BASE ADDRESS OF THE 8250 ON THE
1519 1 CARD LOCATION 400H CONTAINS UP TO 4 R5232 ADDRESSES POSSIBLE
1520 1 DATA AREA LABEL R5232_TH2_OUT (BYTE) CONTAINS OUTER LOOP COUNT
1521 1 VALUE FOR TIMEOUT (DEFAULT=1)
1522 1 OUTPUT
1523 1 AX MODIFIED ACCORDING TO PARM'S OF CALL
1524 1 ALL OTHERS UNCHANGED
1525 1
1526 1-----------------------------------------------------------------------
1527 1 ASSUME CS:CODE,DS:DATA
1528 1
1529 1 ORG 0E729H
1530 1
1531 1 E729
1532 1 1520 A1 LABEL WORD 1 TABLE OF INIT VALUES
1533 1
1534 1 E729 1704
1535 1 1519 DW 1247 1 110 BAUD
1536 1
1537 1 E729 6003
1538 1 1530 DW 768 1 150
1539 1
1540 1 E729 4001
1541 1 1531 DW 304 1 300
1542 1
1543 1 E729 0000
1544 1 1532 DW 192 1 600
1545 1
1546 1 E731 6000
1547 1 1533 DW 96 1 1200
1548 1
1549 1 E733 3000
1550 1 1534 DW 40 1 2400
1551 1
1552 1 E735 1000
1553 1 1535 DW 24 1 4800
1554 1
1555 1 E737 0000
1556 1 1536 DW 12 1 9600
1557 1
1558 1 E739 1538 R5232_ID
1559 1 1539 PROC FAR
1560 1
1561 1 1540 1----- VECTOR TO APPROPRIATE ROUTINE
1562 1
1563 1 E739 FB
1564 1 1542 STI 1 INTERRUPTS BACK ON
1565 1
1566 1 E73A 1E
1567 1 1543 PUSH DS 1 SAVE SEGMENT
1568 1
1569 1 E73B 52
1570 1 1544 PUSH DX
1571 1
1572 1 E73C 56
1573 1 1545 PUSH SI
1574 1
1575 1 E73D 57
1576 1 1546 PUSH DI
1577 1
1578 1 E73E 51
1579 1 1547 PUSH CX
1580 1
1581 1 E73F 53
1582 1 1548 PUSH BX
1583 1
1584 1 E740 0000
1585 1 1549 MOV SI,DX 1 R5232 VALUE TO SI
1586 1
1587 1 E742 0000
1588 1 1550 MOV DL,DX
1589 1
1590 1 E744 D16
1591 1 1551 SHL SI,1 1 WORD OFFSET
1592 1
1593 1 E746 81013
1594 1 1552 CALL 0DS
1595 1
1596 1 E749 6004
1597 1 1553 MOV DX,R5232_BASE+111 1 GET BASE ADDRESS
1598 1
1599 1 E74B 6004
1599 1 1554 OR DX,DX 1 TEST FOR 0 BASE ADDRESS
1600 1
1601 1 E74C 7013
1602 1 1555 JZ AS 1 RETURN
1603 1
1604 1 E74D 8A4
1605 1 1556 OR AH,AH 1 TEST FOR (AH)=0
1606 1
1607 1 E74F 7416
1608 1 1557 JZ AA 1 COMMUN INIT
1609 1
1610 1 E753 7EE5
1611 1 1558 DEC AH 1 TEST FOR (AH)=1
1612 1
1613 1 E755 745C
1614 1 1559 JZ AS 1 SEND AL
1615 1
1616 1 E757 745F
1617 1 1560 DEC AH 1 TEST FOR (AH)=2
1618 1
1619 1 E759 746A
1620 1 1561 JZ A12 1 RECEIVE INTO AL
1621 1
1622 1 E75B 1562 A2:
1623 1 1563 MOV CX
1624 1
1625 1 E75D 7603
1626 1 1564 JNZ AS 1 TEST FOR (AH)=3
1627 1
1628 1 E75F E9320
1629 1 1565 JMP A10 1 COMMUNICATION STATUS
1630 1
1631 1 E762 1566 A3:
1632 1 1566 CALL FROM R5232
1633 1
1634 1 E766 5B
1635 1 1567 POP BX
1636 1
1637 1 E766 59
1638 1 1568 POP CX
1639 1
1640 1 E766 5F
1641 1 1569 POP DI
1642 1
1643 1 E765 5E
1644 1 1570 POP SI
1645 1
1646 1 E766 5A
1647 1 1571 POP DX
1648 1
1649 1 E767 1F
1650 1 1572 POP DS
1651 1
1652 1 E768 CF
1653 1 1573 INT 21H 1 RETURN TO CALLER, NO ACTION
1654 1
1655 1 E769 1574
1656 1
1657 1 E769 004
1658 1 1575 1----- INITIALIZE THE COMMUNICATIONS PORT
1659 1
1660 1 E769 00E
1661 1 1576 1
1662 1 E769 0AE
1663 1 1577 A4:
1664 1 1578 MOV AH,AL 1 SAVE INIT PARM'S IN AH
1665 1
1666 1 E768 63C205
1667 1 1579 ADD DX,3 1 POINT TO 6250 CONTROL REGISTER
1668 1
1669 1 E76E 8005
1670 1 1580 MOV AL,B8H
1671 1
1672 1 E770 1E
1673 1 1581 OUT DX,AL 1 SET DLB8:
1674 1
1675 1 E783 1582
1676 1 1583 1----- DETERMINE BAUD RATE DIVISOR
1677 1
1678 1 E771 0A0
1679 1 1584
1680 1 E773 0104
1681 1 1585
1682 1 E775 020C
1683 1 1586
1684 1 E777 01E00E0
1685 1 1587
1686 1 E77B D1F2ED
1687 1 1588
1688 1 E77C 05FA
1689 1 1589
1690 1 E77D 0101
1691 1 1590
1692 1 E77E 42
1693 1 1591
1694 1 E783 6EAA581
1695 1 1592

A-22 System BIOS
ET97  EE   1994   OUT  DX,AL  ; SET MS OF DIV TO 0
ET98  4A   1995   DEC  DX
ET99  28A08  1996   MOV  AL,BC:IDII  ; GET LOW ORDER OF DIVISOR
ET9C  EE   1997   OUT  DX,AL  ; GET LOW ORDER OF DIVISOR
ET9D  83C203  1998   ADD  DX,3
ET9F  0AC4  1999   MOV  AL,AH  ; GET PARGS BACK
ET9F  241F  1999   AND  AL,0FH  ; STRIP OFF THE BAND BITS
ET9F  EE   2000   OUT  DX,AL  ; LINE CONTROL TO 8 BITS
ET9F  4A   2001   DEC  DX
ET9F  04A   2002   DEC  DX
ET9F  DD00  2003   MOV  AL,0
ET99  EE   2004   OUT  DX,AL  ; INTERRUPT ENABLES ALL OFF
ET9A  ED49  2005   JMP  SHORT A1B  ; COM_STATUS
2006   OUT  DX,AL  ; CMPL_STATUS

1607   ---- SEND CHARACTER IN I ALI OVER COMM LINE
2008

ET9C  1609   1610   A5:  PUSH  AX  ; SAVE CHAR TO SEND
ET9C  50   1611   1612   ADD  DX,A  ; MODM CONTROL REGISTER
ETAC  8035  1613   MOV  AL,3  ; DTR AND RTS
ET9E  EE   1614   OUT  DX,AL  ; DATA TERMINAL READY, REQUEST TO SEND
ET9E  42   1615   INC  DX  ; MODEM STATUS REGISTER
ET9E  42   1616   INC  DX  ; MODEM STATUS REGISTER
ETAC  9350  1617   MOV  BH,3OH  ; DATA SET READY & CLEAR TO SEND
ETAC  E84000  1618   CALL  WAIT_FOR_STATUS  ; ARE BOTH TRUE
ETAA  7A05  1619   JE  A9  ; YES, READY TO TRANSMIT CHAR
ET9C  1620   A7:  POP  CX  ; RELAOD DATA BYTE
ETAC  BACI  1621   MOV  AL,CL
ET9F  1622   1623   AB:  OR  AH,0BH  ; INDICATE TIME OUT
ET9C  0CC08  1624   1625   XT  A3  ; RETURN
ET9C  EBAE  1626   1627   AV:  DEC  DX  ; CLEAR_TO_SEND
ET9C  4A   1628   1629   MOV  DM:2DH  ; M Chiến TRANSMITTER READY
ET9C  1630   CALL  WAIT_FOR_STATUS  ; TEST FOR TRANSMITTER READY
ET9C  ECF0  1631   JNE  A7  ; RETURN WITH TIME OUT SET
ET9C  1632   A11:  SUB  DX,5  ; DATA PORT
ET9B  83A05  1633   PDP  CX  ; RECOVER IN CX TEMPORARILY
ET9C  59   1634   MOV  AL,CL  ; MOVE CHAR TO AL FOR OUT, STATUS IN AN
ET9C  EEE   1635   OUT  DX,AL  ; OUTPUT CHARACTER
ET9C  E900  1636   JMP  A3  ; RETURN

1637   ---- RECEIVE CHARACTER FROM COMM LINE
1638

ET9C  1639   1640   A12:  ADD  DX,4  ; MODM CONTROL REGISTER
ET9C  8001  1641   MOV  AL,1  ; DATA TERMINAL READY
ET9C  EE   1642   1643   OUT  DX,AL  ; MODEM STATUS REGISTER
ET9C  42   1644   INC  DX  ; MODEM STATUS REGISTER
ET9C  42   1645   INC  DX  ; MODEM STATUS REGISTER
ET9C  1646   A13:  MOV  DM:10H  ; DATA SET READY
ET9C  8720  1647   CALL  WAIT_FOR_STATUS  ; TEST FOR DSR
ET9C  882200  1648   JNZ  A8  ; RETURN WITH ERROR
ET9C  750B  1649   JNZ  A8  ; RETURN WITH ERROR
ET9C  1650   A15:  DEC  DX  ; LINE STATUS REGISTER
ET9C  4A   1651   1652   MOV  AH,1  ; RECEIVE BUFFER FULL
ET9C  8701  1653   JNZ  A8  ; RECEIVED ERROR
ET9C  E51000  1654   CALL  WAIT_FOR_STATUS  ; TEST FOR REC. BUFF. FULL
ET9C  7503  1655   JNZ  A8  ; SET TIME OUT ERROR
ET9C  1656   A17:  MOV  AH,00H  ; GET_CHAR
ET9C  80481E  1657   AND  AH.00H  ; TEST FOR ERR CONDITIONS ON RECV CHAR
ET9C  8814  1658   MOV  DM:2832_BASE+511  ; DATA PORT
ET9C  1659   IN  AL,DX  ; GET CHARACTER FROM LINE
ET9C  E9D7FF  1660   JMP  A3  ; RETURN

1661   ---- COMM PORT STATUS ROUTINE
1662

ET9C  1663   1664   A18:  MOV  DX.8523H_BASE+511  ; CONTROL PORT
ET9C  8014  1665   1666   ADD  DX,8  ; GET LINE CONTROL STATUS
ET9C  83C205  1667   IN  AL,DX  ; PUT IN AH FOR RETURN
ET9C  6A40  1668   MOV  AH,AL
ET9C  1669   INC  DX  ; POINT TO MODM STATUS REGISTER

Appendix A

System BIOS  A-23
LOC OBJ | LINE | SOURCE
--- | --- | ---
E7EE EC | 1671 | IN AL,DX \ | SET MODEM CONTROL STATUS
E7EF E70FF | 1672 | JMP AS \ | RETURN
E7F2 | 1673 | \-----------------------------\ | 
E7F2 04507C | 1674 | \ | \ | 
E7F5 | 1675 | \ | \ | 
E7F6 | 1676 | \ | \ | 
E7F7 | 1677 | \ | \ | 
E7F8 0A507C | 1678 | \ | \ | 
E7F9 | 1679 | \ | \ | 
E7FA 22C7 | 1680 | \ | \ | 
E7FC 3AC7 | 1681 | \ | \ | 
E7FF 7A00 | 1682 | \ | \ | 
E800 E8FF | 1683 | \ | \ | 
E802 F2CB | 1684 | \ | \ | 
E804 75EF | 1685 | \ | \ | 
E806 A0F | 1686 | \ | \ | 
E80C C3 | 1687 | \ | \ | 
E809 4552524552EE20 | 1688 | \ | \ | 
E80C 4552524552EE20 | 1689 | \ | \ | 
E80C 4552524552EE20 | 1690 | \ | \ | 
E80C 4552524552EE20 | 1691 | \ | \ | 
E80C 4552524552EE20 | 1692 | \ | \ | 
E80C 4552524552EE20 | 1693 | \ | \ | 
E823 0D | 1694 | \ | \ | 
E824 0A | 1695 | \ | \ | 
E825 0D | 1696 | \ | \ | 
E828 0E | 1697 | \ | \ | 
1705 | 1706 | \ | \ | 
1707 | 1708 | \ | \ | 
1709 | 1710 | \ | \ | 
1711 | 1712 | \ | \ | 
1713 | 1714 | \ | \ | 
1715 | 1716 | \ | \ | 
1717 | 1718 | \ | \ | 
1719 | 1720 | \ | \ | 
1721 | 1722 | \ | \ | 
1723 | 1724 | \ | \ | 
1725 | 1726 | \ | \ | 
1727 | 1728 | \ | \ | 
1729 | 1730 | \ | \ | 
1731 | 1732 | \ | \ | 
1733 | 1734 | \ | \ | 
1735 | 1736 | \ | \ | 
1737 | 1738 | \ | \ | 
1739 | 1740 | \ | \ | 
1741 | \ | \ | 

A-24 System BIOS
LOC OBJ  |  LINE  | SOURCE
------- | ------ | ------------
E642 FB | 1743   | STI         | INTERRUPTS BACK ON DURING LOOP
E645 90 | 1744   | NOP         | ALLOW AN INTERRUPT TO OCCUR
E646 FA | 1745   | CLI         |  INTERRUPTS BACK OFF
E845 001EAD00 | 1746 | MOV BX,BUFFER HEAD | GET POINTER TO HEAD OF BUFFER
E849 381ECDC0 | 1747 | CMP BX,BUFFER_TAIL | TEST END OF BUFFER
E84A 7AF3 | 1748 | JZ K1 | LOOP UNTIL SOMETHING IN BUFFER
E84F 8007 | 1749 | MOV AX,(B) | GET SCAN CODE AND ASCII CODE
E851 001D00 | 1750 | CALL K5 | MOVE POINTER TO NEXT POSITION
E85A 091EAD00 | 1751 | MOV BUFFER HEAD,BX | STORE VALUE IN VARIABLE
E85B ED14 | 1752 | JMP SHORT INT10 END | RETURN

1753

1754 | ASCII STATUS
1755

1756 | K2:
1757 | CLI
1758 | MOV BX,BUFFER HEAD | INTERRUPTS OFF
1759 | CMP BX,BUFFER_TAIL | GET HEAD POINTER
1760 | MOV AX,(B) | IF EQUAL (Z=1) THEN NOTHING THERE
1761 | STI | INTERRUPTS BACK ON
1762 | POP BX | RECOVER REGISTER
1763 | POP DS | RECOVER SEGMENT
1764 | RET 2 | THROW AWAY FLAGS
1765

1766 | SHIFT STATUS
1767

1768 | K5:
1769 | MOV AL,KB_FLAG | GET THE SHIFT STATUS FLAGS
1770 | INT10 END | POP BX | RECOVER REGISTER
1771 | POP DS | POP Registers | RETURN TO CALLER
1772 | INET | KEYSBOARD.IO END | Keyboard,IO END
1773

1774 | INCREMENT A BUFFER POINTER
1775

1776 | K4 PROC | MOVE TO NEXT WORD IN LIST
1777

1778 | JNC BX | MOVE TO NEXT WORD IN LIST
1779 | INC BX
1780

1781 | CMP BX,BUFFER_END | AT END OF BUFFER?
1782 | JNE K5 | NO, CONTINUE
1783 | MOV BX,BUFFER_START | YES, RESET TO BUFFER BEGINNING
1784

1785 | KEYS
1786 | K4 END | TABLE OF SHIFT KEYS AND MASK VALUES
1787

1788 | K6 LABEL BYTE | INSERT KEY
1789 | LEFT_KEY.RIGHT_KEY | DB INB_KEY | DB CHAP_KEY,HUM_KEY,SCROLL_KEY,ALT_KEY,CTL_KEY
1790

1791 | K6L EQU 8-K6
1792

1793 | LETTER_DATA | TABLE_SHIFT_TABLE
1794

1795 | K7 LABEL BYTE | INSERT MODE SHIFT
1796 | DB INB SHIFT | DB CHAP SHIFT,IBM SHIFT,SCROLL SHIFT,ALT SHIFT,CTL SHIFT
1797

1798 | DB LEFT_SHIFT.RIGHT_SHIFT | SCAN CODE TABLES
1799

1800 | K8 DB 27,1,0,1,1,1,-1,-1,-1,138,-1
1801

1802

1803 | SCAN CODE TABLES
1804

1805 | KB DB 27,1,0,1,1,1,-1,-1,-1,138,-1
1806

1807 | E8FF FF
1808 | E890 80
1809 | E891 FF
1810

1811

System BIOS  A-25
Appendix A

System BIOS  A-27
A-28  System BIOS
LOC OBJ | LINE | SOURCE
--- | --- | ---
E902 00261700 | 1904 | OR KB_FLAG, AH : TURN ON SHIFT BIT
E906 002617000 | 1905 | JMP K26 : INTERRUPT_RETURN
E906 002617000 | 1906 | JMP K26 : INTERRUPT_RETURN
E907 002617000 | 1907 | I----- TOGGLING SHIFT KEY, TEST FOR 1ST MAKE OR NOT
E909 002617000 | 1908 | K28: | SHIFT-TOGGLE
E909 F606170000 | 1909 | JNC K28 : JUMP IF CF TRAP IS NOT
E90E 7505 | 1910 | JMP K26 : INTERRUPT_RETURN
E90E 7505 | 1911 | JNC K28 : JUMP IF CF TRAP IS NOT
E90E 7505 | 1912 | TEST KB_FLAG, CTL_SHIFT : CHECK CTRL SHIFT STATE
E90E 7505 | 1913 | CMP AL, INH_KEY : CHECK FOR INSERT KEY
E90E 7505 | 1914 | JNZ K28 : JUMP IF NOT INSERT KEY
E90E 7505 | 1915 | TEST KB_FLAG, ALT_SHIFT : CHECK FOR ALTERNATE SHIFT
E90E 7505 | 1916 | JNC K28 : JUMP IF ALTERNATE SHIFT
E90E F606170020 | 1917 | K19: | Toggling SHIFT KEY, TEST FOR 1ST MAKE OR NOT
E90F 7500 | 1918 | JNC K28 : JUMP IF CF TRAP IS NOT
E90F 7500 | 1919 | TEST KB_FLAG, LEFT_SHIFT+RIGHT_SHIFT : CHECK FOR BASE STATE
E90F 7500 | 1920 | JZ K28 : JUMP IF BASE STATE
E90F 7500 | 1921 | JZ K28 : JUMP IF BASE STATE
E90F 7500 | 1922 | MOV AX, SEZMH : PUT OUT AN ASCII ZERO
E90F 7500 | 1923 | JMP K57 : BUFFER_FILL
E90F 7500 | 1924 | K21: | RIGHT BE HENDING NUMERIC
E90F F60617000 | 1925 | K22: | NUMERIC ZERO, NOT INSERT KEY
EA04 74F3 | 1926 | MOV AX, SEZM : PUT OUT AN ASCII ZERO
EA04 74F3 | 1927 | JZ X80 : JUMP TO INPUT BUFFER
EA04 74F3 | 1928 | K22: | NUMERIC, NOT INSERT KEY
EA06 00261700 | 1929 | TEST AH, KB_FLAG—I : PROCESS IT
EA06 00261700 | 1930 | JNZ K26 : JUMP IF KEY ALREADY DEPRESSED
EA0A 7500 | 1931 | OR KB_FLAG, AH—I : INDICATE THAT THE KEY IS DEPRESSED
EA0C 00261700 | 1932 | JNZ K26 : JUMP IF ALTERNATE SHIFT
EA10 50261700 | 1933 | XOR KB_FLAG, AH—I : Toggling THE SHIFT STATE
EA14 3C82 | 1934 | CMP AL, INH_KEY : TEST FOR 1ST MAKE OR NOT INSERT KEY
EA16 7541 | 1935 | JNZ K28 : JUMP IF NOT INSERT KEY
EA16 B00502 | 1936 | MOV AX, INH_KEY+286 : SET SCAN CODE INTO AH, D INTO AL
EA18 EB701 | 1937 | JMP K57 : BUFFER_FILL
EA18 EB701 | 1938 | I----- BREAK SHIFT FOUND
EA1B 74F3 | 1939 | K23: | BREAK-SHIFT-FOUND
EA1B 74F3 | 1940 | CMP AH, SCROLL_SHIFT : IS THIS A TOGGLE KEY
EA31 751A | 1941 | JAE K28 : YES, HANDLE BREAK TOGGLE
EA32 F60 | 1942 | HLT AH : DEPRESS MASK
EA35 20261700 | 1943 | AND KB_FLAG, AH—I : TURN OFF SHIFT BIT
EA39 3C82 | 1944 | CMP AL, ALT_KEY+60H : IS THIS ALTERNATE SHIFT RELEASE
EA3B 752C | 1945 | JNC K28 : INTERRUPT_RETURN
EA3F 7500 | 1946 | NOT AH : BREAK-SHIFT
EA3F 7500 | 1947 | AND KB_FLAG, AH—I : DEPRESS MASK
EA43 EB14 | 1948 | JMP SHORT K26 : INTERRUPT_RETURN
EA45 K25: | 1949 | I----- ALTERNATE SWIFT KEY RELEASED, GET THE VALUE INTO BUFFER
EA45 00261700 | 1950 | MVI AL, ALT_INPUT : SCAN CODE OF 0
EA45 00261700 | 1951 | MVI AH, 0 : SCAN CODE OF 0
EA45 F60617000 | 1952 | MVI AL, ALT_INPUT, AH : SCAN CODE FROM THE 1ST MAKE OR NOT
EA45 3C82 | 1953 | CMP AL, 0 : SCAN CODE FROM THE 1ST MAKE OR NOT
EA4B 741F | 1954 | JZ K28 : INTERRUPT_RETURN
EA4B 741F | 1955 | JE K28 : INTERRUPT_RETURN
EA4E 7500 | 1956 | JMP K57 : IT TURNS OUT, SO PUT IN BUFFER
EA4E 7500 | 1957 | X80: | BREAK-SHIFT
EA4F 7500 | 1958 | AND KB_FLAG, AH—I : INDICATE NO LONGER DEPRESSED
EA50 20261700 | 1959 | JMP K26 : INTERRUPT_RETURN
EA5C EB14 | 1960 | JMP SHORT K26 : INTERRUPT_RETURN
EA5C EB14 | 1961 | I----- TEST FOR HOLD STATE
EA65 K25: | 1962 | I----- TEST FOR HOLD STATE
EA65 00261700 | 1963 | CMP AL, 80H : TEST FOR BREAK KEY
EA65 00261700 | 1964 | JNZ K28 : JUMP IF NOT BREAK
EA65 3C82 | 1965 | CMP AL, 80H : TEST FOR BREAK KEY
EA67 751A | 1966 | JAE K28 : JUMP IF NOT BREAK
EA6E F60617000 | 1967 | TEST KB_FLAG, ALT_HOLD_STATE : ARE WE IN HOLD STATE
EA74 7500 | 1968 | JZ K28 : JUMP IF NOT BREAK
EA79 3C82 | 1969 | CMP AL, INH_KEY : TEST FOR BREAK CHARS FROM HERE ON
EA7E 7500 | 1970 | JE K28 : JUMP IF NOT BREAK
EA84 002618000 | 1971 | K26: | TURN OFF THE HOLD STATE BIT
EA84 002618000 | 1972 | AND KB_FLAG, ALT_HOLD_STATE : TURN OFF THE HOLD STATE BIT
EA89 7A | 1973 | CLI : INTERRUPT-RETURN
EA8F 0020 | 1974 | MOV AL, EO : END OF INTERRUPT COMMAND
EA9E 8020 | 1975 | OUT 020H, AL : SEND COMMAND TO INT CONTROL PORT
EA9E 8020 | 1976 | K27: | INTERRUPT-RETURN-HD-EOI
EAA5 00261700 | 1977 | POP ES : INTERRUPT-RETURN
EAA5 00261700 | 1978 | POP DS : INTERRUPT-RETURN
EAA6 00261700 | 1979 | POP DI : INTERRUPT-RETURN
EAA6 00261700 | 1980 | POP SI : INTERRUPT-RETURN

System BIOS  A-29
A-30  System BIOS
EB44 2026  K32:  MOV  DI,OFFSET K32  ; ALT-KEY-PAD
EB44 BF07EA 2029  MOV  CIX,10  ; LOOK FOR ENTRY USING KEYPAD
EB57 89A40 2030  MOV  CIX,10  ; LOOK FOR MATCH
EABF F2 2031  REPNE  SCANB  ; LOOK FOR MATCH
EABF AE 2032  JNE  K33  ; NO_ALT-KEYPAD
EABE 81FF00EA 2033  SUB  DI,OFFSET K33+1  ; DI NOW HAS ENTRY VALUE
EAC8 86990 2034  MOV  AL,ALT_INPUT  ; GET THE CURRENT BYTE
EACC 84A0 2035  MOV  AH,10  ; MULTIPLY BY 10
EAC7 80A0 2036  MUL  AH  ;
EAC9 35C7 2037  ADD  AX,DX  ; ADD IN THE LATEST ENTRY
EABC 81900 2038  MOV  ALT_INPUT,AL  ; STORE IT AWAY
EABE 1809 2039  JMP  K26  ; THROW AWAY THAT KEYSTROKE

1991  ;----- LOOK FOR SUPERSHIFT ENTRY
EADD 2040
EADD C606190000 2041  MOV  ALT_INPUT,0  ; ZERO ANY PREVIOUS ENTRY INTO INPUT
EADD 85140 2042  MOV  CIX,16  ; DI,ES ALREADY POINTING
EADD F2 2043  REPNE  SCANB  ; LOOK FOR MATCH IN ALPHABET
EADD AE 2044  JNE  K34  ; NOT FOUND, FUNCTION KEY OR OTHER
EADD 8000 2045  MOV  AL,0  ; ASCII CODE OF ZERO
EADD EF4400 2046  JMP  K57  ; PUT IT IN THE BUFFER

1999  ;----- LOOK FOR TOP ROW OF ALTERNATE SHIFT
EALE 2050
EALE 3C02 2051  CMP  AL,1  ; ALT-TOP-ROW
EAES 730C 2052  JB  K35  ; KEY WITH '1' ON IT
EAES 5C00 2053  CMP  AL,14  ; IN THE REGION
EAES 7300 2054  JAE  K36  ; ALT-FUNCTION
EAEX 80C476 2055  ADD  AH,116  ; CONVERT PSEUDO SCAN CODE TO RANGE
EAEX 8000 2056  MOV  AH,3  ; INDICATE AS SUCH
EAEX 64400 2057  JMP  K57  ; BUFFER_FILL

1991  ;----- TRANSLATE ALTERNATE SHIFT PSEUDO SCAN CODES
EAFL 2061
EAFL 3C02 2062  CMP  AL,159  ; TEST FOR IN TABLE
EAFL 7305 2063  JAE  K37  ; ALT-CONTINUE
EAFL 6767 2064  CMP  AL,14  ; IN KEYPAD REGION
EAFL 861FF 2065  JMP  K26  ; IGNORE THE KEY
EAFL 86990 2066  CMP  AL,71  ; IN KEYPAD REGION
EAFL 73F9 2067  JAE  K36  ; IF SO, IGNORE
EAFL 80B79 2068  MOV  BX,OFFSET K13  ; ALT SHIFT PSEUDO SCAN TABLE
EAFL EF90B1 2069  JMP  K63  ; TRANSLATE THAT
EAFL 7450 2070

1999  ;----- NOT IN ALTERNATE SHIFT
EAFL 2071
EAFL 6A05700004 2072  TEST  EB,FLAG_CTL_SHIFT  ; ARE WE IN CONTROL SHIFT
EAFL 7450 2073  JZ  K44  ; NO-CTL-SHIFT

1999  ;----- CONTROL SHIFT, TEST SPECIAL CHARACTERS
EAFL 2074
EAFL 3C46 2075  CMP  AL,SCROLL_KEY  ; TEST FOR BREAK
EB09 7518 2076  JNE  K39  ; NO-BREAK
EB09 881E0000 2077  MOV  BK,BUFFER_START  ; RESET BUFFER TO EMPTY
EB09 89E1A00 2078  MOV  BUFFER_HEAD,BX  ;
EB09 89E1C00 2079  MOV  BUFFER_TAIL,BX  ;
EB09 C506710800 2080  MOV  BIOS_BREAK,6OH  ; TURN ON BIOS_BREAK BIT
EB09 CD05 2081  INT  10H  ; BREAK INTERRUPT VECTOR
EB09 280C 2082  SUB  AX,AX  ; PUT OUT DUMMY CHARACTER
EB09 980000 2083  JMP  K57  ; BUFFER_FILL
EB09 980000 2084  JMP  K57  ; BUFFER_FILL

1999  ;----- BREAK AND PAUSE KEYS
EB09 3C45 2085  CMP  AL,HEM_KEY  ; LOOK FOR PAUSE KEY
EB27 7521 2086  JNE  K61  ; NO-PAUSE
EB29 8006100006 2087  OR  KB,FLAG_1,HOLD_STATE  ; TURN ON THE HOLD FLAG
EB29 8020 2088  MOV  AL,E01  ; END OF INTERRUPT TO CONTROL PORT
EB30 E620 2089  OUT  020H,AL  ; ALLOW FURTHER KEYSTROKE INTO

1999

1999  ;----- DURING PAUSE INTERVAL, TURN CRT BACK ON

1999

EB32 603E940007 2090  MOV  AL,0  ; IS THIS BLACK AND WHITE CARD
EB32 9D00000007 2091  CMP  CRT_MODE,7  ; IS THIS BLACK AND WHITE CARD

System BIOS  A-31
A-32 System BIOS
<table>
<thead>
<tr>
<th>LOC OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>EB49 B074E9</td>
<td>2100</td>
<td>MOV BX,OFFSET K15</td>
</tr>
<tr>
<td>EB4C EB71</td>
<td>2101</td>
<td>JMP SHORT K64</td>
</tr>
<tr>
<td>EBAE B02D4A</td>
<td>2102</td>
<td>K58:</td>
</tr>
<tr>
<td>EB4E EB0E</td>
<td>2103</td>
<td>MOV AX, 74925566<code>-</code></td>
</tr>
<tr>
<td>EB44 EB04</td>
<td>2104</td>
<td>JMP SHORT K57</td>
</tr>
<tr>
<td>EB53 B0B64E04</td>
<td>2105</td>
<td>K51:</td>
</tr>
<tr>
<td>EB46 EB10D</td>
<td>2106</td>
<td>MOV AX, 7992566<code>+</code></td>
</tr>
<tr>
<td>EB0D EB10D</td>
<td>2107</td>
<td>JMP SHORT K57</td>
</tr>
<tr>
<td>EB48</td>
<td>2108</td>
<td>----- MIGHT BE NAM LOCK, TEST SHIFT STATUS</td>
</tr>
<tr>
<td>EBBA</td>
<td>2109</td>
<td>KS2:</td>
</tr>
<tr>
<td>EB50 5404170003</td>
<td>2110</td>
<td>TEST KB_FLAG,LEFT_SHIFT-RIGHT_SHIFT</td>
</tr>
<tr>
<td>EB50 75E0</td>
<td>2111</td>
<td>JNZ K49</td>
</tr>
<tr>
<td>EB5F</td>
<td>2112</td>
<td>K53:</td>
</tr>
<tr>
<td>EBD6 E046</td>
<td>2113</td>
<td>SUB AL,170</td>
</tr>
<tr>
<td>EBD1 EB04</td>
<td>2114</td>
<td>MOV BX,OFFSET K14</td>
</tr>
<tr>
<td>EBD4 EB0B</td>
<td>2115</td>
<td>JMP SHORT K56</td>
</tr>
<tr>
<td>EB49</td>
<td>2116</td>
<td>----- FLASH OLD LOWER CASE</td>
</tr>
<tr>
<td>EB46</td>
<td>2117</td>
<td>K5A:</td>
</tr>
<tr>
<td>EB46 3C30</td>
<td>2118</td>
<td>CMP AL,59</td>
</tr>
<tr>
<td>EB4C 7204</td>
<td>2119</td>
<td>JB K55</td>
</tr>
<tr>
<td>EBCC B00B</td>
<td>2120</td>
<td>MOV AL,0</td>
</tr>
<tr>
<td>EBD5 EB07</td>
<td>2121</td>
<td>JMP SHORT K57</td>
</tr>
<tr>
<td>EBCC</td>
<td>2122</td>
<td>K55:</td>
</tr>
<tr>
<td>EBD6 EB01EB</td>
<td>2123</td>
<td>MOV BX,OFFSET K10</td>
</tr>
<tr>
<td>EB08</td>
<td>2124</td>
<td>----- TRANSLATE THE CHARACTER</td>
</tr>
<tr>
<td>EB01</td>
<td>2125</td>
<td>K56:</td>
</tr>
<tr>
<td>EB01 FEB0</td>
<td>2126</td>
<td>DEC AL</td>
</tr>
<tr>
<td>EB03 2E07</td>
<td>2127</td>
<td>XLAT CBX,K11</td>
</tr>
<tr>
<td>EB11</td>
<td>2128</td>
<td>----- PUT CHARACTER INTO BUFFER</td>
</tr>
<tr>
<td>EB16</td>
<td>2129</td>
<td>K57:</td>
</tr>
<tr>
<td>EB05 3CFF</td>
<td>2130</td>
<td>CMP AL,-1</td>
</tr>
<tr>
<td>EBD6 741F</td>
<td>2131</td>
<td>JE K59</td>
</tr>
<tr>
<td>EB09 BE0CFF</td>
<td>2132</td>
<td>CMP AH,-1</td>
</tr>
<tr>
<td>EBD6 741A</td>
<td>2133</td>
<td>JE K59</td>
</tr>
<tr>
<td>EB0E</td>
<td>2134</td>
<td>----- HANDLE THE CAPS LOCK PROBLEM</td>
</tr>
<tr>
<td>EB0E</td>
<td>2135</td>
<td>K58:</td>
</tr>
<tr>
<td>EBD6 506170040</td>
<td>2136</td>
<td>TEST KB_FLAG,CAPS_STATE</td>
</tr>
<tr>
<td>EB4E 7420</td>
<td>2137</td>
<td>JZ K51</td>
</tr>
<tr>
<td>EB28</td>
<td>2138</td>
<td>----- IN CAPS LOCK STATE</td>
</tr>
<tr>
<td>EB55 506170003</td>
<td>2139</td>
<td>TEST KB_FLAG,LEFT_SHIFT-RIGHT_SHIFT</td>
</tr>
<tr>
<td>EB4E 740F</td>
<td>2140</td>
<td>JZ K60</td>
</tr>
<tr>
<td>EB23</td>
<td>2141</td>
<td>----- CONVERT ANY UPPER CASE TO LOWER CASE</td>
</tr>
<tr>
<td>EB4E 3C41</td>
<td>2142</td>
<td>CMP AL,'A'</td>
</tr>
<tr>
<td>EBEE 7215</td>
<td>2143</td>
<td>JB K61</td>
</tr>
<tr>
<td>EBFD 3C5A</td>
<td>2144</td>
<td>CMP AL,'Z'</td>
</tr>
<tr>
<td>EB6F 7711</td>
<td>2145</td>
<td>JA K61</td>
</tr>
<tr>
<td>EB66 5060</td>
<td>2146</td>
<td>ADD AL,'a'-'A'</td>
</tr>
<tr>
<td>EB4E 5060</td>
<td>2147</td>
<td>JMP SHORT K61</td>
</tr>
<tr>
<td>EB46 55555555</td>
<td>2148</td>
<td>K59:</td>
</tr>
<tr>
<td>EB4E 59555555</td>
<td>2149</td>
<td>JMP K26</td>
</tr>
<tr>
<td>2150</td>
<td>2151</td>
<td>----- CONVERT ANY LOWER CASE TO UPPER CASE</td>
</tr>
<tr>
<td>EB4E</td>
<td>2152</td>
<td>K60:</td>
</tr>
<tr>
<td>EBFD 3C41</td>
<td>2153</td>
<td>CMP AL,'a'</td>
</tr>
<tr>
<td>EBFD 7206</td>
<td>2154</td>
<td>JB K61</td>
</tr>
<tr>
<td>EBFD 3C7A</td>
<td>2155</td>
<td>CMP AL,'z'</td>
</tr>
<tr>
<td>EB2C 7702</td>
<td>2156</td>
<td>JA K61</td>
</tr>
<tr>
<td>EC40 3C20</td>
<td>2157</td>
<td>SUB AL,'a'-'A'</td>
</tr>
<tr>
<td>EC05 5060</td>
<td>2158</td>
<td>JMP SHORT K61</td>
</tr>
<tr>
<td>EC05 50615000</td>
<td>2159</td>
<td>MOV BX,BUFFER_TAIL</td>
</tr>
<tr>
<td>EC09 8083</td>
<td>2160</td>
<td>MOV SI,DX</td>
</tr>
<tr>
<td>EC0B EB65FC</td>
<td>2161</td>
<td>CALL K4</td>
</tr>
</tbody>
</table>

System BIOS  A-33
E0E 0811A00 2257 CHP BX,BUFFER_HEAD 1 HAS THE BUFFER WRAPPED AROUND
E12 7F45 2258 JE K62 1 BUFFER_FULL_BEEP
E14 0906 2259 MOV SI,IAK 1 STORE THE VALUE
E16 8911C00 2260 MOV BUFFER_TAIL,BX 1 MOVE THE POINTER UP
EC1A 89CCE 2261 JMP K26 1 INTERRUPT_RETURN
E262 2263
E265 1------ TRANSLATE SCAN FOR PSEUDO SCAN CODES
E266 1
E267 1
EC1D 2268 K63: 1 TRANSLATE-SCAN
EC1D EC3B 2269 SUB AL,59 1 CONVERT ORIGIN TO FUNCTION KEYS
EC1F 2267 K64: 1 TRANSLATE-SCAN-DRD
EC1F E007 2268 XLAT CS:K9
EC21 DAE0 2269 MOV AH,AL 1 CTL TABLE SCAN
EC23 8000 226A MOV AL,0 1 PUT VALUE INTO AH
EC25 E0AE 226B JMP K57 1 ZERO ASCII CODE
EC27 226C KB_INT ENDP
EC27 226D
EC27 226E KB: 1 BUFFER IS FULL, SOUND THE BEEPER
EC27 226F
EC27 2270 MOV AL,EOI 1 BUFFER-FULL-BEEP
EC27 2271 OUT IOP,AL 1 SEND COMMAND TO INT CONTROL PORT
EC2B 8B6000 2272 MOV BX,06CH 1 NUMBER OF CYCLES FOR 12/SECOND TONE
EC2E E461 2273 IN AL,KBCTL 1 GET CONTROL INFORMATION
EC30 2274 PUSH AX 1 SAVE
EC31 2275 K65: 1 DB-CYCLE
EC31 2276 E4FC 1 TURN OFF TIMER GATE AND SPEAKER DATA
EC33 E661 2277 OUT KB_CTL,AL 1 OUTPUT TO CONTROL
EC35 2278 MOV CX,48H 1 HALF CYCLE TIME FOR TONE
EC37 2279 K66: 1
EC3B 227A E0FE 1 SPEAKER OFF
EC3C E0C0 227B OR AL,2 1 TURN ON SPEAKER BIT
EC3E E661 227C OUT KB_CTL,AL 1 OUTPUT TO CONTROL
EC3E 227D MOV CX,48H 1 SET UP COUNT
EC3F 227E K67: 1
EC41 227F E0FE 1 ANOTHER HALF CYCLE
EC43 4B 2280 DEC BX 1 TOTAL TIME COUNT
EC44 75EB 2281 JNZ K65 1 DO ANOTHER CYCLE
EC46 50 2282 POP AX 1 RECOVER CONTROL
EC47 E461 2283 OUT KB_CTL,AL 1 OUTPUT THE CONTROL
EC49 E91FE 2284 JMP K27
EC4C 2285 8033031
EC50 2286 F1 DB '301',13H,10H 1 KEYBOARD ERROR
EC52 2287 8633031
EC55 2288 F3 DB '601',13H,10H 1 DISKETTE ERROR

A-34 System BIOS
DATA VARIABLE -- DISK_POINTER

OUTPUT

AN = STATUS OF OPERATION

STATUS BITS ARE DEFINED IN THE ENCODES FOR

DISKETTE_STATUS VARIABLE IN THE DATA SEGMENT OF THIS

MODULE.

CY = 0 SUCCESSFUL OPERATION (AH=0 ON RETURN)

CT = 1 FAILED OPERATION (AH HAS ERROR REASON)

FOR READ/WRITE/VERIFY

DS,DX,DX,DX,CL PRESERVED

AL = NUMBER OF SECTORS ACTUALLY READ

******* AL MAY NOT BE CORRECT IF TIME OUT ERROR OCCURS

NOTE: IF AN ERROR IS REPORTED BY THE DISKETTE CODE, THE

APPROPRIATE ACTION IS TO RESET THE DISKETTE, THEN RETRY

THE OPERATION. ON READ ACCEESSES, NO MOTOR START DELAY

IS TAKEN, SO THAT THREE RETRIES ARE REQUIRED ON READS.

TO ENSURE THAT THE PROBLEM IS NOT DUE TO MOTOR

START-UP.

ASSUME CS:CODE,DS:DATA,ES:DATA

ORG 0C50H

DISKETTE_ID PROC FAR

INTERRUPTS BACK ON

SAVE ADDRESS

SAVE SEGMENT REGISTER VALUE

SAVE ALL REGISTERS DURING OPERATION

SET UP POINTER TO README

CALL BS:4

CALL THE REST TO ENSURE DS RESTORED

SET THE MOTOR WAIT PARAMETER

SET THE TIMER COUNT FOR THE MOTOR

GET STATUS OF OPERATION

SET THE CARRY FLAG TO INDICATE

SUCCESS OR FAILURE

RESTORE ALL REGISTERS

RECOVER ADDRESS

THROW AWAY SAVED FLAGS

DISKETTE_ID ENDP

J1 PROC NEAR

SAVE # SECTORS IN DH

INDICATE A READ OPERATION

AH=0

MOV DH,AL

AND MOTOR_STATUS.07FH

OR AH,AL

JZ DISK_RESET

DEC AH

JZ DISK_STATUS

MOV DISKETTE_STATUS.0

CMP DL,4

JMP DISK_WRITE

J1:

DEC AH

JZ TEST_DISK_VERB

TEST_DISK_VERB

AH=4

DEC AH

JZ TEST_DISK_VERE

TEST_DISK_VERE

AH=5
LOC OBJ       LINE  SOURCE
---          ---  ------------------------
ECB1        2457  J3:           : B Ada_COMMAND
ECB1 C646610001  2460  MOV   DISKETTE_STATUS,BAD_CMD       : ERROR CODE, NO SECTORS TRANSFERRED
ECB6 C3      2469  RET           : UNDEFINED OPERATION
                2470  J1   ENDP
                2471
                2472
ECB7        2473  ----- RESET THE DISKETTE SYSTEM
                2474
                2475  ---- DISK_RESET PROC NEAR
                2476
                2477  MOV   DX,0320H  : ADAPTER CONTROL PORT
ECB8 A4023  2478  CLI
                2479  MOV   AL,MOTOR_STATUS  : WHICH MOTOR IS ON
                2480  MOV   CL,4  : SHIFT COUNT
ECB0 A2500  2481  SAL   AL,CL  : MOVE MOTOR VALUE TO HIGH NYTBLE
                2482  MOV   AL,20H  : SELECT CORRESPONDING DRIVE
ECB4 750C    2483  JNZ   JS  : JUMP IF MOTOR ONE IS ON
ECB6 A500    2484  TEST  AL,40H  : JUMP IF MOTOR TWO IS ON
ECB8 7506    2485  JNZ   J4  : JUMP IF MOTOR THREE IS ON
ECB6 A800    2486  TEST  AL,60H  : JUMP IF MOTOR ZERO IS ON
ECB6 FEC0    2487  INC   AL
               2488  J4:
ECB0 FE0C    2489  INC   AL
ECB2        2490  JS:               : INC AL
ECB2 FE00    2491  INC   AL
ECB4        2492  J6:               : INC AL
ECB0 DCA8    2493  OR    AL,0  : TURN ON INTERRUPT ENABLE
ECB6 EE      2494  OUT   DX,AL  : RESET THE ADAPTER
ECB7 C6653E0000  2495  MOV   SEEK_STATUS,0  : SET RECAL REQUIRES ON ALL DRIVES
ECB6 C646410000  2496  MOV   DISKETTE_STATUS,0  : SET OK STATUS FOR DISKETTE
ECB1 DCA6    2497  OR    AL,4  : TURN OFF RESET
ECB3 EE      2498  OUT   DX,AL  : TURN OFF THE RESET
ECB4 F9      2499  STI
ECB5 EB902    2500  CALL   CHK_STAT_E  : RENDABLE THE INTERRUPTS
ECB6 A9020    2501  MOV   AL,NEC_STATUS  : EVOKE ERROR RETURN AND DO OWN TEST
ECB7 AEC00    2502  CMP   AL,OCMN  : TEST FOR DRIVE READY TRANSITION
ECB6 7406    2503  JZ    J7  : EVERYTHING OK
ECB6 80001010002  2504  OR   DISKETTE_STATUS,BAD_NE  : SET ERROR CODE
ECF4 C5      2505  RET
                2506
ECF5        2507  ----- SEND SPECIFY COMMAND TO NEC
ECF5 B403    2508  J7:
ECF7 EB701    2509  MOV   AH,03H  : DRIVE READY
ECFA B0100    2510  CALL   NEC_OUTPUT  : SPECIFY COMMAND
ECFD E60C001  2511  MOV   BX,1  : OUTPUT THE COMMAND
ECF0 D000100  2512  CALL   GET_PARM  : FIRST BYTE PARM IN BLOCK
ECF0 EB60001  2513  MOV   BX,3  : SECOND BYTE PARM IN BLOCK
ECF0 C5      2514  CALL   GET_PARM  : TO THE NEC CONTROLLER
                2515  RET
ECF0 C3      2516  J8:
                2517  RET
                2518  RETURN TO CALLER
                2519
                2520
                2521
                2522
                2523
                2524
                2525
                2526
                2527
                2528
                2529
                2530
                2531
                2532
                2533
                2534
                2535
                2536
                2537
                2538
                2539
                2540
                2541
                2542
                2543
                2544
                2545
                2546
                2547
                2548
                2549
                2550
                2551
                2552
                2553
                2554
                2555
                2556
                2557
                2558
                2559
                2560
                2561
                2562
                2563
                2564
                2565
                2566
                2567
                2568
                2569
                2570
                2571
                2572
                2573
                2574
                2575
                2576
                2577
                2578
                2579
                2580
                2581
                2582
                2583
                2584
                2585
                2586
                2587
                2588
                2589
                2590
                2591
                2592
                2593
                2594
                2595
                2596
                2597
                2598
                2599
                2600
                2601
                2602
                2603
                2604
                2605
                2606
                2607
                2608
                2609
                2610
                2611
                2612
                2613
                2614
                2615
                2616
                2617
                2618
                2619
                2620
                2621
                2622
                2623
                2624
                2625
                2626
                2627
                2628
                2629
                2630
                2631
                2632
                2633
                2634
                2635
                2636
                2637
                2638
                2639
                2640
                2641
                2642
                2643
                2644
                2645
                2646
                2647
                2648
                2649
                2650
                2651
                2652
                2653
                2654
                2655
                2656
                2657
                2658
                2659
                2660
                2661
                2662
                2663
                2664
                2665
                2666
                2667
                2668
                2669
                2670
                2671
                2672
                2673
                2674
                2675
                2676
                2677
                2678
                2679
                2680
                2681
                2682
                2683

A-36  System BIOS
LOC OBJ

2404 l----- DISKETTE FORMAT
2405
2406 ED10 8803F0068 2407 PROC NEAR
2408 OR MOTOR_STATUS,80H ; INDICATE WRITE OPERATION
2409 MOV AL,04AH ; SNAIL WRITE TO DISKETTE
2410 CALL DMA_SETUP ; SET UP THE DMA
2411 MOV AH,040H ; ESTABLISH THE FORMAT COMMAND
2412 JMP SHORT RN_GRP ; DO THE OPERATION
2413
2414 J10:
2415 MOV BX,7 ; REDIRECTION OF FN_GRP FOR FN
2416 CALL GET_PARM ; BYTES/SECTOR VALUE TO NEC
2417 MOV BX,09 ; GET THE
2418 CALL GET_PARM ; SECTORS/TRACK VALUE TO NEC
2419 MOV BX,15 ; GET THE
2420 CALL GET_PARM ; GAP LENGTH VALUE TO NEC
2421 MOV BX,17 ; GET THE FILLED BYTE
2422 JMP J10 ; TO THE CONTROLLER
2423
2424 DISK_FORMAT ENDP
2425
2426 l----- DISKETTE WRITE ROUTINE
2427
2428 ED3E 8803F0068 2429 PROC NEAR
2430 OR MOTOR_STATUS,80H ; INDICATE WRITE OPERATION
2431 MOV AL,04AH ; DMA WRITE COMMAND
2432 CALL DMA_SETUP ; Stellar DMA SETUP
2433 MOV AH,3C3H ; NEC COMMAND TO WRITE TO DISKETTE
2434 DISK_WRITE ENDP
2435
2436 l----- ALLOW WRITE ROUTINE TO FALL INTO RN_GRP
2437
2438 ED49 8803F0068 2439 PROC NEAR
2440 RN_GRP PROC NEAR
2441 JC J11 ; TEST FOR DMA ERROR
2442 MOV DISKETTE_STATUS,DMA_BOUNDARY ; SET ERROR
2443 MOV AX,0 ; NO SECTORS TRANSFERRED
2444 RET ; RETURN TO MAIN ROUTINE
2445 J11:
2446 PUSH AX ; SAVE THE COMMAND
2447 POP AX ; TURN THE MOTOR AND SELECT THE DRIVE
2448
2449 ED55 51 2450 PUSH CX ; SAVE THE I/O PARMS
2451 MOV CL,DL ; GET DRIVE NUMBER AS SHIFT COUNT
2452 MOV AL,1 ; MASK FOR DETERMINING MOTOR BIT
2453 SAL AL,CL ; SHIFT THE MASK BIT
2454 CLI ; NO INTERRUPTS WHILE DETERMINING
2455 MOV MOTOR_COUNT,0FFH ; START COUNTER FOR DURING OPERATING
2456 TEST AL,MOTOR_STATUS ; TEST THAT MOTOR IS OPERATING
2457 J14 ; IF M0,00, SKIP THE WAIT
2458 XOR MOTOR_STATUS,0FH ; CLEAR ALL MOTOR BITS
2459 OR MOTOR_STATUS,AL ; TURN ON THE CURRENT MOTOR
2460 FB 2461 STI ; INTERRUPTS BACK ON
2462 MOV AL,00H ; MASK BIT
2463 SAL AL,CL ; DEVELOP BIT MASK FOR MOTOR ENABLE
2464 OR AL,DL ; GET DRIVE SELECT BITS IN
2465 OR AL,08H ; NO RESET, ENABLE DMA/INT
2466 PUSH DX ; SAVE REG
2467 MOV DX,DXshell ; CONTROL PORT ADDRESS
2468 OUT DX,AL ; RECOVER REGISTERS
2469
2470 l----- WAIT FOR MOTOR IF WRITE OPERATION
2471
2472 ED00 8803F0068 2473 PROC NEAR
2474 MOV AL,0AH ; IS THIS A WRITE
2475 JE J14 ; NO, CONTINUE WITHOUT WAIT
2476 MOV BX,20 ; GET THE MOTOR WAIT
2477 CALL GET_PARM ; PARAMETER
2478 OR AH,AH ; TEST FOR NO WAIT
2479 J12:
2480 MOV CX,0 ; EXIT WITH TIME EXPIRED
2481 SUB CX,CX ; SET UP 1/8 SECOND LOOP TIME
2482 J13:
2483 LOOP J13 ; WAIT FOR THE REQUIRED TIME
2484
DEC AH    ; DECREMENT TIME VALUE
JMP J1E   ; ARE WE DONE YET
J14: MOV BH,AN  ; MOTOR_RUNNING
STI       ; INTERRUPTS BACK ON FOR BYPASS WAIT
POP CX    

----- DO THE SEEK OPERATION-----
CALL SEEK  ; MOVE TO CORRECT TRACK
POP AX     ; RECOVER COMMAND
MOV BH,AN  ; SAVE COMMAND IN BH
MOV DH,0   ; SET NO SECTORS READ IN CASE OF ERROR
JC J17    ; IF ERROR, THEN EXIT AFTER MOTOR OFF
MOV $1,OFFSET J17  ; DUMMY RETURN ON STACK FOR NEC_OUTPUT
PUSH SI    ; SO THAT IT WILL RETURN TO MOTOR OFF
MOV AX,83  ; LOCATION

----- SEND OUT THE PARAMETERS TO THE CONTROLLER-----
CALL NEC_OUTPUT  ; OUTPUT THE OPERATION COMMAND
MOV AH,BP+11  ; GET THE CURRENT HEAD NUMBER
MOV AH,1      ; MOVE IT TO BIT 2
MOV AH,1      ; ISOLATE THAT BIT
AND AH,4      ; OR IN THE DRIVE NUMBER
OR AH,DL      
CALL NEC_OUTPUT  ; OR IN THE DRIVE NUMBER

----- TEST FOR FORMAT COMMAND-----
CMP BM,04EH  ; IS THIS A FORMAT OPERATION
JNE J15      ; NO, CONTINUE WITH R/W/V
JMP J10      ; IF SO, HANDLE SPECIAL
JC J15:

MOV AH,CH    ; CYLINDER NUMBER
CALL NEC_OUTPUT  ; HEAD NUMBER FROM STACK
MOV AH,CL    ; SECTOR NUMBER
MOV AH,BP+11  ; BYTES/SECTOR PARM FROM BLOCK
MOV AX,7      ; TO THE NEC
MOV BX,9      ; EOT PARM FROM BLOCK
MOV AH,CL    ; TO THE NEC
MOV AH,BP+11  ; GAP LENGTH PARM FROM BLOCK
MOV AX,15     ; TO THE NEC
MOV AX,13     ; DLT PARM FROM BLOCK
MOV AX,13     ; RM_OK_FROM
CALL GET_PARM  ; TO THE NEC
CALL GET_PARM  ; CAN NOW DISCARD THAT DUMMY
POP SI        ; RETURN ADDRESS

----- LET THE OPERATION HAPPEN-----
CALL WAIT_INT  ; WAIT FOR THE INTERRUPT
J17: MOV J2I   ; MOTOR_OFF
JC J2I         ; LOOK FOR ERROR
JC J2I         ; LOOK FOR ERROR

----- CHECK THE RESULTS RETURNED BY THE CONTROLLER-----
CLD           ; SET THE CORRECT DIRECTION
MOV SI,OFFSET NEC_STATUS  ; POINT TO STATUS FIELD
LDS NEC_STATUS  ; GET STO
AND AL,0C8H    ; TEST FOR NORMAL TERMINATION
JZ J2I         ; CMP AL,040H  ; TEST FOR ABNORMAL TERMINATION
JNZ J16        ; NOT ABNORMAL, BAD NEC

----- ABNORMAL TERMINATION, FIND OUT WHY-----
LDS NEC_STATUS  ; GET STI
SAL AL+1       ; TEST FOR EOT FOUND
MOV AH,RECORD_HOT_END
JC J19         ; RM_FAIL
SAL AL+1       ; TEST FOR CRC ERROR
EEEF B410
EE11 721C
EE13 DO00
EE15 B400
EE17 7216
EE19 DO00
EE1B DO00
EE1D B404
EE1F 720E
EE21 DO00
EE23 B403
EE25 720B
EE27 DO00
EE29 B402
EE2B 7202

MOV AH, BAD_CRC
J C J19
J C J19
MOV AH, BAD_DNAS
J C J19
MOV AH, BAD_DNAS
J C J19
MOV AH, BAD_DNAS
J C J19
MOV AH, BAD_DNAS
J C J19
MOV AH, BAD_DNAS
J C J19

------ NEC MUST HAVE FAILED

EE2D
EE2D B400
EE2F
EE2F 0264100
EE33 E87041
EE36
EE36 C3
EE37
EE37 E87001
EE3A C3

MOV AH, BAD_NEC
J C J19
OR DISKETTE_STATUS, AH
CALL HAM_TRANS
RET
CALL RESULTS
RET

------ OPERATION HAS SUCCESSFUL

EE3B
EE3B E87001
EE3E 3264
EE40 C3

CALL HAM_TRANS
XOR AH, AH
RET

RH_DPM ENDP

EE41
EE41 32
EE42 B1
EE43 EAF503
EE45 E970C9
EE46 02641000
EE47 E40 EC
EE49 AA04
EE4B 740C
EE4D 8F99
EE4F

PUSH OX
PUSH CX
MOV DX, 03FH
XOR CX, CX
IN AL, OX
TEST AL, OAH
JZ JE8
LOOP JE3

NEC_OUTPUT PROC NEAR

SAVE REGISTERS
STATUS PORT
COUNT FOR TIME OUT
GET STATUS
TEST DIRECTION BIT
DIRECTION OK
TIME_ERROR

DISKETTE_STATUS, TIME OUT
POP CX
POP AX
STC
RET

XOR CX, CX
RESET THE COUNT

System BIOS A-39
LOC OBJ LINE SOURCE

EEX8 EB 2715 JMP J24 1 ERROR CONDITION
EEX4 2716 J271: 1 OUTPUT
EEX4 16A4 2717 MDV AL,AH 1 GET BYTE TO OUTPUT
EEX4 2718 MDV DL,0FH 1 DATA PORT (3F5)
EEX6 EE 2719 OUT DX,AL 1 OUTPUT THE BYTE
EEX9 E9 2720 POP CX 1 RECOVER REGISTERS
EEXA BA 2721 RET 1 CT = 0 FROM TEST INSTRUCTION
EEXB C3 2722

2724 HE OUTPUT ENDH 1

2725 GET_PARAM 1

2726 THIS ROUTINE FETCHES THE INDEXED POINTER FROM THE DISK_BASE 1
2727 BLOCK POINTED AT BY THE DATA VARIABLE DISK_POINTER. A BYTE FROM 1
2728 THAT TABLE IS THEN MOVED INTO AH, THE INDEX OF THAT BYTE BEING 1
2729 THE PARAM IN BX 1
2730 ENTRY -- 1
2731 BX = INDEX OF BYTE TO BE.FETCHED = 2
2732 IF THE LOW BIT OF BX IS ON, THE BYTE IS IMMEDIATELY OUTPUT 1
2733 TO THE NEC CONTROLLER 1
2734 EXIT -- 1
2735 AH = THAT BYTE FROM BLOCK 1
2736

EEXC 2737 GET_PARAM PROC NEAR
EEX4 1C 2738 PUSH DS 1 SAVE SEGMENT
EEX6 2BC0 2739 SUB AX,AX 1 ZERO TO AX
EEX6 9E8B 2740 MDV DS,AX 1
2741 ASSUME DS:ADE0 1

EEX7 C8378000 2742 LDS SS,GDISK_POINTER 1 POINT TO BLOCK
2743 SHR BX,1 1 DIVIDE BX BY 2, AND SET FLAG 1
2744 FOR EXIT 1

EEX7 B20D 2745 MDV AH,SI+BX 1 GET THE WORD 1

EEX7 1F 2746 POP DS 1 RESTORE SEGMENT
2747 ASSUME DS:DATA 1

EEX7 72C5 2748 JC NE_OUTPUT 1 IF FLAG SET, OUTPUT TO CONTROLLER 1
2749 RET 1 RETURN TO CALLER
2750 GET_PARAM-ENDP 1

2752 SEEK 1

2753 THIS ROUTINE WILL MOVE THE HEAD ON THE NAMED DRIVE TO THE 1
2754 NAMED TRACK. IF THE DRIVE HAS NOT BEEN ACCESSED SINCE THE 1
2755 DRIVE RESET COMMAND HAS BEEN ISSUED, THE DRIVE WILL BE RECALIBRATED. 1
2756 INPUT 1
2757 (DL) = DRIVE TO SEEK ON 1
2758 (CH) = TRACK TO SEEK TO 1
2759 OUTPUT 1
2760 CY = 0 SUCCESS 1
2761 CY = 1 FAILURE -- DISKETTE STATUS SET ACCORDINGLY 1
2762 A13 DESTROYED 1
2763

EEX9 2764 SEEK PROC NEAR
EEX7 B001 2765 MDV AL,1 1 ESTABLISH MASK FOR RECAL TEST
EEX7 51 2766 PUSH CX 1 SAVE INPUT VALUES
2767 MDV CL,DL 1 GET DRIVE VALUE INTO CL
2768 ROL AL,CL 1 SHIFT IT BY THE DRIVE VALUE
2769 POP CX 1 RECOVER TRACK VALUE

EEX9 00632000 2770 TEST AL,SEEK_STATUS 1 TEST FOR RECAL REQUIRED
2771 JNC J2B 1 NO-RECAL 1
2772 OR SEEK_STATUS,AL 1 TURN ON THE ND RECAL BIT IN FLAG

EEXF DA07 2773 MDV AH,0TH 1 RECALIBRATE COMMAND
EEX1 E00F 2774 CALL NE_OUTPUT 1
EEX9 8AE2 2775 MDV AH,DL 1
EEX9 E00F 2776 CALL NEC_OUTPUT 1 OUTPUT THE DRIVE NUMBER
2777 CALL CHK_STAT,2 1 GET THE INTERRUPT AND SENSE INT STATUS
2778 JC J32 1 SEEK_ERROR
2779
2780 ----> DRIVE IS IN SYNCH WITH CONTROLLER, SEEK TO TRACK
2781

EEX0 2782 J2B: 1

EEX7 DA0F 2783 MDV AH,0TH 1 SEEK COMMAND TO NEC
EEXA E04F 2784 CALL NEC_OUTPUT 1
EEXA AE3 2785 MDV AH,DL 1 DRIVE NUMBER
EEXA E04F 2786 CALL NEC_OUTPUT 1
EEXA 8AE5 2787 MDV AH,CH 1 TRACK NUMBER
EEXA E04F 2788 CALL NEC_OUTPUT 1 GET ENDING INTERRUPT AND 1
2789 MODE NUMBER
2790 SEEK_ERROR 1

2791

A-40 System BIOS
I------ WAIT FOR HEAD SETTLE

2792  2793  2794  2795
EE0 9C  EEB D8200  EEB E005FF  EB7 7877  EB8 7298  J28:
EE8 92602  2796  CALL GET_PART  2797  PUSH CX  2798  HEAD_SETTLE
EEB 8AE4  2799  MOV CX,550  279A  1 MS LOOP
EEE 7406  279B  OR AH,AM  279C  TEST FOR TIME EXPIRED
EEF 0002  279D  JZ J31  279E  J30:
EEF 0003  279F  LOOP J30  2800  DELAY FOR 1 MS
EEC 1004  2801  DEC AH  2802  DECREMENT THE COUNT
EEC 1005  2803  JMP J29  2804  DO IT SOME MORE
EEC 7506  2805  J31:
EEC 59  2806  POP CX  2807  RECOVER STATE
EEC 60  2808  POPF
EEC 7509  2809  J32:  280A  SEEK_ERROR
EEC 750A  280B  RET  280C  RETURN TO CALLER
EEC 750E
2811  SEEK ENDP

2812  2813  2814  2815
EEC 8201  2816  DMA_SETUP  2817  THIS ROUTINE SETS UP THE DMA FOR READ/WRITE/VERIFY OPERATIONS.
EEC 8202  2818  INPUT
EEC 8203  2819  (AL) = MODE BYTE FOR THE DMA
EEC 8204  2820  (ES:BX) - ADDRESS TO READ/WRITE THE DATA
EEC 8205  2821  OUTPUT
EEC 8206  2822  (AX) DESTROYED
EEC 8207  2823

2824  2825  2826
EEC 8251  2827  DMA_SETUP  PROC NEAR
EEC 8252  2828  PUSH CX  2829  SAVE THE REGISTER
EEC 8253  282A  CLI  282B  NO MORE INTERRUPTS
EEC 8254  282C  OUT DMA+12,AL  282D  SET THE FIRST/LAST F/F
EEC 8255  282E  PUSH AX  282F  OUTPUT THE MODE BYTE
EEC 8256  2830  POP AX  2831  OUTPUT
EEC 8257  2832  OUT DMA+12,AL  2833  GET THE MODE BYTE
EEC 8258  2834  MOV AX,ES  2835  GET THE ES VALUE
EEC 8259  2836  MOV CL,4  2837  SHIFT COUNT
EEC 825A  2838  ROL AX,CL  2839  ROTATE LEFT
EEC 825B  283A  MOV CH,AL  283B  GET HIGHEST BYTE OF ES TO CH
EEC 825C  283C  AND AL,0F0H  283D  ZERO THE LOW BYTE OF SEGMENT
EEC 825D  283E  ADD AX,DX  283F  TEST FOR CARRY FROM ADDITION
EEC 825E  2840  INC CX  2841  CARRY MEANS HIGH 4 BITS MUST BE INC
EEC 825F  2842  J33:
EE 8260  2843  PUSH AX  2844  SAVE START ADDRESS
EE 8261  2845  OUT DMA+4,AL  2846  OUTPUT LOW ADDRESS
EE 8262  2847  MOV AL,AL  2848  OUTPUT HIGH ADDRESS
EE 8263  2849  OUT DMA+5,AL  284A  GET HIGH 4 BITS
EE 8264  284B  MOV AL,CH  284C  OUTPUT THE HIGH 4 BITS TO
EE 8265  284D  AND AL,OFH  284E  THE PAGE REGISTER
EE 8266
284F  2850  2851
EE 8267  2852  MOV AH,BH  2853  NUMBER OF SECTORS
EE 8268  2854  MOV AL,DL  2855  TIMES 256 INTO AX
EE 8269  2856  SHR AX,1  2857  SECTORS = 128 INTO AX
EE 826A  2858  MOV AX,0  2859  GET THE BYTES/SECTOR PARAM
EE 826B  285A  MOV BX,6  285B  USE AS SHIFT COUNT (0-128, 1-256 ETC)
EE 826C  285C  CALL GET_PART  285D  GET THE BYTES/SECTOR PARAM
EE 826D  285E  MOV CL,AL  285F  GET THE BYTES/SECTOR PARAM
EE 826E  2860  POP AX  2861  GET THE BYTES/SECTOR PARAM
EE 826F  2862  SHL AX,CL  2863  MULTIPLY BY CORRECT AMOUNT
EE 8270  2864  MOV AL,AL  2865  1 FOR DMA VALUE
EE 8271  2866  MOV AX,DX  2867  SAVE COUNT VALUE
EE 8272  2868  MOV AX,AX  2869  LOW BYTE OF COUNT
EE 8273  286A  MOV AL,AM  286B  HIGH BYTE OF COUNT
EE 8274  286C  MOV AL,AM  286D  HIGH BYTE OF COUNT
EE 8275  286E  MOV AL,CH  286F  INTERRUPTS BACK ON
EE 8276  2870  MOV CL,CM  2871  RECOVER COUNT VALUE
EE 8277  2872  MOV AX,AX  2873  RECOVER ADDRESS VALUE
EE 8278  2874  MOV AX,AX  2875  ADD, TEST FOR 644 OVERFLOW
EE 8279  2876  MOV AX,0F  2877  RECOVER REGISTER
EE 827A  2878  MOV AX,AM  2879  MODE FOR 8237
EE 827B  287A  MOV AX,10,AL  287B  INITIALIZE THE DISKETTE CHANNEL
<table>
<thead>
<tr>
<th>Location</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2849</td>
<td>C3</td>
<td>RET</td>
</tr>
<tr>
<td>2850</td>
<td></td>
<td>RETURN TO CALLER;</td>
</tr>
<tr>
<td>2851</td>
<td></td>
<td>CFL SET BY ABOVE IF ERROR</td>
</tr>
<tr>
<td>2852</td>
<td></td>
<td>--------------</td>
</tr>
<tr>
<td>2853</td>
<td></td>
<td>-----------------------------------</td>
</tr>
<tr>
<td>2854</td>
<td></td>
<td>CHK_STAT_2</td>
</tr>
<tr>
<td>2855</td>
<td></td>
<td>THIS ROUTINE HANDLES THE INTERRUPT RECEIVED AFTER A</td>
</tr>
<tr>
<td>2856</td>
<td></td>
<td>RECALIBRATE, SEEK, OR RESET TO THE ADAPTER.</td>
</tr>
<tr>
<td>2857</td>
<td></td>
<td>THE INTERRUPT IS WAITED FOR, THE INTERRUPT STATUS SENSED,</td>
</tr>
<tr>
<td>2858</td>
<td></td>
<td>AND THE RESULT RETURNED TO THE CALLER.</td>
</tr>
<tr>
<td>2859</td>
<td></td>
<td>INPUT</td>
</tr>
<tr>
<td>2860</td>
<td></td>
<td>NONE</td>
</tr>
<tr>
<td>2861</td>
<td></td>
<td>OUTPUT</td>
</tr>
<tr>
<td>2862</td>
<td></td>
<td>CY = 0 SUCCESS</td>
</tr>
<tr>
<td>2863</td>
<td></td>
<td>CY = 1 FAILURE -- ERROR IS IN DISKETTE_STATUS</td>
</tr>
<tr>
<td>2864</td>
<td></td>
<td>I AXI DESTROYED</td>
</tr>
<tr>
<td>2888</td>
<td>C3</td>
<td>CHK_STAT_2</td>
</tr>
<tr>
<td>2889</td>
<td></td>
<td>PROC N hormonal</td>
</tr>
<tr>
<td>2890</td>
<td></td>
<td>CALL WAIT_INT</td>
</tr>
<tr>
<td>2891</td>
<td></td>
<td>WAIT FOR THE INTERRUPT</td>
</tr>
<tr>
<td>2892</td>
<td></td>
<td>JC J34</td>
</tr>
<tr>
<td>2893</td>
<td></td>
<td>IF ERROR, RETURN IT</td>
</tr>
<tr>
<td>2894</td>
<td></td>
<td>MOV AN.0BH</td>
</tr>
<tr>
<td>2895</td>
<td></td>
<td>SENSE INTERRUPT STATUS COMMAND</td>
</tr>
<tr>
<td>2896</td>
<td></td>
<td>CALL NEC_OUTPUT</td>
</tr>
<tr>
<td>2897</td>
<td></td>
<td>CALL RESULTS</td>
</tr>
<tr>
<td>2898</td>
<td></td>
<td>READ IN THE RESULTS</td>
</tr>
<tr>
<td>2899</td>
<td></td>
<td>JC J34</td>
</tr>
<tr>
<td>2900</td>
<td></td>
<td>CHK2_RETURN</td>
</tr>
<tr>
<td>2901</td>
<td></td>
<td>MOV AL.NEC_STATUS</td>
</tr>
<tr>
<td>2902</td>
<td></td>
<td>GET THE FIRST STATUS BYTE</td>
</tr>
<tr>
<td>2903</td>
<td></td>
<td>AND AL.0BH</td>
</tr>
<tr>
<td>2904</td>
<td></td>
<td>ISOLATE THE BITS</td>
</tr>
<tr>
<td>2905</td>
<td></td>
<td>CHX AL.0BH</td>
</tr>
<tr>
<td>2906</td>
<td></td>
<td>TEST FOR CORRECT VALUE</td>
</tr>
<tr>
<td>2907</td>
<td></td>
<td>JE J35</td>
</tr>
<tr>
<td>2908</td>
<td></td>
<td>IF ERROR, GO BACK IT</td>
</tr>
<tr>
<td>2909</td>
<td></td>
<td>CLC</td>
</tr>
<tr>
<td>2910</td>
<td></td>
<td>GOOD RETURN</td>
</tr>
<tr>
<td>2911</td>
<td></td>
<td>J34:</td>
</tr>
<tr>
<td>2912</td>
<td></td>
<td>RET</td>
</tr>
<tr>
<td>2913</td>
<td></td>
<td>RETURN TO CALLER</td>
</tr>
<tr>
<td>2914</td>
<td></td>
<td>CHK2_ERROR</td>
</tr>
<tr>
<td>2915</td>
<td></td>
<td>J35:</td>
</tr>
<tr>
<td>2916</td>
<td></td>
<td>OR DISKETTE_STATUS.BAD_SEEK</td>
</tr>
<tr>
<td>2917</td>
<td></td>
<td>STC</td>
</tr>
<tr>
<td>2918</td>
<td></td>
<td>ERROR RETURN CODE</td>
</tr>
<tr>
<td>2919</td>
<td></td>
<td>RET</td>
</tr>
<tr>
<td>2920</td>
<td></td>
<td>CHK_STAT_2</td>
</tr>
<tr>
<td>2921</td>
<td></td>
<td>PROC N hormonal</td>
</tr>
<tr>
<td>2922</td>
<td></td>
<td>WAIT_INT</td>
</tr>
<tr>
<td>2923</td>
<td></td>
<td>THIS ROUTINE WAITS FOR AN INTERRUPT TO OCCUR. A TIME OUT</td>
</tr>
<tr>
<td>2924</td>
<td></td>
<td>ROUTINE TAKES PLACE DURING THE WAIT, SO THAT AN ERROR MAY BE</td>
</tr>
<tr>
<td>2925</td>
<td></td>
<td>RETURNED IF THE DRIVE IS NOT READY.</td>
</tr>
<tr>
<td>2926</td>
<td></td>
<td>INPUT</td>
</tr>
<tr>
<td>2927</td>
<td></td>
<td>NONE</td>
</tr>
<tr>
<td>2928</td>
<td></td>
<td>OUTPUT</td>
</tr>
<tr>
<td>2929</td>
<td></td>
<td>CY = 0 SUCCESS</td>
</tr>
<tr>
<td>2930</td>
<td></td>
<td>CY = 1 FAILURE -- DISKETTE_STATUS IS SET ACCORDINGLY</td>
</tr>
<tr>
<td>2931</td>
<td></td>
<td>I AXI DESTROYED</td>
</tr>
<tr>
<td>2917</td>
<td>C3</td>
<td>F33</td>
</tr>
<tr>
<td>2918</td>
<td></td>
<td>WAIT_INT</td>
</tr>
<tr>
<td>2919</td>
<td></td>
<td>PROC N hormonal</td>
</tr>
<tr>
<td>2920</td>
<td></td>
<td>BIT1</td>
</tr>
<tr>
<td>2921</td>
<td></td>
<td>TURN ON INTERRUPTS, JUST IN CASE</td>
</tr>
<tr>
<td>2922</td>
<td></td>
<td>PUSH BX</td>
</tr>
<tr>
<td>2923</td>
<td></td>
<td>SAVE REGISTERS</td>
</tr>
<tr>
<td>2924</td>
<td></td>
<td>PUSH CX</td>
</tr>
<tr>
<td>2925</td>
<td></td>
<td>CLEAR THE COUNTERS</td>
</tr>
<tr>
<td>2926</td>
<td></td>
<td>NOV BL.2</td>
</tr>
<tr>
<td>2927</td>
<td></td>
<td>FOR 2 SECOND WAIT</td>
</tr>
<tr>
<td>2928</td>
<td></td>
<td>J36:</td>
</tr>
<tr>
<td>2929</td>
<td></td>
<td>TEST SEEK_STATUS.INT_FLAG</td>
</tr>
<tr>
<td>2930</td>
<td></td>
<td>TEST FOR INTERRUPT OCCURRING</td>
</tr>
<tr>
<td>2931</td>
<td></td>
<td>ANZ J37</td>
</tr>
<tr>
<td>2932</td>
<td></td>
<td>LOOP J36</td>
</tr>
<tr>
<td>2933</td>
<td></td>
<td>COUNT DOWN WHILE WAITING</td>
</tr>
<tr>
<td>2934</td>
<td></td>
<td>DEC BL</td>
</tr>
<tr>
<td>2935</td>
<td></td>
<td>SECOND LEVEL COUNTER</td>
</tr>
<tr>
<td>2936</td>
<td></td>
<td>JNZ J36</td>
</tr>
<tr>
<td>2937</td>
<td></td>
<td>OR DISKETTE_STATUS.TIME_OUT</td>
</tr>
<tr>
<td>2938</td>
<td></td>
<td>NOTHING HAPPENED</td>
</tr>
<tr>
<td>2939</td>
<td></td>
<td>STC</td>
</tr>
<tr>
<td>2940</td>
<td></td>
<td>ERROR RETURN</td>
</tr>
<tr>
<td>2941</td>
<td></td>
<td>J37:</td>
</tr>
<tr>
<td>2942</td>
<td></td>
<td>PUSHF</td>
</tr>
<tr>
<td>2943</td>
<td></td>
<td>SAVE CURRENT CARRY</td>
</tr>
<tr>
<td>2944</td>
<td></td>
<td>SEEK_STATUS.INT_FLAG</td>
</tr>
<tr>
<td>2945</td>
<td></td>
<td>TURN OFF INTERRUPT FLAG</td>
</tr>
<tr>
<td>2946</td>
<td></td>
<td>POPF</td>
</tr>
<tr>
<td>2947</td>
<td></td>
<td>RECOVER CARRY</td>
</tr>
<tr>
<td>2948</td>
<td></td>
<td>SFR.5V</td>
</tr>
<tr>
<td>2949</td>
<td></td>
<td>POP CX</td>
</tr>
<tr>
<td>2950</td>
<td></td>
<td>RECOVER REGISTERS</td>
</tr>
<tr>
<td>2951</td>
<td></td>
<td>POP BX</td>
</tr>
<tr>
<td>2952</td>
<td></td>
<td>GOOD RETURN CODE COMES</td>
</tr>
<tr>
<td>2953</td>
<td></td>
<td>RET</td>
</tr>
<tr>
<td>2954</td>
<td></td>
<td>FROM TEST INT</td>
</tr>
</tbody>
</table>

A-42 System BIOS
EF57  2947  DBG  DEFSTK
EF57  2948  DISK_INT  PROC  FAR
EF57  2949  STI  RE_ENABLE_INTURPTS
EF57  2950  PUSH  DS
EF57  2951  PUSH  AX
EF57  2952  CALL  DOS
EF57  2953  OR  SEEK_STATUS_INT_FLAG
EF57  2954  MOV  AL,2EH  END_OF_INTERRUPT_MARKER
EF57  2955  OUT  00H,AL  INTERRUPT_CONTROL_PORT
EF57  2956  POP  AX
EF57  2957  POP  DS  RECOVER_SYSTEM
EF57  2958  IRET  RETURN_FROM_INTERRUPT
EF57  2959  DISK_INT  ENDP
EF57  2960
EF57  2961  1  RESULTS
EF57  2962  1  THIS_ROUTINE_WILL_READ_ANYTHING_THAT_THE_NEC_CONTROLLER_HAS
EF57  2963  1  TO_SAY_FOLLOWING_AN_INTERRUPT.
EF57  2964  1  INPUT
EF57  2965  1  NONE
EF57  2966  1  OUTPUT
EF57  2967  1  CY = 0 SUCCESSFUL_TRANSFER
EF57  2968  1  CY = 1 FAILURE -- TIME_OUT_IN_WAITING_FOR_STATUS
EF57  2969  1  NEC_STATUS_AREA_HAS_STATUS_BYTE_LOADED_INTO_IT
EF57  2970  1  (AH)_DESTROYED
EF57  2971
EF57  2972  RESULTS_PROC_63C9
EF57  2973  CLD
EF57  2974  MOV  DI,OFFSET_NEC_STATUS  POINTER_TO_DATA_AREA
EF57  2975  XCH  CX,DX  SAVE_COUNTER
EF57  2976  PUSH  DX
EF57  2977  PUSH  BX
EF57  2978  MOV  BL,7  MAX_STATUS_BYTES
EF57  2979
EF57  2980  ------  WAIT_FOR_REQUEST_FOR_MASTER
EF57  2981
EF57  2982  JAO:  RESULTS_ERROR
EF57  2983  XOR  CX,CX  INPUT_LOOP
EF57  2984  MOV  DX,03FH  COUNTER
EF57  2985  J39:  STATUS_PORT
EF57  2986  J39:  WAIT_FOR_MASTER
EF57  2987  IN  AL,DX  GET_STATUS
EF57  2988  TEST  AL,00H  MASTER_READY
EF57  2989  LOOP  J39  WAIT_MASTER
EF57  2990  OR  DISKETTE_STATUS,TIME_OUT
EF57  2991  J39:  RESULTS_ERROR
EF57  2992  STC  SET_ERROR_RETURN
EF57  2993  POP  BX
EF57  2994  POP  DX
EF57  2995  POP  CX
EF57  2996  RET
EF57  2997
EF57  2998  1-----  TEST_THE_DIRECTION_BIT
EF57  2999
EF57  3000  3000  JAOA:  GET_STATUS_REG AGAIN
EF57  3001  IN  AL,DX  GET_STATUS
EF57  3002  TEST  AL,040H  TEST_DIRECTION_BIT
EF57  3003  JNZ  J42  OK_TO_READ_STATUS
EF57  3004  J41:  NEC_FAIL
EF57  3005  OR  DISKETTE_STATUS,BAD_NEC
EF57  3006  JMP  J40  RESULTS_ERROR
EF57  3007
EF57  3008  1-----  READ_IN_THE_STATUS
EF57  3009
EF57  3010  3010  J42:  INPUT_STAT
EF57  3011  IN  DX  POINT_AT_DATA_PORT
EF57  3012  IN  AL,DX  GET_THE_DATA
EF57  3013  MOV  1011AL  STORE_THE_BYTE
EF57  3014  INC  DI  INCREMENT_THE_POINTER
EF57  3015  MOV  CX,10  LOOP_TO_KILL_TIME_FOR_NEC
EF57  3016  LOOPJ  J43  POINT_AT_STATUS_PORT
EF57  3017  DEC  DX
EF57  3018  DEC  AL,DX  GET_STATUS
EF57  3019  TEST  AL,010H  TEST_FOR_NEC_STILL_BUSY
EF57  3020  JX  J44  RESULTS_DONE
EF57  3021  DEC  BL  DECREMENT_THE_STATUS_COUNTER
EF57  3022  JNZ  J56  GO_BACK_FOR_MORE

System BIOS  A-43
LOC OBJ

EFAS IDES
3203 JMP JAI ; CHIP HAS FAILED
3204
3205 ; ------ RESULT OPERATION IS DONE
3206
3207 EFAS
3208 EFAS 5B
3209 POP BX
3210 POP DX
3211 EFAS 59
3212 POP CX
3213 RET ; GOOD RETURN CODE FROM TEST INST
3214
3215 ; NUM_TRANS
3216 ; THIS ROUTINE CALCULATES THE NUMBER OF SECTORS THAT
3217 ; WERE ACTUALLY TRANSFERRED TO/FROM THE DISKETTE
3218 ; INPUT
3219 (CH) = CYLINDER OF OPERATION
3220 (CL) = START SECTOR OF OPERATION
3221 ; OUTPUT
3222 (AL) = NUMBER ACTUALLY TRANSFERRED
3223 ; NO OTHER REGISTERS MODIFIED
3224
3225 EFAX
3226 EFAC AD4500
3227 3246 MOV AL,EC_STATUS+3 ; GET CYLINDER ENDED UP ON
3228 CHI AL,CH ; SAME AS WE STARTED
3229 3247 MOV AL,EC_STATUS+3 ; GET ENDING SECTOR
3230 JE JAS ; IF ON SAME CYL, THEN NO ADJUST
3231 3248 MOV BL,B
3232 CALL GET_PARM ; GET EOT VALUE
3233 3249 MOV AL,AN ; INTO AL
3234 SUB AL,CL ; USE EOT+1 FOR CALCULATION
3235 3250 JAS ; SUBTRACT START FROM END
3236 3251 RET
3237
3238 NAM TRANS
3239 ENDP
3240
3241 RESULTS ENDP
3242
3243 ; DISK_BASE
3244 ; THIS IS THE SET OF PARAMETERS REQUIRED FOR DISKETTE OPERATION
3245 ; THEY ARE POINTED AT BY THE DATA VARIABLE DISK_POINTER, TO
3246 ; MODIFY THE PARAMETERS, BUILD ANOTHER PARAMETER BLOCK AND POINT
3247 ; TO OBJECT
3248
3249 EFC7
3250 ORG 0EFC7
3251 DISK_BASE LABEL BYTE
3252 EFC7 CP
3253 3254 DB 11001111B ; SET=C, HD UNLOAD=OF - 1ST SPECIFY BYTE
3255 EFC7 CP
3256 3257 DB 2 ; HD LOAD=1, HD=OPM - 2ND SPECIFY BYTE
3258 EFC7 25
3259 3260 DB MOTOR_WAIT ; WAIT AFTER OPM'IL MOTOR OFF
3261 EFC7 02
3262 3263 DB 2 ; 512 BYTES/SECTOR
3264 EFC7 OB
3265 3266 DB 3 ; EOT (LAST SECTOR ON TRACK)
3267 EFC7 FF
3268 3269 DB 0BAH ; GAP LENGTH
3270 EFC7 5D
3271 3272 DB OFFH ; DTL
3273 EFC7 6F
3274 3275 DB 050H ; GAP LENGTH FOR FORMAT
3276 EFC7 FD
3277 3278 DB 0F3H ; FILF BYTE FOR FORMAT
3279 EFC7 19
3280 3281 DB 2S ; HEAD SETTLE TIME (MILLISECONDS)
3282 EFC7 04
3283 3284 DB 4 ; MOTOR START TIME (1/00 SECONDS)
3285
3286 ; ------ INT 17
3287 ; PRINTER_IOCTL
3288 ; THIS ROUTINE PROVIDES COMMUNICATION WITH THE PRINTER
3289 ; INPUT
3290 ; (AH)=0 PRINT THE CHARACTER IN (AL)
3291 ; ON RETURN, AH=1 IF CHARACTER COULD NOT BE PRINTED
3292 ; (AH)=1 TIME OUT, OTHER BITS SET AS ON NORMAL STATUS CALL
3293 ; (AH)=2 INITIALIZE THE PRINTER PORT
3294 ; RETURNS WITH (AH) SET WITH PRINTER STATUS
3295 ; (AH)=1 READ THE PRINTER STATUS INTO (AH)
3296 ; (AH)=0 7 6 5 4 3 2-1 0
3297 ; 7 = TIME OUT; 6 = BUSY; 5 = ACKNOWLEDGE; 4 = UNSELECTED;
3298 ; 3= EOT; 2= Z/E Error; 1= OUT OF PAPER
3299 ; 0=NOK (I/E) = ETA CORRESPONDING TO ACTUAL
3300
3301 ; VALUES IN PRINTER_BASE AREA

A-44 System BIOS
; DATA AREA. PRINTER_BASE CONTAINS THE BASE ADDRESS OF THE PRINTER
; CARD81 AVAILABLE (LOCATED AT BEGINNING OF DATA SEGMENT).
; 40BH ABSOLUTE, 3 WORDS
; DATA AREA. PRINT Tim.OUT (BYTE) MAY BE CHANGED TO CAUSE DIFFERENT
; TIME-OUT MWT. DEFAULT=20
; REGISTERS AH IS MODIFIED
; ALL OTHERS UNCHANGED

;------------------------------------------
; A860 ASSUME CS:CODE,DS:DATA
; ORG 8E0H
; PRINTER.ID PROC FAR
; STI
; PUSH DS
; PUSH DX
; PUSH SI
; PUSH CX
; PUSH BX
; CALL DDS
; MOV SI,DX
; SET PRINTER PARA
; MOV BL,PRINT Tim.OUT
; LOAD TIME-OUT PARA
; SHL SI,1
; XOR OFFSET INTO TABLE
; MOV DX,PRINTER_BASE
; SET BASE ADDRESS FOR PRINTER CARD
; OR DX,DX
; TEST DX FOR ZERO
; INDICATING NO PRINTER
; JE B1
; RET
; OR AH,AH
; TEST FOR AH=0
; JZ B2
; JE B3
; IOS
; PRINT_AL
; TEST FOR AH=1
; JZ B4
; INIT_PAR
; DEC AH
; TEST FOR AH=2
; JZ B5
; PRINTER_STATUS
; B1:
; POP BX
; POP CX
; POP SI
; RECOVER REGISTERS
; POP DS
; RECOVER REGISTERS
; IRET

;------- PRINT THE CHARACTER IN (AL) -------
; PUSH AX
; SAVE VALUE TO PRINT
; OUT DX,AL
; OUTPUT CHAR TO PORT
; INC DX
; POINT TO STATUS PORT
; SUB CX,CX
; WAIT_BUSY
; IN AL,DX
; GET STATUS
; MOV AH,AL
; STATUS TO AH ALSO
; TEST AL,0AH
; IS THE PRINTER CURRENTLY BUSY
; JNZ B4
; OUT_StROBE
; LOOP BS:1
; TRY AGAIN
; DEC BL
; DROP LOOP COUNT
; JNZ B3
; GO TILL TIMEOUT ENDS
; XOR AH,1
; SET ERROR FLAG
; AMD AH,6FH
; TURN OFF THE OTHER BITS
; JMP SHORT BS
; RETURN WITH ERROR FLAG SET
; MOV AL,0DH
; STROBE IS BIT 0 OF PORT C OF 8255
; MOV DX,AL
; SET THE STROBE HIGH
; MOV AL,6CH
; SET THE STROBE LOW
; OUT DX,AL
; RECOVER THE OUTPUT CHAR
; PUSH AX
; SAVE AL REG
; MOV DX,PRINTER_BASE
; OR DX,SI
; GET PRINTER STATUS
; pushing Ax
; push 8e0h
; push ds
; push dx
; push si
; push cx
; push bx
; call dds
; mov si, dx
; set printer para
; mov bl, print tim, out
; load time-out para
; shl si, 1
; xor offset into table
; mov dx, printer_base
; set base address for printer card
; or dx, dx
; test dx for zero
; indicating no printer
; je b1
; return
; or ah, ah
; test for ah = 0
; jz b2
; je b3
; ios
; print_al
; test for ah = 1
; jz b4
; init_par
; dec ah
; test for ah = 2
; jz b5
; printer_status
; b1:
; pop bx
; pop cx
; pop si
; recover registers
; pop ds
; recover registers
; iret

;------- print the character in (al) -------
; push ax
; save value to print
; out dx, al
; output char to port
; inc dx
; point to status port
; sub cx, cx
; wait_busy
; in al, dx
; get status
; mov ah, al
; status to ah also
; test al, 0ah
; is the printer currently busy
; jnz b4
; out_strobe
; loop bs:1
; try again
; dec bl
; drop loop count
; jnz b3
; go till timeout ends
; xor ah, 1
; set error flag
; and ah, 6fh
; turn off the other bits
; jmp short bs
; return with error flag set
; mov al, 0dh
; strobe is bit 0 of port c of 8255
; mov dx, al
; set the strobe high
; mov al, 6ch
; set the strobe low
; out dx, al
; recover the output char
; push ax
; save al reg
; mov dx, printer_base
; or dx, si
; get printer status

; system BIOS  a-45
LOC OBJ   LINE   SOURCE

; INITIALIZE THE PRINTING PORT

F030  BA: 3186
F030  50  3187  PUSH AX
F031  42  3188  INC DX
F032  42  3189  INC DX
F033  B00  3190  MOV AL,08H
F035  EE  3191  OUT DX,AL
F036  80H  3192  MOV AX,1000
F039  B9: 3193  DL,AL
F03A  75F0  3194  DEC AX
F03C  80CH  3195  JNZ B9
F03E  EE  3196  MOV AL,0CH
F03F  E800  3197  OUT DX,AL
F200  BA: 3199  MOV AX,PIR0
F201  ENDP

; VIDEO_IO

; THESE ROUTINES PROVIDE THE CRT INTERFACE

; THE FOLLOWING FUNCTIONS ARE PROVIDED:

; (AH)=0  SET MODE (AL) CONTAINS MODE VALUE
;     (AL)=0 40X80 BM (POWER ON DEFAULT)
;     (AL)=1 40X25 BM
;     (AL)=2 80X25 BM
;     (AL)=3 132X25 BM
;     (AL)=4 132X20 COLOR
;     (AL)=5 250X20 BM
;     (AL)=6 400X25 BM
;     (AL)=7 BM (NOT IN BM) Color mode same as 80 mode, but
;          COLOR BURST IS NOT ENABLED

; (AH)=1  SET CURSOR TYPE
; (CH) = BIT 4-0 = START LINE FOR CURSOR
; ** HARDWARE WILL ALWAYS CAUSE BLINK
; ** SETTING BIT 5 OR 6 WILL CAUSE ERRATIC
; ** BLINKING ON OR NO CURSOR AT ALL
; (CL) = BIT 6-0 = END LINE FOR CURSOR

; (AH)=2  SET CURSOR POSITION
; (DH,DL) = ROW,COLUMN (0,0) IS UPPER LEFT
; (BM) = PAGE NUMBER (MUST BE 0 FOR GRAPHICS MODES)

; (AH)=3  READ CURSOR POSITION
; (BH) = PAGE NUMBER (MUST BE 0 FOR GRAPHICS MODES)
; ON EXIT (DH,DL) = ROW,COLUMN OF CURRENT CURSOR

; (AH)=4  CURSOR MODE CURRENTLY SET

; (AH)=5  READ LIGHT PEN POSITION
; ON EXIT:
; (AH) = 0 = LIGHT PEN SWITCH NOT DOWN/NOT TRIGGERED
; (AH) = 1 = VALID LIGHT PEN VALUE IN REGISTERS
; (DH,DL) = ROW,COLUMN OF CHARACTER LF POSH
; (CH) = RASTER LINE (0-199)
; (DX) = PIXEL COLUMN (0-319,439)

; (AH)=6  SELECT ACTIVE DISPLAY PAGE (VALID ONLY FOR ALPHA MODES)
; (AL)=NEW PAGE VAL (0-7 FOR MODES 01H, 0-3 FOR MODES 23H)

; (AH)=7  SCROLL ACTIVE PAGE UP
; (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT BOTTOM
;       OF WINDOW
; AL = 0 MEANS BLANK ENTIRE WINDOW

; (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL
; (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL
; (BH) = ATTRIBUTE TO BE USED ON BLANK LINE

; ON EXIT (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL
; (AH)=8  SCROLL ACTIVE PAGE DOWN
; (AL) = NUMBER OF LINES, INPUT LINES BLANKED AT TOP
;       OF WINDOW
; AL = 0 MEANS BLANK ENTIRE WINDOW

; (CH,CL) = ROW,COLUMN OF UPPER LEFT CORNER OF SCROLL
; (DH,DL) = ROW,COLUMN OF LOWER RIGHT CORNER OF SCROLL

A-46  System BIOS
(BH) = ATTRIBUTE TO BE USED ON BLANK LINE

CHARACTER HANDLING Routines

(AN) = 0 READ ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION

(BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)

ON EXIT:

(AL) = CHARS READ

(AN) = ATTRIBUTE OF CHARACTER READ (ALPHA MODES ONLY)

(AN) = 9 WRITE ATTRIBUTE/CHARACTER AT CURRENT CURSOR POSITION

(BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)

(CX) = COUNT OF CHARACTERS TO WRITE

(AL) = CHARS TO WRITE

(BL) = ATTRIBUTE OF CHARACTER (ALPHA/COLOR OF CHAR

GRAPHICS

SEE NOTE ON WRITE DOT FOR BIT 7 OF BL = 1.

(AN) = 10 WRITE CHARACTER ONLY AT CURRENT CURSOR POSITION

(BH) = DISPLAY PAGE (VALID FOR ALPHA MODES ONLY)

(CX) = COUNT OF CHARACTERS TO WRITE

(AL) = CHARS TO WRITE

FOR READ/WRITE CHARACTER INTERFACE WHILE IN GRAPhICS MODE, THE

CHARACTERS ARE FORMED FROM A CHARACTER GENERATOR IMAGE

MAINTAINED IN THE SYSTEM ROM. ONLY THE 1ST 128 CHAR

ARE CONTAINED THERE. TO READ/WRITE THE SECOND 128

CHARS, THE USER MUST INITIALIZE THE POINTER AT

INTEMPTRPH (LOCATION 0007) TO POINT TO THE KO BYTE

TABLE CONTAINING THE CODE POINTS FOR THE SECOND

128 CHARAS (128-255).

FOR WRITE CHARACTER INTERFACE IN GRAPHICS MODE, THE REPETITION

FACTOR CONTAINED IN (CX) ON ENTRY WILL PRODUCE VALID

RESULTS ONLY FOR CHARACTERS CONTAINED ON THE SAME ROM.

CONTINUATION TO SUCCEEDING LINES WILL NOT PRODUCE

CORRECTLY.

GRAPHICS INTERFACE

(AN) = 11 SET COLOR PALETTE

(BH) = PALETTE COLOR ID BEING SET (0-127)

(BL) = COLOR VALUE TO BE USED WITH THAT COLOR ID

NOTE: FOR THE CURRENT COLOR CARD, THIS ENTRY POINT

HAS MEANING ONLY FOR 320X200 GRAPHICS.

COLOR ID = 0 SELECTS THE BACKGROUND COLOR (0-15)

COLOR ID = 1 SELECTS THE PALETTE TO BE USED:

0 = GREEN/LI/RED/11/YELLOW/S

1 = CYAN/12/MAGENTA/13/WHITE

IN 400X25 OR 80X25 ALPHA MODES, THE VALUE SET

FOR PALETTE ID 0 INDICATES THE

BORDER COLOR TO BE USED (VALUES 0-31),

WHERE 16-31 SELECT THE HIGH INTENSITY

BACKGROUND SET.

(AN) = 12 WRITE DOT

(DX) = ROM NUMBER

(CX) = COLUMN NUMBER

(AL) = COLOR VALUE

IF BIT 7 OF AL = 1, THEN THE COLOR VALUE IS

EXCLUSIVE OR'D WITH THE CURRENT CONTENTS OF

THE DOT

(AN) = 13 READ DOT

(DX) = ROM NUMBER

(CX) = COLUMN NUMBER

(AL) = RETURNS THE DOT READ

ASCII TELETYPID ROUTINE FOR OUTPUT

(AN) = 14 WRITE TELETYPID TO ACTIVE PAGE

(AL) = CHARS TO WRITE

(BL) = FOREGROUND COLOR IN GRAPHICS MODE

NOTE -- SCREEN WIDTH IS CONTROLLED BY PREVIOUS MODE SET

(AN) = 15 CURRENT VIDEO STATE

RETURNS THE CURRENT VIDEO STATE

(AL) = MODE CURRENTLY SET (SEE AN# FOR EXPLANATION)

(CX) = NUMBER OF CHARACTER COLUMNS ON SCREEN

(BH) = CURRENT ACTIVE DISPLAY PAGE

CS:8050.5:8X.CH.CH BY PRESERVED DURING CALL

ALL OTHERS DESTROYED
A-48  System BIOS
LOC OBJ   LINE   SOURCE

F0AD IC   3402   DB   ICH.2,7,6,7
F0AC 02   3403   DB   0,0,0,0
F0AD 07   3404   M6   EQU  0-VIDEO_PARMS
F0AE 06   3405   DB   71H,56H,5AH,6AH,1FH,61H       ; SET UP FOR BOXES
F0AF 07   3406   DB   1CH.2,7,6,7
F0DD 00   3407   DB   0,0,0,0
F0DD 00   3408   DB   0,0,0,0
F0DD 00   3409   DB   38H,25H,20H,0AH,7FH,6,6,6H   ; SET UP FOR GRAPHICS
F0DD 00   3410   DB   38H,25H,20H,0AH,7FH,6,6,6H   ; SET UP FOR GRAPHICS
F0DD 00   3411   DB   70H,2,1,6,7
F0DD 00   3412   DB   0,0,0,0
F0DD 00   3413   DB   61H,5EH,52H,4FH,1FH,6,19H       ; SET UP FOR AX25 BAH CARD
F0DD 00   3414   DB   61H,5EH,52H,4FH,1FH,6,19H       ; SET UP FOR AX25 BAH CARD
F0DD 00   3415   DB   19H,2,0OH,0BH,1CH
F0DD 00   3416   DB   0,0,0,0
F0DD 00   3417   DB   0,0,0,0
F0DD 00   3418   M5   LABEL  WORD
F0DD 00   3419   DN   2048
F0DD 00   3420   DN   4096
F0DD 00   3421   DN   16384
F0DD 00   3422   DN   16384
F0DD 00   3423   DN   16384
F0DD 00   3424   I----  COLUMNS
F0DD 00   3425   I----  COLUMNS
F0DD 00   3426   M6   LABEL  BYTE
F0DD 00   3427   DB   80,80,80,80,80,80,80,80
LOC OBJ

LINE SOURCE

3629 1----- C_REG_TAB

3630

3631 M7 LABEL BITE 1 TABLE OF MODE SETS

3632 DB EDH, EDH, EDH, EDH, EDH, EDH, EDH, EDH

3633

3634 SET_MODE PROC NEAR

3635 MOV DX,05DH

3636 MOV DL,0

3637 CMP DI,30H

3638 JNE MH 1 IS SM CARD INSTALLED

3639 IN AL,7

3640 MOV DI,0AH

3641 INC BL 1 MODE SET FOR SM CARD

3642

3643 MOV AH,AL 1 SAVE MODE IN AH

3644 MOV CX,0Fh 1 SAVE IN GLOBAL VARIABLE

3645 MOV ADDR_4D,DX 1 SAVE ADDRESS OF BASE

3646 PUSH DS 1 SAVE POINTER TO DATA SEGMENT

3647 PUSH AX 1 SAVE MODE

3648 PUSH DX 1 SAVE OUTPUT PORT VALUE

3649 ADD DL,AL 1 POINT TO CONTROL REGISTER

3650 MOV AL,BL 1 GET MODE SET FOR CARD

3651 OUT DX,AL 1 RESET VIDEO

3652 POP AX 1 BACK TO BASE REGISTER

3653 SUB AX,AX 1 SET UP FOR AB50 SEGMENT

3654 MOV DS,AX 1 ESTABLISH VECTOR TABLE ADDRESSING

3655 ASSUME ds:ab50

3656 LDS BX,PARM_PTR 1 GET POINTER TO VIDEO PARMS

3657 POP AX 1 RECOVER PARMS

3658 ASSUME ds:CODE

3659 MOV CX,5h 1 LENGTH OF EACH ROW OF TABLE

3660 CMP AH,2 1 DETERMINE WHICH ONE TO USE

3661 JC M2 1 MODE IS 0 OR 1

3662 ADD BX,CX 1 MOVE TO NEXT ROW OF INIT TABLE

3663 CMP AH,4 1 MODE IS 2 OR 3

3664 JC M2 1 MOVE TO GRAPHICS ROW OF INIT_TABLE

3665 ADD BX,CX 1 MOVE TO BM CARD ROW OF INIT_TABLE

3666 CMP AH,7 1 MODE IS 4, 5, OR 6

3667 JC M2 1 MOVE TO BM CARD ROW OF INIT_TABLE

3668 ADD BX,CX 1

3669

3670 1----- BX POINTS TO CORRECT ROW OF INITIALIZATION TABLE

3671

3672 MOV AX,0000

3673 OUT_INIT

3674 XOR AH,00

3675 AH WILL SERVE AS REGISTER

3676 NUMBER DURING LOOP

3677

3678 1----- LOOP THROUGH TABLE, OUTPUTTING REG ADDRESS, THEN VALUE FROM TABLE

3679

3680 MOV AL,0AH 1 INIT LOOP

3681 OUT DX,AL 1 GET 6845 REGISTER NUMBER

3682 INC AX 1 POINT TO DATA PORT

3683 NEXT REGISTER VALUE

3684 MOV AL,1(BX) 1 GET TABLE VALUE

3685 OUT DX,AL 1 OUT TO CHIP

3686 INC BX 1 NEXT IN TABLE

3687 DEC BX 1 BACK TO POINTER REGISTER

3688 LOOP MIB 1 DO THE WHOLE TABLE

3689 POP AX 1 GET MODE BACK

3690 POP DS 1 RECOVER SEGMENT VALUE

3691 ASSUME DS:BIO_DATA

3692

3693 1----- FILL RESERN AREA WITH BLANK

3694

3695 XOR DI,O1 1 SET UP POINTER FOR REGEN

-----

A-50 System BIOS
<table>
<thead>
<tr>
<th>LOC</th>
<th>OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>F156</td>
<td>09364E00</td>
<td>3494</td>
<td>MOV CRT_START,DX</td>
</tr>
<tr>
<td>F158</td>
<td>C646620000</td>
<td>3497</td>
<td>MOV ACTIVE_BUSY,0</td>
</tr>
<tr>
<td>F150</td>
<td>800200</td>
<td>3498</td>
<td>MOV CX,019E</td>
</tr>
<tr>
<td>F166</td>
<td>00FC06</td>
<td>3499</td>
<td>CMP AH,4</td>
</tr>
<tr>
<td>F166</td>
<td>7018</td>
<td>3500</td>
<td>JC M16</td>
</tr>
<tr>
<td>F166</td>
<td>00FC07</td>
<td>3501</td>
<td>CMP AH,7</td>
</tr>
<tr>
<td>F166</td>
<td>7004</td>
<td>3502</td>
<td>JE M13</td>
</tr>
<tr>
<td>F166</td>
<td>33D0</td>
<td>3503</td>
<td>XOR AX,AX</td>
</tr>
<tr>
<td>F164</td>
<td>E805</td>
<td>3504</td>
<td>JMP SHORT M13</td>
</tr>
<tr>
<td>F166</td>
<td>5535</td>
<td>M11:</td>
<td>3505</td>
</tr>
<tr>
<td>F166</td>
<td>559B</td>
<td>3506</td>
<td>MOV CH,68H</td>
</tr>
<tr>
<td>F176</td>
<td>5557</td>
<td>M12:</td>
<td>3507</td>
</tr>
<tr>
<td>F170</td>
<td>020207</td>
<td>3508</td>
<td>MOV AX,'+74526</td>
</tr>
<tr>
<td>F173</td>
<td>5559</td>
<td>M13:</td>
<td>3509</td>
</tr>
<tr>
<td>F173</td>
<td>F3</td>
<td>3510</td>
<td>REP STOSH</td>
</tr>
<tr>
<td>F174</td>
<td>AB</td>
<td>3511</td>
<td>FILL THE REGEN BUFFER WITH BLANKS</td>
</tr>
</tbody>
</table>

---

**Appendix A**

---

**System BIOS**

---

A-51
LOC  OBJ  

3571  ; THIS ROUTINE SETS THE CURSOR VALUE :  
3572  ; INPUT  :  
3573  ; (CX) HAS CURSOR VALUE OR-START LINE, CL-STOP LINE :  
3574  ; OUTPUT  :  
3575  ; NONE  :  
3576  ; ------------------------------------------  
3577  FICD  3577  SET_CTYPE  PROC  NEAR  
3578  FICD  3578  B8A4  MOV  AL,10  ; 6445 REGISTER FOR CURSOR SET  
3579  FICF  3579  00 64500  MOV  CURSOR_MODE.CX  ; SAVE IN DATA AREA  
3580  FII3  3580  EB00200  CALL  MIL  ; OUTPUT CX REG  
3581  FIDC  3581  EB00  JMP  VIDEO_RETURN  
3582  FICD  3582  3583  E----- THIS ROUTINE OUTPUTS THE CX REGISTER TO THE 6445 REGS NAMED IN AH  
3584  FICO  3584  EB00  
3585  FIDC  3585  B0166300  MOV  DX,ADDR_6445  ; ADDRESS REGISTER  
3586  FIDC  3586  BAC6  MOV  AL,AH  ; SET VALUE  
3587  FIOE  3587  EE  OUT  DX,AL  ; REGISTER SET  
3588  FIOE  3588  EF4  INC  DX  ; DATA REGISTER  
3589  FIDC  3589  BAC5  MOV  AL,CH  ; DATA  
3590  FIOE  3590  EE  OUT  DX,AL  
3591  FIOE  3591  4A  DEC  DX  
3592  FIOE  3592  BAC4  MOV  AL,AH  ; POINT TO OTHER DATA REGISTER  
3593  FIOE  3593  EFC0  INC  AL  
3594  FIOE  3594  EE  OUT  DX,AL  ; SET FOR SECOND REGISTER  
3595  FIOE  3595  EF4  INC  DX  ; SECOND DATA VALUE  
3596  FIOE  3596  BAC1  MOV  AL,CL  
3597  FIOE  3597  EE  OUT  DX,AL  
3598  FICD  3598  C3  RET  ; ALL DONE  
3599  FICD  3600  SET_CTYPE  ENDP  
3600  ; ------------------------------------------  
3601  FIEE  3601  EBACF  MOV  CX,BH  
3602  FIF0  3602  31ED  MOV  CH,CH  ; ESTABLISH LOOP COUNT  
3603  FIF2  3603  D1F1  SAL  CX,1  ; WORD OFFSET  
3604  FIF4  3604  BF01  MOV  SI,CX  ; USE INDEX REGISTER  
3605  FIF6  3605  094509  MOV  ($I+OFFSET CURSOR_POSI),DX  ; SAVE THE POINTER  
3606  FIF9  3606  3566200  CMP  ACTIVE_PAGE,BH  
3607  FIFD  3607  7905  JNZ  MIL  ; SET_CPOS_RETURN  
3608  FIFF  3608  B8C2  MOV  AX,DX  ; SET ROW/COLUMN TO AX  
3609  FIF0  3609  EB0200  CALL  MIL  ; CURSOR_SET  
3610  FIF1  3610  EBD2  MI7  ; SET_CPOS_RETURN  
3611  FIF4  3611  EBDF  JMP  VIDEO_RETURN  
3612  FICD  3612  SET_CPOS  PROC  NEAR  
3613  FICD  3613  B8A4  MOV  CL,CM  
3614  FIF0  3614  00ED  XOR  CX,CH  
3615  FIF2  3615  D1F1  SAL  CX,1  
3616  FIF4  3616  BF01  MOV  SI,CX  ; USE INDEX REGISTER  
3617  FIF6  3617  094509  MOV  ($I+OFFSET CURSOR_POSI),DX  ; SAVE THE POINTER  
3618  FIF9  3618  3566200  CMP  ACTIVE_PAGE,BH  
3619  FIFD  3619  7905  JNZ  MIL  ; SET_CPOS_RETURN  
3620  FIFF  3620  B8C2  MOV  AX,DX  ; SET ROW/COLUMN TO AX  
3621  FIF0  3621  EB0200  CALL  MIL  ; CURSOR_SET  
3622  FIF1  3622  EBD2  MI7  ; SET_CPOS_RETURN  
3623  FIF4  3623  EBDF  JMP  VIDEO_RETURN  
3624  FICD  3624  SET_CPOS  ENDP  
3625  ; ------------------------------------------  
3626  FICD  3626  EB06  MOV  AL,XA  ; SET CURSOR POSITION, AX HAS ROW/COLUMN FOR CURSOR  
3627  FICD  3627  3567  ; DETERMINE LOCATION IN REGEN BUFFER  
3628  FICD  3628  B8A4  MOV  CX,AX  
3629  FICD  3629  EBC000  CALL  POSITION  
3630  FICD  3630  08C6  MOV  CX,AX  
3631  FICD  3631  DD5F  ADD  CX,CRT_START  ; ADD IN THE START ADDR FOR THIS PAGE  
3632  FICD  3632  03D4200  SAL  CX,1  ; DIVIDE BY 2 FOR CHAR ONLY COUNT  
3633  FICD  3633  B01E  MOV  AH,1A  ; REGISTER NUMBER FOR CURSOR  
3634  FICD  3634  EB28FF  CALL  MIL  ; OUTPUT THE VALUE TO THE 6445  
3635  FICD  3635  CX  RET  
3636  FICD  3636  B8A4  MOV  AL,MX  
3637  FICD  3637  3565  ; ACT_DISP PAGE  
3638  FICD  3638  3563  ; THIS ROUTINE SETS THE ACTIVE DISPLAY PAGE, ALLOWING THE  
3639  FICD  3639  3560  ; FULL USE OF THE RAM SET ALONG WITH THE VIDEO ATTACHMENT  
3640  FICD  3640  3569  ; AL HAS THE NEW ACTIVE DISPLAY PAGE  
3641  FICD  3641  3564  ; OUTPUT  
3642  FICD  3642  3565  ; THE 6445 IS RESET TO DISPLAY THAT PAGE  
3643  FICD  3643  3566  ; ------------------------------------------  
3644  FICD  3644  3567  ACT_DISP_PAGE  PROC  NEAR  
3645  FICD  3645  3568  MOV  ACTIVE_PAGE,AL  ; SAVE ACTIVE PAGE VALUE  

A-52  System BIOS
LOC OBJ

LINE SOURCE

F234 000410B 3460 MOV CX,CRT_LEN  
3461  
3462  
3463  
3464  
3465  
3466  
3467  
3468  
3469  
3470  
3471  
3472  
3473  
3474  
3475  
3476  
3477  
3478  
3479  
3480  
3481  
3482  
3483  
3484  
3485  
3486  
3487  
3488  
3489  
3490  
3491  
3492  
3493  
3494  
3495  
3496  
3497  
3498  
3499  
3500  
3501  
3502  
3503  
3504  
3505  
3506  
3507  
3508  
3509  
3510  
3511  
3512  
3513  
3514  
3515  
3516  
3517  
3518  
3519  
3520  
3521  
3522  
3523  
3524  
3525  
3526  
3527  
3528  
3529  
3530  
3531  
3532  
3533  
3534  
3535  
3536  
3537  
3538  
3539  
3540  
3541  
3542  
3543  
3544  
3545  
3546  
3547  
3548  
3549  
3550  
3551  
3552  
3553  
3554  
3555  
3556  
3557  
3558  
3559  
3560  
3561  
3562  
3563  
3564  
3565  
3566  
3567  
3568  
3569  
3570  
3571  
3572  
3573  
3574  
3575  
3576  
3577  
3578  
3579  
3580  
3581  
3582  
3583  
3584  
3585  
3586  
3587  
3588  
3589  
3590  
3591  
3592  
3593  
3594  
3595  
3596  
3597  
3598  
3599  
3600  
3601  
3602  
3603  
3604  
3605  
3606  
3607  
3608  
3609  
3610  
3611  
3612  
3613  
3614  
3615  
3616  
3617  
3618  
3619  
3620  
3621  
3622  
3623  
3624  

Appendix A

System BIOS A-53
F26A 3725 M6D:
F26A 240F 3726 AND AL,DL8H; TURN OFF PALETTE SELECT BIT
F26C 008H 3727 SHR BL,1; TEST THE LOW ORDER BIT OF BL
F29E 7F53 3728 JNC H9; ALREADY DONE
F270 8C28 3729 OR AL,DL8H; TURN ON PALETTE SELECT BIT
F272 EBFF 3730 JMP H9; GO DO IT
F271 3731 SET_COLOR ENDP

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>VIDEO STATE</td>
</tr>
<tr>
<td>--------------------------</td>
</tr>
<tr>
<td>1 RETURN THE CURRENT VIDEO STATE IN AX</td>
</tr>
<tr>
<td>1 AH = NUMBER OF COLUMNS ON THE SCREEN</td>
</tr>
<tr>
<td>1 AL = CURRENT VIDEO MODE</td>
</tr>
<tr>
<td>1 BH = CURRENT ACTIVE PAGE</td>
</tr>
<tr>
<td>--------------------------</td>
</tr>
</tbody>
</table>

F274 3734 VIDEO_STATE PROC NEAR
F274 0A26400 3740 MOV AH,BYTE PTR CRT_COLS; GET NUMBER OF COLUMNS
F274 0A04900 3741 MOV AL,CRT_MODE; CURRENT MODE
F274 0A36209 3742 MOV BH,ACTIVE_PAGE; GET CURRENT ACTIVE PAGE
F274 5F 3743 POP DI; RECOVER REGISTERS
F280 9E 3744 POP SI
F280 99 3745 POP CX; DISCARD SAVED BX
F282 E943FF 3746 JMP NS; RETURN TO CALLER
F282 3747 VIDEO_STATE ENDP

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>POSITION</td>
</tr>
<tr>
<td>--------------------------</td>
</tr>
<tr>
<td>1 THIS SERVICE ROUTINE CALCULATES THE REGEN</td>
</tr>
<tr>
<td>1 BUFFER ADDRESS OF A CHARACTER IN THE ALPHA MODE</td>
</tr>
<tr>
<td>1 AX = ROW, COLUMN POSITION</td>
</tr>
<tr>
<td>1 OUTPUT</td>
</tr>
<tr>
<td>--------------------------</td>
</tr>
</tbody>
</table>

F285 3750 POSITION PROC NEAR
F285 33 3756 PUSH BX; SLEAVE REGISTER
F286 600A 3757 MOV BX,AX
F288 7AC4 3758 MOV AL,AX; ROWS TO AL
F28A F266440 3759 MUL BYTE PTR CRT_COLS; DETERMINE BYTES TO ROW
F29E 30FF 3760 XOR BH,BH
F29D 00C3 3761 ADD AX,BX; ADD IN COLUMN VALUE
F29D D10 3762 SAL AX,1; = 2 FOR ATTRIBUTE BYTES
F29D 5B 3763 POP BX
F29E C3 3764 RET
F29E 3765 POSITION ENDP

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>SCROLL_UP</td>
</tr>
<tr>
<td>--------------------------</td>
</tr>
<tr>
<td>1 THIS ROUTINE MOVES A BLOCK OF CHARACTERS UP</td>
</tr>
<tr>
<td>1 ON THE SCREEN</td>
</tr>
<tr>
<td>--------------------------</td>
</tr>
</tbody>
</table>

F29F 3769 SCROLL_UP PROC NEAR
F29F 6A08 3770 MOV BL,AL; SAVE LINE COUNT IN BL
F29F 00C04 3771 CMP AN,4; TEST FOR GRAPHICS MODE
F29F 7208 3772 JC H1; HANDLE SEPARATELY
F29F 80CF37 3773 CMP AN,7; TEST FOR BH CARD
F29F 7403 3774 JE H1
F2A2 69001 3775 JMP GRAPHICS_UP
F2A4 3776 SCROLL_UP ENDP

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CS/OS:DATA,E5:DATA</td>
</tr>
</tbody>
</table>

A-54 System BIOS
LOC OBJ  
LINE  
SOURCE  

F2BB 03FD 3002 ADD DI,BP  
POINT TO NEXT LINE IN BLOCK  
F2BA 0ECC 3003 DEC AH  
COUNT OF LINES TO MOVE  
F2BC 75F5 3004 JNZ N2  
ROLL_LOOP  
F2BE 3005 N3:  
CLEAR_ENTRY  
F2BE 50 3006 POP AX  
RECOVER ATTRIBUTE IN AL  
F2BF 8020 3007 MOV AL,0  
FILL WITH BLANKS  
F2C1 3008  
CLEAR_LOOP  
F2C2 E80000 3009 CALL AH  
CLEAR THE ROM  
F2C4 03FD 3010 ADD DI,BP  
POINT TO NEXT LINE  
F2C6 FEOB 3011 DEC BL  
COUNTER OF LINES TO SCROLL  
F2C8 75F7 3012 JNZ N4  
CLEAR_LOOP  
F2CA 3013 N5:  
SCROLL_END  
F2CA E80000 3014 CALL 0BB  
F2CD 003E900007 3015 CMP CRT_MODE,7  
IS THIS THE BLACK AND WHITE CARD  
F2DE 7407 3016 JE N6  
IF SO, SKIP THE MODE RESET  
F2DF A06500 3017 MOV AL,CRT_MODE_SET  
GET THE VALUE OF THE MODE SET  
F2D7 8AD503 3018 MOV AX,030BH  
ALWAYS SET COLOR CARD PORT  
F2DA EE 3019 OUT DX,AL  
VIDEO_SET_HERE  
F2DB 3020 N6:  
BLANK_FIELD  
F2DE E87F7E 3021 JMP VIDEO_RETURN  
F2EE 3022 N7:  
GET_ROM_COUNT  
F2ED 800C 3023 MOV BL,0H  
F2EE 3024 JMP N3  
GO CLEAR THAT AREA  
F2ED 3025 SCROLL_UP ENDP  
F2EE 3026  
F2EE 3027 I----- HANDLE COMMON SCROLL SET UP HERE  
F2EE 3028  
F2EE 3029 SCROLL_POSITION PROC NEAR  
F2EE 803E490002 3030 CMP CRT_MODE,2  
TEST FOR SPECIAL CASE HERE  
F2EE 7716 3031 JB N9  
HAVE TO HANDLE BOXES SEPARATELY  
F2EE 003E490003 3032 CMP CRT_MODE,3  
F2EE 7711 3033 JA N9  
F2EE 3034  
F2EE 3035 I----- BOXES COLOR CARD SCROLL  
F2EE 3036  
F2FE 80 3037 PUSH BX  
F2FE 8E0D3 3038 MOV BX,3DAH  
F2FF 50 3039 PUSH AX  
F2FF 3040 N8:  
F2FF EC 3041 IN AL,DX  
F2FF 8A0D 3042 TEST AL,8  
F2FF 74FB 3043 JE HB  
F2FF 8025 3044 MOV AL,DX  
F2FF 8206 3045 MOV DL,0DH  
F2FF EE 3046 OUT DX,AL  
F2FF 50 3047 POP AX  
F2FF 3048 POP DX  
F2F0 3049  
F2F0 80 3050 MOV BX,0DX  
F2F1 8AD3 3051 PUSH BX  
F2F4 50 3052 MOV BX,3DAH  
F2F5 3053 PUSH AX  
F2F5 3054 N9:  
F2F5 EC 3055 IN AL,DX  
F2F6 8A0D 3056 TEST AL,16  
F2F6 74FB 3057 JE HB  
F2F6 8025 3058 MOV AL,DX  
F2F6 8206 3059 MOV DL,0DH  
F2F6 EE 3060 OUT DX,AL  
F2F7 50 3061 POP AX  
F2F7 3062 POP DX  
F2F8 3063  
F2F8 8E0D3 3064 ADD BX,CRT_START  
F2F9 3065 MOV DI,AX  
F2FA 0B0F 3066 MOV SI,AX  
F2FC 2BD1 3067 SUB DX,AX  
F2FC FECC 3068 INC DH  
F2FC F0CD 3069 INC DL  
F2FD 32ED 3070 XOR CH,CH  
F2F1 4B2E4600 3071 MOV BP,CRT_COLS  
F2F1 03ED 3072 ADD BP,BP  
F2F1 8A0C 3073 MOV DI,0  
F2F1 F26A4600 3074 MOV AL,BL  
F2F1 8C00 3075 ADD BYTE PTR CRT_COLS  
F2F1 06 3076 ADD AX,AX  
F2F2 1F 3077 PUSH ES  
F2F2 06 3078 POP DS  
F2F2 8F7800 3079 CMP BL,0  
F2F2 86 307A RET  
F2F2 307B SCROLL_POSITION ENDP  
F2F2 307C  
F2F2 307D I----- MOVE_ROM  
F2F2 307E  
F2F2 307F N10:  
PROG_NEAR  
F2F2 8AC4 3080 MOV CL,DL  
GET 8 OF COLS TO MOVE  
F2F2 A6 3081 PUSH SI  
F2F2 57 3082 PUSH DI  
F2F2 F4 3083 REP MOVSB  
F2F2 A6 3084  
F2F2 5F 3085 POP DI  
F2F2 3086 POP SI  
F2F2 BE 3087 I----- RECOVER ADDRESSES  

Appendix A

System BIOS A-55
F330 C5
3678  RET
3679     H10  EMFP
3680
3681  ------- CLEAR ROW
3682
F331   PROC NEAR
F331 BACA
3686  MOV  CL,DL    ; GET # COLUMNS TO CLEAR
3687  PUSH  DI
3688  REP  STOWH    ; STORE THE FILL CHARACTER
F335 AB
F336 SF
3687  PDP  DI
F337 C5
3688  RET
3689     H11  EMFP
3690
3691  SCROLL_DOWN
3692  THIS ROUTINE MOVES THE CHARACTERS WITHIN A
3693  DEFINED BLOCK DOWN ON THE SCREEN, FILLING THE
3694  TOP LINES WITH A DEFINED CHARACTER
3695  INPUT
3696  (AH) = CURRENT CRT MODE
3697  (AL) = NUMBER OF LINES TO SCROLL
3698  (CX) = UPPER LEFT CORNER OF REGION
3699  (DX) = LOWER RIGHT CORNER OF REGION
3700  (BH) = FILL CHARACTER
3701  (DS) = DATA SEGMENT
3702  (ES) = REGSEG SEGMENT
3703  OUTPUT
3704  NONE -- SCREEN IS SCROLLED
3705
F338   PROC NEAR
F338 FD
3707  STD
3708  MOV  DL,AL    ; DIRECTION FOR SCROLL DOWN
3709  MOV  BL,AL    ; LINE COUNT TO BL
3710  CMP  AH,4     ; TEST FOR GRAPHICS
3711  JC   H16
3712  JE    H12
3713  JMP  GRAPHICS_DOWN
F33A
3714  N12:  CONTINUE_DOWN
F33B 53
3715  PUSH  BX
3716  MOV  AX,DX    ; SAVE ATTRIBUTE IN BX
F33D  E9AFF
3717  CALL  SCROLL_POSITION    ; GET REGSEG LOCATION
F33E  7420
3718  JC   H16
F33F  2BF0
3719  SUB  SI,AX    ; SI IS FROM ADDRESS
3720  MOV  AX,DX    ; GET TOTAL # RNS
3721  MOV  SI,AX    ; COUNT TO MOVE IN SCROLL
F332  EA66
3722  MOV  AX,H10    ; MOVE ONE ROW
F333  2BF5
3723  MOV  AX,DX    ;езульт
3724  DEC  AX
3725  JNZ  H13
F361  5B
3726  POP  AX    ; RECOVER ATTRIBUTE IN AX
3727
F362  B020
3728  MOV  AL,1    ;
3729
F364   N15:  RESTART
F366  E9AFF
3732  CALL  H11    ; CLEAR ONE ROW
F367  2DF0
3733  SUB  DI,DX    ; GO TO NEXT ROW
F368  FFBC
3734  DEC  BL
F369  57FF
3735  JNZ  H13
F36A  E9AFF
3736  JMP  H5    ; SCROLL_END
F370   N16:  CONTINUE
F371  8ADE
3738  MOV  BL,DX    ;
F372  ED8D
3739  JMP  H14
F390  SCROLL_DOWN  EMFP
F391  -------------------------------
F392  READ_AC_CURRENT
F393  THIS ROUTINE READS THE ATTRIBUTE AND CHARACTER
F394  AT THE CURRENT CURSOR POSITION AND RETURNS THEM
F395  TO THE CALLER
F396  INPUT
F397  (AH) = CURRENT CRT MODE
F398  (BH) = DISPLAY PAGE  (ALPHA MODES ONLY)
F399  (CS) = DATA SEGMENT
3900  (ES) = REGSEG SEGMENT
3901  OUTPUT
3902  (AL) = CHAR READ
3903  (AH) = ATTRIBUTE READ
3904
A-56  System BIOS
ASSUME CS:CODE,DS:DATA,ES:DATA

3954  F3 74 B0FC04 READ_AC_CURRENT PROC NEAR
3957  F3 77 B10B  JMP  FIND_POSITION   I IS THIS GRAPHICS
3958  F3 79 80FC07  JMP  FIND_POSITION   I IS THIS BM CARD
3959  F3 7C 7903  JMP  FIND_POSITION   I IS THIS BM CARD
3960  F3 7E EE0020  JMP  FIND_POSITION
3962  F3 81 8B1A00  JMP  FIND_POSITION
3963  F3 84 00F3  JMP  FIND_POSITION
3964  F3 86 408600  JMP  FIND_POSITION
3965  F3 88 806600  JMP  FIND_POSITION
3966  F3 8A 6CC206  JMP  FIND_POSITION
3967  F3 8D 06  JMP  FIND_POSITION
3968  F3 8E 1F  JMP  FIND_POSITION
3969  F3 90 8000  JMP  FIND_POSITION
3970  F3 92 750B  JMP  FIND_POSITION
3971  F3 94 6A  JMP  FIND_POSITION
3972  F3 95 78  JMP  FIND_POSITION
3973  F3 96 4001  JMP  FIND_POSITION
3974  F3 98 57B0  JMP  FIND_POSITION
3975  F3 99 7A  JMP  FIND_POSITION
3976  F3 9A 79  JMP  FIND_POSITION
3977  F3 9B 6F08  JMP  FIND_POSITION
3978  F3 9C 4506  JMP  FIND_POSITION
3979  F3 9D 0100  JMP  FIND_POSITION
3980  F3 9E 827E  JMP  FIND_POSITION
3981  F3 9F E00001  JMP  FIND_POSITION
3982  F3 A0 0000  JMP  FIND_POSITION
3983  F3 A1 0000  JMP  FIND_POSITION
3984  F3 A2 0000  JMP  FIND_POSITION
3985  F3 A3 0000  JMP  FIND_POSITION
3986  F3 A4 0000  JMP  FIND_POSITION
3987  F3 A5 0000  JMP  FIND_POSITION
3988  F3 A6 0000  JMP  FIND_POSITION
3989  F3 A7 0000  JMP  FIND_POSITION
3990  F3 A8 0000  JMP  FIND_POSITION
3991  F3 A9 0000  JMP  FIND_POSITION
3992  F3 AA 0000  JMP  FIND_POSITION
3993  F3 AB 0000  JMP  FIND_POSITION
3994  F3 AC 0000  JMP  FIND_POSITION
3995  F3 AD 0000  JMP  FIND_POSITION
3996  F3 AE 0000  JMP  FIND_POSITION
3997  F3 AF 0000  JMP  FIND_POSITION
3998  F3 B0 0000  JMP  FIND_POSITION
3999  F3 B1 0000  JMP  FIND_POSITION
4000  F3 B2 0000  JMP  FIND_POSITION
4001  F3 B3 0000  JMP  FIND_POSITION
4002  F3 B4 0000  JMP  FIND_POSITION
4003  F3 B5 0000  JMP  FIND_POSITION
4004  F3 B6 0000  JMP  FIND_POSITION
4005  F3 B7 0000  JMP  FIND_POSITION
4006  F3 B8 0000  JMP  FIND_POSITION
4007  F3 B9 0000  JMP  FIND_POSITION
4008  F3 BA 0000  JMP  FIND_POSITION
4009  F3 BB 0000  JMP  FIND_POSITION
4010  F3 BC 0000  JMP  FIND_POSITION
4011  F3 BD 0000  JMP  FIND_POSITION
4012  F3 BE 0000  JMP  FIND_POSITION
4013  F3 BF 0000  JMP  FIND_POSITION
4014  F3 C0 0000  JMP  FIND_POSITION
4015  F3 C1 0000  JMP  FIND_POSITION
4016  F3 C2 0000  JMP  FIND_POSITION
4017  F3 C3 0000  JMP  FIND_POSITION
4018  F3 C4 0000  JMP  FIND_POSITION
4019  F3 C5 0000  JMP  FIND_POSITION
4020  F3 C6 0000  JMP  FIND_POSITION
4021  F3 C7 0000  JMP  FIND_POSITION
4022  F3 C8 0000  JMP  FIND_POSITION
4023  F3 C9 0000  JMP  FIND_POSITION
4024  F3 CA 0000  JMP  FIND_POSITION
4025  F3 CB 0000  JMP  FIND_POSITION
4026  F3 CC 0000  JMP  FIND_POSITION
4027  F3 CD 0000  JMP  FIND_POSITION
4028  F3 CE 0000  JMP  FIND_POSITION
4029  F3 CF 0000  JMP  FIND_POSITION
4030  F3 D0 0000  JMP  FIND_POSITION
4031  F3 D1 0000  JMP  FIND_POSITION
4032  F3 D2 0000  JMP  FIND_POSITION
4033  F3 D3 0000  JMP  FIND_POSITION
4034  F3 D4 0000  JMP  FIND_POSITION
4035  F3 D5 0000  JMP  FIND_POSITION
4036  F3 D6 0000  JMP  FIND_POSITION
4037  F3 D7 0000  JMP  FIND_POSITION
4038  F3 D8 0000  JMP  FIND_POSITION
4039  F3 D9 0000  JMP  FIND_POSITION
4040  F3 DA 0000  JMP  FIND_POSITION
4041  F3 DB 0000  JMP  FIND_POSITION
4042  F3 DC 0000  JMP  FIND_POSITION
4043  F3 DD 0000  JMP  FIND_POSITION
4044  F3 DE 0000  JMP  FIND_POSITION
4045  F3 DF 0000  JMP  FIND_POSITION
4046  F3 E0 0000  JMP  FIND_POSITION
4047  F3 E1 0000  JMP  FIND_POSITION
4048  F3 E2 0000  JMP  FIND_POSITION
4049  F3 E3 0000  JMP  FIND_POSITION
4050  F3 E4 0000  JMP  FIND_POSITION
4051  F3 E5 0000  JMP  FIND_POSITION
4052  F3 E6 0000  JMP  FIND_POSITION
4053  F3 E7 0000  JMP  FIND_POSITION
4054  F3 E8 0000  JMP  FIND_POSITION
4055  F3 E9 0000  JMP  FIND_POSITION
4056  F3 EA 0000  JMP  FIND_POSITION
4057  F3 EB 0000  JMP  FIND_POSITION
4058  F3 EC 0000  JMP  FIND_POSITION
4059  F3 ED 0000  JMP  FIND_POSITION
4060  F3 EE 0000  JMP  FIND_POSITION
4061  F3 EF 0000  JMP  FIND_POSITION
4062  F3 F0 0000  JMP  FIND_POSITION
4063  F3 F1 0000  JMP  FIND_POSITION
4064  F3 F2 0000  JMP  FIND_POSITION
4065  F3 F3 0000  JMP  FIND_POSITION
4066  F3 F4 0000  JMP  FIND_POSITION
4067  F3 F5 0000  JMP  FIND_POSITION
4068  F3 F6 0000  JMP  FIND_POSITION
4069  F3 F7 0000  JMP  FIND_POSITION
4070  F3 F8 0000  JMP  FIND_POSITION
4071  F3 F9 0000  JMP  FIND_POSITION
4072  F3 FA 0000  JMP  FIND_POSITION
4073  F3 FB 0000  JMP  FIND_POSITION
4074  F3 FC 0000  JMP  FIND_POSITION
4075  F3 FD 0000  JMP  FIND_POSITION
4076  F3 FE 0000  JMP  FIND_POSITION
4077  F3 FF 0000  JMP  FIND_POSITION

---

Appendix A

System BIOS  A-57
FOE 4031 P7:  \_ WRITE_LOOP
4032
4033 \_----- WAIT FOR HORIZONTAL RETRACE
4034
4035 MOV DX, ADDR_6445  \_ GET BASE ADDRESS
4036 4037 PO6  \_ POINT AT STATUS PORT
4038 4039 IN AL, DX  \_ GET STATUS
4040 TEST AL, 1  \_ IS IT LOW
4041 JNZ P6  \_ WAIT UNTIL IT IS
4042 CLI  \_ NO MORE INTERRUPTS
4043 PO8  \_ P9:
4044 IN AL, DX  \_ GET STATUS
4045 TEST AL, 1  \_ IS IT HIGH
4046 JZ P9  \_ WAIT UNTIL IT IS
4047 MOV AX, BL  \_ RECOVER THE CHAR/ATTR
4048 STOS  \_ PUT THE CHAR/ATTR
4049 STI  \_ INTERRUPTS BACK ON
4050 LOOP P7  \_ AS MANY TIMES AS REQUESTED
4051 JMP VIDEO_RETURN
4052 \_ WRITE_C_CURRENT ENDP
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) = RESEG SEGMENT
4064 \_ OUTPUT
4065 \_ NONE
4066 \_ WRITE_C_CURRENT PROC NEAR
4067 MOV AH, 4  \_ IS THIS GRAPHICS
4068 JC P10  \_ IS THIS BM CARD
4069 JMP GRAPHICS_WRITE
4070 JE P10
4071 JMP GRAPHICS_WRITE
4072 JMP DATA_WRITE
4073 P10:
4074 PUSH AX  \_ SAVE ON STACK
4075 PUSH CX  \_ SAVE WRITE COUNT
4076 CALL FIND_POSITION
4077 MOV DI, BX  \_ ADDRESS TO DI
4078 PDP CX  \_ WRITE COUNT
4079 PDP BX  \_ BL HAS CHAR TO WRITE
4080 P1I:
4081 \_ WRITE_LOOP
4082 \_----- WAIT FOR HORIZONTAL RETRACE
4083
4084 MOV DX, ADDR_6445  \_ GET BASE ADDRESS
4085 4086 PO6  \_ POINT AT STATUS PORT
4087 IN AL, DX  \_ GET STATUS
4088 TEST AL, 1  \_ IS IT LOW
4089 JNZ P10  \_ WAIT UNTIL IT IS
4090 CLI  \_ NO MORE INTERRUPTS
4091 PO8  \_ P9:
4092 IN AL, DX  \_ GET STATUS
4093 TEST AL, 1  \_ IS IT HIGH
4094 JE P10  \_ WAIT UNTIL IT IS
4095 MOV AL, [DI]  \_ RECOVER CHAR
4096 STOSB  \_ PUT THE CHAR/ATTR
4097 STI  \_ INTERRUPTS BACK ON
4098 CALL DI,  \_ BUMP POINTER PAST ATTRIBUTE
4099 LOOP P1I  \_ AS MANY TIMES AS REQUESTED
4100 JMP VIDEO_RETURN
4101 \_ WRITE_C_CURRENT ENDP
4102 \_----- WRITE_DOT
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)

A-58 System BIOS
LOC OBJ

LINE  SOURCE

4108  1  CX = COLUMN ( 0-439 ) ( THE VALUES ARE NOT RANGE CHECKED ) :  
4109  1  AL = DOT VALUE TO WRITE (1,2 OR 4 BITS DEPENDING ON MODE ).  
4110  1  REG'D FOR WRITE DOT ONLY; RIGHT JUSTIFIED ) :  
4111  1  BIT 7 OF AL=1 INDICATES HOR THE VALUE INTO THE LOCATION :  
4112  1  DS = DATA SEGMENT :  
4113  1  ES = RESEG SEGMENT :  
4114  1  MOV [AL],BYTE1 :  
4115  1  EXIT :  
4116  1  AL = DOT VALUE READ; RIGHT JUSTIFIED; READ ONLY  

4117  1  I------------------ 

4118  1  ASSUME CS:CODE,DS:DATA,ES:DATA  

4119  1  PROC NEAR READ_DOT  

4120  1  CALL R3  

4121  1  MOV AL,ES:[SI] ; GET THE BYTE  

4122  1  AND AL,AH  

4123  1  SHL AL,CL  

4124  1  MOV CL,ES:[SI] ; LEFT JUSTIFY THE VALUE  

4125  1  OR AL,CL  

4126  1  JMP VIDEO_RETURN ; RETURN FROM VIDEO IO  

4127  1  I------------------ 

4128  1  ENDP : 

4129  1  WRITE_DOT  

4130  1  MOV AX, ES:[SI] ; SAVE DOT VALUE  

4131  1  THICK :  

4132  1  CALL R3  

4133  1  MOV AX, ES:[SI] ; DETERMINE DOT POSITION OF THE DOT  

4134  1  SHR AL,CL  

4135  1  SFR AL,CL  

4136  1  MOV CL,ES:[SI] ; SHIFT TO SET UP THE BITS FOR OUTPUT  

4137  1  OR NOT AH, AH  

4138  1  AND CL,AL  

4139  1  POP BX  

4140  1  JCX R2  

4141  1  IS IT ON  

4142  1  JNZ R2  

4143  1  MOV ES:[SI],AL  

4144  1  OR AL,CL  

4145  1  JP VIDEO_RETURN ; RETURN FROM VIDEO IO  

4146  1  XOR DOT  

4147  1  XOR AL,CL  

4148  1  XOR CL,AL  

4149  1  XOR R1  

4150  1  XOR AX  

4151  1  XOR AX  ; SAVE AX DURING OPERATION  

4152  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4153  1  XOR AX  ; SAVE AX DURING OPERATION  

4154  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4155  1  XOR AX  ; SAVE AX DURING OPERATION  

4156  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4157  1  XOR AX  ; SAVE AX DURING OPERATION  

4158  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4159  1  XOR AX  ; SAVE AX DURING OPERATION  

4160  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4161  1  XOR AX  ; SAVE AX DURING OPERATION  

4162  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4163  1  XOR AX  ; SAVE AX DURING OPERATION  

4164  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4165  1  XOR AX  ; SAVE AX DURING OPERATION  

4166  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4167  1  XOR AX  ; SAVE AX DURING OPERATION  

4168  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4169  1  XOR AX  ; SAVE AX DURING OPERATION  

4170  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4171  1  XOR AX  ; SAVE AX DURING OPERATION  

4172  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4173  1  XOR AX  ; SAVE AX DURING OPERATION  

4174  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4175  1  XOR AX  ; SAVE AX DURING OPERATION  

4176  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4177  1  XOR AX  ; SAVE AX DURING OPERATION  

4178  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4179  1  XOR AX  ; SAVE AX DURING OPERATION  

4180  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4181  1  XOR AX  ; SAVE AX DURING OPERATION  

4182  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4183  1  XOR AX  ; SAVE AX DURING OPERATION  

4184  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4185  1  XOR AX  ; SAVE AX DURING OPERATION  

4186  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4187  1  XOR AX  ; SAVE AX DURING OPERATION  

4188  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4189  1  XOR AX  ; SAVE AX DURING OPERATION  

4190  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4191  1  XOR AX  ; SAVE AX DURING OPERATION  

4192  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4193  1  XOR AX  ; SAVE AX DURING OPERATION  

4194  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4195  1  XOR AX  ; SAVE AX DURING OPERATION  

4196  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4197  1  XOR AX  ; SAVE AX DURING OPERATION  

4198  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4199  1  XOR AX  ; SAVE AX DURING OPERATION  

4200  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4201  1  XOR AX  ; SAVE AX DURING OPERATION  

4202  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

4203  1  XOR AX  ; SAVE AX DURING OPERATION  

4204  1  XOR AX  ; WILL SAVE AX DURING OPERATION  

Appendix A

System BIOS  A-59
F464 B0C002
F465 B0C023
F466 03E6400006
F467 7206
F468 BB80D1
F469 B90357
F470 221A
F471 D3EA
F472 03F2
F473 0AF7
F474 2AC9
F475 D0CB
F476 02CD
F477 7F08
F478 A83
F479 D1EC
F47A 5B
F47B C3

F495
F496 BADA
F497 08C1
F498 E6902
F499 08F8

F4A0 01C20101
F4A4 D6E6
F4A6 D6E6
F4A8 03E6400006

F4A9

LOC OBJ LINE SOURCE

4185 |------------------------------------------------------------------------------------------------------------------|
4186 | SET UP THE REGISTERS ACCORDING TO THE MODE |
4187 | CH = MASK FOR LOW OF COLUMN ADDRESS ( 7/2 FOR HIGH/LOW RES) |
4188 | CL = # OF ADDRESS BITS IN COLUMN VALUE ( 3/2 FOR H/M) |
4189 | BL = MASK TO SELECT BITS FROM POINTED BYTE (0H/1H FOR H/M) |
4190 | BH = NUMBER OF VALID BITS IN POINTED BYTE ( 1/2 FOR H/M) |
4191 |------------------------------------------------------------------------------------------------------------------|
4192 |
4193 | MOV BX,ECH |
4194 | MOV CX,320H |
4195 | CMP CR1,MODE,6 |
4196 | JC BS |
4197 | MOV BX,180H |
4198 | MOV CX,70H |
4199 |
4200 | I----- DETERMINE BIT OFFSET IN BYTE FROM COLUMN MASK |
4201 |
4202 | RS: |
4203 | AND CH,DL |
4204 |
4205 | I----- DETERMINE BYTE OFFSET FOR THIS LOCATION IN COLUMN |
4206 |
4207 | SHR DX,CL |
4208 | ADD SI,DX |
4209 | MOV DH,BH |
4210 |
4211 | I----- MULTIPLY BH (VALID BITS IN BYTE) BY CH (BIT OFFSET) |
4212 |
4213 | SUB CL,CL |
4214 | RS: |
4215 | MOV AL,1 |
4216 | MOV AL,(FOR WRITE) |
4217 | ADD CL,CH |
4218 | DEC BH |
4219 | JNZ R3 |
4220 |
4221 | MOV AH,BL |
4222 | SHR AH,CL |
4223 | MOV BX,SI |
4224 |
4225 | R3 |
4226 | ENDI |
4227 |
4228 | I----- SCROLL UP |
4229 | I----- 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 | BM = FULL VALUE FOR BLANKED LINES |
4234 | AL = # LINES TO SCROLL (AL=0 MEANS BLANK THE ENTIRE |
4235 | FIELD) |
4236 | DS = DATA SEGMENT |
4237 | ES = SEGMENT |
4238 | EXIT |
4239 | NOTHING, THE SCREEN IS SCROLLED |
4240 |
4241 | I----- GRAPHICS_UP PROC NEAR |
4242 | MOV BL,AL |
4243 | MOV AL,AX |
4244 |
4245 | I----- USE CHARACTER SUBROUTINE FOR POSITIONING |
4246 | I----- ADDRESS RETURNED IS MULTIPLIED BY 2 FROM CORRECT VALUE |
4247 |
4248 | CALL GRAPH推SH |
4249 | MOV DI,AX |
4250 |
4251 | I----- DETERMINE SIZE OF WINDOW |
4252 |
4253 | SUB DX,CX |
4254 | ADD DX,10H |
4255 | SAL DH,1 |
4256 | MULH BY 4 |
4257 | SINCE 0 VERT DOTS/CHAR |
4258 | AND EVERY ODD ROWS |
4259 |
4260 | I----- DETERMINE CRT MODE |
4261 |
4262 |
4263 |
4264 |
4265 |
4266 |
4267 |
4268 |
4269 |
4270 |
4271 |
4272 |
4273 |
4274 |
4275 |
4276 |
4277 |
4278 |
4279 |
4280 |
4281 |
4282 |
4283 |
4284 |
4285 |
4286 |
4287 |
4288 |
4289 |
4290 |
4291 |
4292 |
4293 |
4294 |
4295 |
4296 |
4297 |
4298 |
4299 |
4300 |
4301 |
4302 |
4303 |
4304 |
4305 |
4306 |
4307 |
4308 |
4309 |
4310 |
4311 |
4312 |
4313 |
4314 |
4315 |
4316 |
4317 |
4318 |
4319 |
4320 |
4321 |
4322 |
4323 |
4324 |
4325 |
4326 |
4327 |
4328 |
4329 |
4330 |
4331 |
4332 |
4333 |
4334 |
4335 |
4336 |
4337 |
4338 |
4339 |
4340 |
4341 |
4342 |
4343 |
4344 |
4345 |
4346 |
4347 |
4348 |
4349 |
4350 |
4351 |
4352 |
4353 |
4354 |
4355 |
4356 |
4357 |
4358 |
4359 |
4360 |
4361 |
4362 |
4363 |
4364 |
4365 |
4366 |
4367 |
4368 |
4369 |
4370 |
4371 |
4372 |
4373 |
4374 |
4375 |
4376 |
4377 |
4378 |
4379 |
4380 |
4381 |
4382 |
4383 |
4384 |
4385 |
4386 |
4387 |
4388 |
4389 |
4390 |
4391 |
4392 |
4393 |
4394 |
4395 |
4396 |
4397 |
4398 |
4399 |
4400 |
4401 |
4402 |
4403 |
4404 |
4405 |
4406 |
4407 |
4408 |
4409 |
4410 |
4411 |
4412 |
4413 |
4414 |
4415 |
4416 |
4417 |
4418 |
4419 |
4420 |
4421 |
4422 |
4423 |
4424 |
4425 |
4426 |
4427 |
4428 |
4429 |
4430 |
4431 |
4432 |
4433 |
4434 |
4435 |
4436 |
4437 |
4438 |
4439 |
4440 |
4441 |
4442 |
4443 |
4444 |
4445 |
4446 |
4447 |
4448 |
4449 |
4450 |
A-60  System BIOS
FA0E 7504 | FA0E 7505 | FA0E 7506 | FA0E 7507 | FA0E 7508 | FA0E 7509 | FA0E 750A | FA0E 750B | FA0E 750C | FA0E 750D | FA0E 750E | FA0E 750F | FA0E 7510 | FA0E 7511 | FA0E 7512 | FA0E 7513 | FA0E 7514 | FA0E 7515 | FA0E 7516 | FA0E 7517 | FA0E 7518 | FA0E 7519 | FA0E 751A | FA0E 751B | FA0E 751C | FA0E 751D | FA0E 751E | FA0E 751F | FA0E 7520 | FA0E 7521 | FA0E 7522 | FA0E 7523 | FA0E 7524 | FA0E 7525 | FA0E 7526 | FA0E 7527 | FA0E 7528 | FA0E 7529 | FA0E 752A | FA0E 752B | FA0E 752C | FA0E 752D | FA0E 752E | FA0E 752F | FA0E 7530 | FA0E 7531 | FA0E 7532 | FA0E 7533 | FA0E 7534 | FA0E 7535 | FA0E 7536 | FA0E 7537 | FA0E 7538 | FA0E 7539 | FA0E 753A | FA0E 753B | FA0E 753C | FA0E 753D | FA0E 753E | FA0E 753F | FA0E 7540 | FA0E 7541 | FA0E 7542 | FA0E 7543 | FA0E 7544 | FA0E 7545 | FA0E 7546 | FA0E 7547 | FA0E 7548 | FA0E 7549 | FA0E 754A | FA0E 754B | FA0E 754C | FA0E 754D | FA0E 754E | FA0E 754F | FA0E 7550 | FA0E 7551 | FA0E 7552 | FA0E 7553 | FA0E 7554 | FA0E 7555 | FA0E 7556 | FA0E 7557 | FA0E 7558 | FA0E 7559 | FA0E 755A | FA0E 755B | FA0E 755C | FA0E 755D | FA0E 755E | FA0E 755F | FA0E 7560 | FA0E 7561 | FA0E 7562 | FA0E 7563 | FA0E 7564 | FA0E 7565 | FA0E 7566 | FA0E 7567 | FA0E 7568 | FA0E 7569 | FA0E 756A | FA0E 756B | FA0E 756C | FA0E 756D | FA0E 756E | FA0E 756F | FA0E 7570 | FA0E 7571 | FA0E 7572 | FA0E 7573 | FA0E 7574 | FA0E 7575 | FA0E 7576 | FA0E 7577 | FA0E 7578 | FA0E 7579 | FA0E 757A | FA0E 757B | FA0E 757C | FA0E 757D | FA0E 757E | FA0E 757F | FA0E 7580 | FA0E 7581 | FA0E 7582 | FA0E 7583 | FA0E 7584 | FA0E 7585 | FA0E 7586 | FA0E 7587 | FA0E 7588 | FA0E 7589 | FA0E 758A | FA0E 758B | FA0E 758C | FA0E 758D | FA0E 758E | FA0E 758F | FA0E 7590 | FA0E 7591 | FA0E 7592 | FA0E 7593 | FA0E 7594 | FA0E 7595 | FA0E 7596 | FA0E 7597 | FA0E 7598 | FA0E 7599 | FA0E 759A | FA0E 759B | FA0E 759C | FA0E 759D | FA0E 759E | FA0E 759F | FA0E 75A0 | FA0E 75A1 | FA0E 75A2 | FA0E 75A3 | FA0E 75A4 | FA0E 75A5 | FA0E 75A6 | FA0E 75A7 | FA0E 75A8 | FA0E 75A9 | FA0E 75AA | FA0E 75AB | FA0E 75AC | FA0E 75AD | FA0E 75AE | FA0E 75AF | FA0E 75B0 | FA0E 75B1 | FA0E 75B2 | FA0E 75B3 | FA0E 75B4 | FA0E 75B5 | FA0E 75B6 | FA0E 75B7 | FA0E 75B8 | FA0E 75B9 | FA0E 75BA | FA0E 75BB | FA0E 75BC | FA0E 75BD | FA0E 75BE | FA0E 75BF | FA0E 75C0 | FA0E 75C1 | FA0E 75C2 | FA0E 75C3 | FA0E 75C4 | FA0E 75C5 | FA0E 75C6 | FA0E 75C7 | FA0E 75C8 | FA0E 75C9 | FA0E 75CA | FA0E 75CB | FA0E 75CC | FA0E 75CD | FA0E 75CE | FA0E 75CF | FA0E 75D0 | FA0E 75D1 | FA0E 75D2 | FA0E 75D3 | FA0E 75D4 | FA0E 75D5 | FA0E 75D6 | FA0E 75D7 | FA0E 75D8 | FA0E 75D9 | FA0E 75DA | FA0E 75DB | FA0E 75DC | FA0E 75DD | FA0E 75DE | FA0E 75DF | FA0E 75E0 | FA0E 75E1 | FA0E 75E2 | FA0E 75E3 | FA0E 75E4 | FA0E 75E5 | FA0E 75E6 | FA0E 75E7 | FA0E 75E8 | FA0E 75E9 | FA0E 75EA | FA0E 75EB | FA0E 75EC | FA0E 75ED | FA0E 75EE | FA0E 75EF | FA0E 75F0 | FA0E 75F1 | FA0E 75F2 | FA0E 75F3 | FA0E 75F4 | FA0E 75F5 | FA0E 75F6 | FA0E 75F7 | FA0E 75F8 | FA0E 75F9 | FA0E 75FA | FA0E 75FB | FA0E 75FC | FA0E 75FD | FA0E 75FE | FA0E 75FF | System BIOS A-61
A-62 System BIOS
F564 A4
F564 SF
F565 SE
F566 C3
F567
F567 BACA
F569 77
F56A F3
F56B AA
F56C SF
F56D 03C70020
F571 77
F572 BACA
F574 F3
F575 AA
F576 SF
F577 C3
F578
F579 8400
F57A 50
F57B E80401
F57C 0808
F580 58
F581 3C80
F583 7306
F585 8164FA
F586 0E
F589 100F
F590
F591
F592 R10
F593
F594
F595
F596
F597
F598
F599
F600
F601
F602
F603
F604
F605
F606
F607
F608
F609
F610
F611
F612
F613
F614
F615
F616
F617
F618
F619
F620
F621
F622
F623
F624
F625
F626
F627
F628
F629
F62A
F62B
F62C
F62D
F62E
F62F
F630
F631
F632
F633
F634
F635
F636
F637
F638
F639
F63A
F63B
F63C
F63D
F63E
F63F
F640
F641
F642
F643
F644
F645
F646
F647
F648
F649
F64A
F64B
F64C
F64D
F64E
F64F
F650
F651
F652
F653
F654
F655
F656
F657
F658
F659
F65A
F65B
F65C
F65D
F65E
F65F
F660
F661
F662
F663
F664
F665
F666
F667
F668
F669
F66A
F66B
F66C
F66D
F66E
F66F
F670
F671
F672
F673
F674
F675
F676
F677
F678
F679
F67A
F67B
F67C
F67D
F67E
F67F
F680
F681
F682
F683
F684
F685
F686
F687
F688
F689
F68A
F68B
F68C
F68D
F68E
F68F
F690
F691
F692
F693
F694
F695
F696
F697
F698
F699
F69A
F69B
F69C
F69D
F69E
F69F
F700
F701
F702
F703
F704
F705
F706
F707
F708
F709
F70A
F70B
F70C
F70D
F70E
F70F
F710
F711
F712
F713
F714
F715
F716
F717
F718
F719
F71A
F71B
F71C
F71D
F71E
F71F
F720
F721
F722
F723
F724
F725
F726
F727
F728
F729
F72A
F72B
F72C
F72D
F72E
F72F
F730
F731
F732
F733
F734
F735
F736
F737
F738
F739
F73A
F73B
F73C
F73D
F73E
F73F
F740
F741
F742
F743
F744
F745
F746
F747
F748
F749
F74A
F74B
F74C
F74D
F74E
F74F
F750
F751
F752
F753
F754
F755
F756
F757
F758
F759
F760
F761
F762
F763
F764
F765
F766
F767
F768
F769
F76A
F76B
F76C
F76D
F76E
F76F
F770
F771
F772
F773
F774
F775
F776
F777
F778
F779
F77A
F77B
F77C
F77D
F77E
F77F
F780
F781
F782
F783
F784
F785
F786
F787
F788
F789
F78A
F78B
F78C
F78D
F78E
F78F
F790
F791
F792
F793
F794
F795
F796
F797
F798
F799
F79A
F79B
F79C
F79D
F79E
F79F
F7A0
F7A1
F7A2
F7A3
F7A4
F7A5
F7A6
F7A7
F7A8
F7A9
F7AA
F7AB
F7AC
F7AD
F7AE
F7AF
F7B0
F7B1
F7B2
F7B3
F7B4
F7B5
F7B6
F7B7
F7B8
F7B9
F7BA
F7BB
F7BC
F7BD
F7BE
F7BF
F7C0
F7C1
F7C2
F7C3
F7C4
F7C5
F7C6
F7C7
F7C8
F7C9
F7CA
F7CB
F7CC
F7CD
F7CE
F7CF
F7D0
F7D1
F7D2
F7D3
F7D4
F7D5
F7D6
F7D7
F7D8
F7D9
F7DA
F7DB
F7DC
F7DD
F7DE
F7DF
F7E0
F7E1
F7E2
F7E3
F7E4
F7E5
F7E6
F7E7
F7E8
F7E9
F7EA
F7EB
F7EC
F7ED
F7EE
F7EF
F7F0
F7F1
F7F2
F7F3
F7F4
F7F5
F7F6
F7F7
F7F8
F7F9
F7FA
F7FB
F7FC
F7FD
F7FE
F7FF

---

Appendix A

System BIOS A-63
<table>
<thead>
<tr>
<th>LOC</th>
<th>OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>F080</td>
<td>23C3</td>
<td>4566</td>
<td>AND AX,BX</td>
</tr>
<tr>
<td>F080</td>
<td>F9C280</td>
<td>4567</td>
<td>TEST DL,OH</td>
</tr>
<tr>
<td>F080</td>
<td>76A0</td>
<td>4568</td>
<td>JZ SI,0</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50020</td>
<td>4569</td>
<td>XOR AH,ES:[DI+200H]</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4570</td>
<td>XOR AL,ES:[DI+200H]</td>
</tr>
<tr>
<td>F080</td>
<td>B8550020</td>
<td>4571</td>
<td>SI,0</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4572</td>
<td>MOV ES:[DI+200H],AX</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4573</td>
<td>MOV ES:[DI+200H]+1,AL</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4574</td>
<td>ADD DI,0</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4575</td>
<td>DEC DI</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4576</td>
<td>JNZ SP</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4577</td>
<td>POP SI</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4578</td>
<td>POP DI</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4579</td>
<td>INC DI</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4580</td>
<td>INC DI</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4581</td>
<td>LOOP 50</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4582</td>
<td>JMP VIDEO_RETURN</td>
</tr>
</tbody>
</table>

**Appendix A**

<table>
<thead>
<tr>
<th>LOC</th>
<th>OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4583</td>
<td>GRAPHICS_WRITE</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4584</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4585</td>
<td>i GRAPHICS_READ</td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4586</td>
<td></td>
</tr>
</tbody>
</table>

**Other instructions and comments**

<table>
<thead>
<tr>
<th>LOC</th>
<th>OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4587</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4588</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4589</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4590</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4591</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4592</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4593</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4594</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4595</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4596</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4597</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4598</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4599</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4600</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4601</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4602</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4603</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4604</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4605</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4606</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4607</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4608</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4609</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4610</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4611</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4612</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4613</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4614</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4615</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4616</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4617</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4618</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4619</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4620</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4621</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4622</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4623</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4624</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4625</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4626</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4627</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4628</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4629</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4630</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4631</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4632</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4633</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4634</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4635</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4636</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4637</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4638</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4639</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4640</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4641</td>
<td></td>
</tr>
<tr>
<td>F080</td>
<td>2632A50010</td>
<td>4642</td>
<td></td>
</tr>
</tbody>
</table>

**System BIOS A-65**
LOC OBJ | LINE | SOURCE
--- | --- | ---
F67A | 4643 | S16:  
F67A 16 | 4644 | PUSH DS  
F67A 17 | 4645 | POP DS  
F67C B4000 | 4646 | MOV DX,12B  
F67F | 4647 | S17:  
F67F 56 | 4648 | PUSH SI  
F680 57 | 4649 | PUSH DI  
F681 B9000 | 4650 | MOV CX,8  
F684 53 | 4651 | REP MOVSB  
F685 46 | 4652 | POP DI  
F687 8E | 4653 | POP SI  
F68A 7AE | 4654 | JE SI  
F68A F6C0 | 4655 | INC AL  
F68C 03C700 | 4656 | ADD DI,0  
F68F 4A | 4657 | DEC DX  
F690 75D0 | 4658 | JNZ SI  
F698 | 4659 | DO ALL OF THEM  
F699 | 4660 |  
F699 3C00 | 4661 | CMP AL,0  
F69A 7412 | 4662 | JE SI  
F69A 2B00 | 4663 | SUB AX,AX  
F69B 82D0 | 4664 | MOV DS,AX  
F69A C5E7C00 | 4665 |  
F69A 4C0 | 4666 | LES DI,EXT_PTR  
F69A B8C7 | 4667 | MOV AX,ES  
F69B 7404 | 4668 | OR AX,01  
F69C B0D8 | 4669 | JE SI  
F69C D0B8 | 4670 | MOV AL,12B  
F69D E9F8 | 4671 | JMP SI  
F69E | 4672 |  
F699 | 4673 | ASSUME DS:DATA  
F69G | 4674 |  
F69A 80:  
F69A 8B3408 | 4675 | ADD SP,4  
F69A E917F8 | 4676 | JMP VIDEO_RETURN  
F69A | 4677 | ALL DONE  
F69B | 4678 |  
F69B | 4679 |  
F69B | 4680 |  
F69B | 4681 |  
F69B | 4682 |  
F69B | 4683 |  
F69B | 4684 |  
F69B | 4685 |  
F69B | 4686 |  
F69B | 4687 |  
F69B | 4688 |  
F69B | 4689 |  
F69B | 4690 |  
F69B | 4691 |  
F69B | 4692 |  
F69B | 4693 |  
F69B | 4694 |  
F69B | 4695 |  
F69B | 4696 |  
F69B | 4697 |  
F69B | 4698 |  
F69B | 4699 |  
F69B | 4700 |  
F69B | 4701 |  
F69B | 4702 |  
F69B | 4703 |  
F69B | 4704 |  
F69B | 4705 |  
F69B | 4706 |  
F69B | 4707 |  
F69B | 4708 |  
F69B | 4709 |  
F69B | 4710 |  
F69B | 4711 |  
F69B | 4712 |  
F69B | 4713 |  
F69B | 4714 |  
F69B | 4715 |  
F69B | 4716 |  
F69B | 4717 |  
F69B | 4718 |  
--- | --- | ---
A-66 System BIOS
<table>
<thead>
<tr>
<th>LOC OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>F46D  E309</td>
<td>4719</td>
<td>AND BX,CK</td>
</tr>
<tr>
<td>F46F  0803</td>
<td>4720</td>
<td>OR DX,BX</td>
</tr>
<tr>
<td>F461  D1E9</td>
<td>4721</td>
<td>SHL AX,1</td>
</tr>
<tr>
<td>F463  D1E1</td>
<td>4722</td>
<td>SHL CX,1</td>
</tr>
<tr>
<td>F465  0808</td>
<td>4723</td>
<td>MOV BX,AX</td>
</tr>
<tr>
<td>F467  2309</td>
<td>4724</td>
<td>AND BX,CK</td>
</tr>
<tr>
<td>F469  0803</td>
<td>4725</td>
<td>OR DX,BX</td>
</tr>
<tr>
<td>F468  D1E1</td>
<td>4726</td>
<td>SHL CX,1</td>
</tr>
<tr>
<td>F46B  730C</td>
<td>4727</td>
<td>MOV AX,DX</td>
</tr>
<tr>
<td>F46F  88C2</td>
<td>4728</td>
<td>POP BX</td>
</tr>
<tr>
<td>F461  5B</td>
<td>4729</td>
<td>POP CX</td>
</tr>
<tr>
<td>F462  99</td>
<td>4730</td>
<td>POP DK</td>
</tr>
<tr>
<td>F464  C3</td>
<td>4731</td>
<td>RET</td>
</tr>
</tbody>
</table>

**ENDP**

<table>
<thead>
<tr>
<th>LOC OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>F465</td>
<td>4734</td>
<td>ENDP</td>
</tr>
</tbody>
</table>

**END**

### Appendix A

**PROC READ BYTE**

1. **THIS ROUTINE WILL TAKE 2 BYTES FROM THE SEGEN**
2. **BUFFER, COMPARE AGAINST THE CURRENT FOREGROUND**
3. **COLOR, AND PLACE THE CORRESPONDING ON/OF BIT**
4. **PATTERN INTO THE CURRENT POSITION IN THE SAVE**
5. **AREA**
6. **ENTRY**
7. **SI.05 = POINTER TO SEGEN AREA OF INTEREST**
8. **BX = EXPANDED FOREGROUND COLOR**
9. **BP = POINTER TO SAVE AREA**
10. **EXIT**

<table>
<thead>
<tr>
<th>LOC OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>F465</td>
<td>4734</td>
<td>MOV AH,(SI)</td>
</tr>
<tr>
<td>F466</td>
<td>4735</td>
<td>MOV AL,(SI+1)</td>
</tr>
<tr>
<td>F467</td>
<td>4736</td>
<td>MOV CX,DX</td>
</tr>
<tr>
<td>F468</td>
<td>4737</td>
<td>MOV DL,0</td>
</tr>
</tbody>
</table>

**ENDP**

### PROC READ GRAPHICS

1. **THIS ROUTINE TAKES THE CURSOR POSITION**
2. **CONTAINED IN THE MEMORY LOCATION, AND CONVERTS IT INTO AN OFFSET INTO THE**
3. **SEGREG BUFFER, ASSUMING ONE BYTE/DOT**
4. **FOR MEDIUM RESOLUTION GRAPHICS, THE NUMBER MUST BE DOUBLED.**
5. **ENTRY**
6. **NO REGISTERS, MEMORY LOCATION**
7. **CURSOR_POSN IS USED**
8. **EXIT**

<table>
<thead>
<tr>
<th>LOC OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>F702</td>
<td>4781</td>
<td>MOV AX,CURSOR_POSN</td>
</tr>
</tbody>
</table>

**ENDP**

### PROC SAVE GRAPHICS

1. **GET CURRENT CURSOR**
2. **SAVE REGISTER**
3. **SAVE A COPY OF CURRENT CURSOR**
4. **SET REGS TO OLD**
5. **MULTIPLY BY BYTES/COLUMN**
6. **MULTIPLY # 4 SINCE # BYTES/BYTE**
7. **ISOLATE COLUMN VALUE**
8. **DETERMINE OFFSET**
9. **RECOVER POINTER**

<table>
<thead>
<tr>
<th>LOC OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>F714</td>
<td>4794</td>
<td>MOV AX,BX</td>
</tr>
</tbody>
</table>

**ENDP**

### System BIOS A-67
6795  | WRITE_TTY:
6796  |  
6797  |  THIS INTERFACE PROVIDES A TELETYPewriter INTERFACE TO THE VIDEO:
6798  |  
6799  |  CARD. THE INPUT CHARACTER IS WRITTEN TO THE CURRENT CURSOR:
6800  |  
6801  |  POSITION, AND THE CURSOR IS MOVED TO THE NEXT POSITION, IF THE:
6802  |  
6803  |  CURSOR LEAVES THE LAST COLUMN OF THE FIELD, THE COLUMN IS SET:
6804  |  TO ZERO, AND THE ROW VALUE IS INCREMENTED. IF THE ROW VALUE:
6805  |  LEAVES THE FIELD, THE CURSOR IS PLACED ON THE LAST ROW, FIRST:
6806  |  
6807  |  COLUMN, AND THE ENTIRE SCREEN IS SCROLLED UP ONE LINE, WHEN:
6808  |  
6809  |  THE SCREEN IS SCROLLED UP, THE ATTRIBUTE FOR FILLING THE NEWLY:
6810  |  BLANKED LINE IS READ FROM THE CURSOR POSITION ON THE PREVIOUS:
6811  |  
6812  |  LINE BEFORE THE SCROLL. IN CHARACTER MODE, IN GRAPHICS MODE:
6813  |  
6814  |  THE 0 COLOR IS USED.
6815  |  
6816  |  ENTRY:
6817  |  
6818  |  (AK) = CURRENT CRT MODE
6819  |  
6820  |  (AL) = CHARACTER TO BE WRITTEN
6821  |  
6822  |  NOTE THAT BACK SPACE, CAR RET, BELL AND LINE FEED ARE HANDLED
6823  |  
6824  |  AS COMMANDS RATHER THAN AS DISPLAYABLE GRAPHICS:
6825  |  
6826  |  
6827  |  (BL) = FOREGROUND COLOR FOR CHAR WRITE IF CURRENTLY IN A:
6828  |  
6829  |  
6830  |  
6831  |  
6832  |  ALL REGISTERS SAVED
6833  |  
6834  |  
6835  |  
6836  |  
6837  |  
6838  |  
6839  |  
6840  |  
6841  |  
6842  |  
6843  |  
6844  |  
6845  |  
6846  |  
6847  |  
6848  |  
6849  |  
6850  |  
6851  |  
6852  |  
6853  |  
6854  |  
6855  |  
6856  |  
6857  |  
6858  |  
6859  |  
6860  |  
6861  |  
6862  |  
6863  |  
6864  |  
6865  |  
6866  |  
6867  |  
6868  |  
6869  |  
6870  |  
6871  |  
6872  |  
6873  |  
6874  |  
6875  |  
6876  |  
6877  |  
6878  |  
6879  |  
6880  |  
6881  |  
6882  |  
6883  |  
6884  |  
6885  |  
6886  |  
6887  |  
6888  |  
6889  |  
6890  |  
6891  |  
6892  |  
6893  |  
6894  |  
6895  |  
6896  |  
6897  |  
6898  |  
6899  |  
6900  |  
6901  |  
6902  |  
6903  |  
6904  |  
6905  |  
6906  |  
6907  |  
6908  |  
6909  |  
6910  |  
6911  |  
6912  |  
6913  |  
6914  |  
6915  |  
6916  |  
6917  |  
6918  |  
6919  |  
6920  |  
6921  |  
6922  |  
6923  |  
6924  |  
6925  |  
6926  |  
6927  |  
6928  |  
6929  |  
6930  |  
6931  |  
6932  |  
6933  |  
6934  |  
6935  |  
6936  |  
6937  |  
6938  |  
6939  |  
6940  |  
6941  |  
6942  |  
6943  |  
6944  |  
6945  |  
6946  |  
6947  |  
6948  |  
6949  |  
6950  |  
6951  |  
6952  |  
6953  |  
6954  |  
6955  |  
6956  |  
6957  |  
6958  |  
6959  |  
6960  |  
6961  |  
6962  |  
6963  |  
6964  |  
6965  |  
6966  |  
6967  |  
6968  |  
6969  |  
6970  |  
6971  |  
6972  |  
A-68 System BIOS
<table>
<thead>
<tr>
<th>LOC OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>F769 50D0</td>
<td>4873</td>
<td>MOV AH, 69h</td>
</tr>
<tr>
<td>F765 32C9</td>
<td>4874</td>
<td>SUB CX, CX</td>
</tr>
<tr>
<td>F765 B416</td>
<td>4875</td>
<td>MOV DH, 26</td>
</tr>
<tr>
<td>F767 4A164A00</td>
<td>4876</td>
<td>MOV DL, BYTE PTR CRT_COLS</td>
</tr>
<tr>
<td>F768 FECA</td>
<td>4877</td>
<td>DEC DL</td>
</tr>
<tr>
<td>F760</td>
<td>4878</td>
<td></td>
</tr>
<tr>
<td>F760 CD10</td>
<td>4879</td>
<td></td>
</tr>
<tr>
<td>F76F</td>
<td>487A</td>
<td></td>
</tr>
<tr>
<td>F76F 5A</td>
<td>487B</td>
<td></td>
</tr>
<tr>
<td>F775 E9054A</td>
<td>487C</td>
<td></td>
</tr>
<tr>
<td>F775</td>
<td>487D</td>
<td></td>
</tr>
<tr>
<td>F773 FECA</td>
<td>487E</td>
<td></td>
</tr>
<tr>
<td>F772</td>
<td>487F</td>
<td></td>
</tr>
<tr>
<td>F772 5402</td>
<td>4880</td>
<td>MOV AH, 2</td>
</tr>
<tr>
<td>F777 EB74</td>
<td>4881</td>
<td>JMP UA</td>
</tr>
<tr>
<td>F779</td>
<td>4882</td>
<td></td>
</tr>
<tr>
<td>F779 507A00</td>
<td>4883</td>
<td>CHRP DL, 0</td>
</tr>
<tr>
<td>F77C 74F7</td>
<td>4884</td>
<td>JE U7</td>
</tr>
<tr>
<td>F77E FECA</td>
<td>4885</td>
<td>DEC DL</td>
</tr>
<tr>
<td>F780 EB73</td>
<td>4886</td>
<td>JMP U7</td>
</tr>
<tr>
<td>F780</td>
<td>4887</td>
<td></td>
</tr>
<tr>
<td>F782</td>
<td>4888</td>
<td></td>
</tr>
<tr>
<td>F782 D200</td>
<td>4889</td>
<td>MOV DL, 0</td>
</tr>
<tr>
<td>F784 EB7F</td>
<td>488A</td>
<td>JMP U7</td>
</tr>
<tr>
<td>F784</td>
<td>488B</td>
<td></td>
</tr>
<tr>
<td>F786</td>
<td>488C</td>
<td></td>
</tr>
<tr>
<td>F786 D0F18</td>
<td>488D</td>
<td>CHRP DH, 24</td>
</tr>
<tr>
<td>F789 750B</td>
<td>488E</td>
<td>JNE U6</td>
</tr>
<tr>
<td>F78B EB7C</td>
<td>488F</td>
<td>JMP U1</td>
</tr>
<tr>
<td>F78B</td>
<td>4890</td>
<td></td>
</tr>
<tr>
<td>F78B</td>
<td>4891</td>
<td></td>
</tr>
<tr>
<td>F78B</td>
<td>4892</td>
<td></td>
</tr>
<tr>
<td>F78B</td>
<td>4893</td>
<td></td>
</tr>
<tr>
<td>F78B</td>
<td>4894</td>
<td></td>
</tr>
<tr>
<td>F78B</td>
<td>4895</td>
<td></td>
</tr>
<tr>
<td>F78B</td>
<td>4896</td>
<td></td>
</tr>
<tr>
<td>F78B</td>
<td>4897</td>
<td></td>
</tr>
<tr>
<td>F78B</td>
<td>4898</td>
<td></td>
</tr>
<tr>
<td>F78B</td>
<td>4899</td>
<td></td>
</tr>
<tr>
<td>F78B</td>
<td>489A</td>
<td></td>
</tr>
<tr>
<td>F78B</td>
<td>489B</td>
<td></td>
</tr>
<tr>
<td>F78C</td>
<td>489C</td>
<td></td>
</tr>
<tr>
<td>F794 03</td>
<td>489D</td>
<td>MOV AH, 0</td>
</tr>
<tr>
<td>F794 05</td>
<td>489E</td>
<td>MOV DX, ADDR_LIGHT</td>
</tr>
<tr>
<td>F797 03</td>
<td>489F</td>
<td>ADD DI, 6</td>
</tr>
<tr>
<td>F799 04</td>
<td>48A0</td>
<td></td>
</tr>
<tr>
<td>F79C 84C0</td>
<td>48A1</td>
<td>MOV DH, 0</td>
</tr>
<tr>
<td>F79E B816300</td>
<td>48A2</td>
<td>MOV AX, ADDR_LIGHT</td>
</tr>
<tr>
<td>F7A2 3C206</td>
<td>48A3</td>
<td>ADD DX, 6</td>
</tr>
</tbody>
</table>

Appendix A

System BIOS A-69
F7A5 EC 4963 IN AL,DX
F7A6 A004 4964 TEST AL,4
F7A6 757E 4965 JNZ V6
F7A7 946E 4966
F7AA A0D2 4966 TEST AL,E
F7AC 7503 4969 JZ V7A
F7AE E90180 4971 JMP V7
F7A1 4972
F7B1 4955 VTA:
F7B1 D410 4956 MOV AH,AH
F7B2 4957 ; LIGHT PEN REGISTERS ON 6845
F7B3 B1663D0 4958 MOV DX,ADDR_6845
F7B7 BAC4 4959 MOV AL,AL
F7B9 EE 4960 ; REGISTER TO ADJUST FOR 6845
F7BA 42 4961 OUT DX,AL
F7BB EC 4962 INC DX
F7BC BAEB 4963 MOV CH,AL
F7BE 4A 4964 DEC DX
F7BF F6EC 4965 INC AH
F7C1 BAC4 4966 MOV AL,AL
F7C3 EE 4967 ; SECOND DATA REGISTER
F7C4 42 4969 OUT DX,AL
F7C6 EC 4971 IN AL,DX
F7C8 BAE5 4972 MOV AH,CH
F7C9 4973 ; AX HAS INPUT VALUE
F7CD BA14900 4974 MOV BL,CRT_MODE
F7CC 2AFF 4975 SUB BH,BH
F7CE 2EB0F067 4976 MOV BL,CRT_MODE
F7D0 B3C3 4977 ; DETERMINE AMOUNT TO SUBTRACT
F7D3 8814E000 4978 MOV BX,CRT_MODE
F7D5 722A 4979 MOV CX,AX
F7D7 9927 4980 MOV CX,CRT_MODE
F7D9 7902 4981 MOV AX,AX
F7DA B2C0 4982 MOV AX,AX
F7E0 4983 ; IF positive, determine mode
F7E1 4984 ; DETERMINE MODE OF OPERATION
F7E2 B103 4985 MOV CL,3
F7E3 603E90000 4986 CMP CRT_MODE,4
F7E5 722A 4987 JB V4
F7E6 803E90007 4988 CMP CRT_MODE,7
F7EA 7423 4989 JE V4
F7EB 4990 ; ALPHA PEN
F7EF 4991 ; DETERMINE GRAPHICS MODE
F7F1 B2D8 4992 MOV DL,40
F7F3 FA72 4993 DIV DL
F7F9 4994 ; DETERMINE ROMALI AND COLUMN(AM)
F7FC 5000 4995 ; AL RANGE 0-59, AM RANGE 0-39
F7F7 BAE8 5003 MOV CH,AL
F7F7 02CD 5004 ADD CH,CH
F7F8 BAC4 5005 MOV BH,BH
F7F9 2AFF 5006 MOV BL,BH
F7FC 5007 CMP CRT_MODE,6
F802 7504 5008 JNE V3
F804 B106 5009 MOV CL,4
F806 D064 5010 SAL AH,1
F808 5011 V3: SHR BX,CL
F80A B3E3 5012 ; MULTIPLE #16 FOR HIGH RES
F80C BAFD 5013 ; DETERMINE GRAPHIC ROW POSITION
F80E 5014 ; DETERMINE ALPHA CHAR POSITION
F800 BAF4 5016 MOV DL,AL
F80C BAF0 5017 MOV DL,AL
F80E 5018 SHR DL,1
F810 5019 SHR DL,1

A-70 System BIOS
LOC OBJ  
LINE  
SOURCE  

F812 EB12  
5820  
5821  
5822  
5823  
5824  
5825  
5826  
5827  
5828  
5829  
5830  
5831  
5832  
5833  
5834  
5835  
5836  
5837  
5838  
5839  
5840  
5841  
5842  
5843  
5844  
5845  
5846  
5847  
5848  
5849  
5850  
5851  
5852  
5853  
5854  
5855  
5856  
5857  
5858  
5859  
5860  
5861  
5862  
5863  
5864  
5865  
5866  
5867  
5868  
5869  
5870  
5871  
5872  
5873  
5874  
5875  
5876  
5877  
5878  
5879  
5880  
5881  
5882  
5883  
5884  
5885  
5886  
5887  
5888  
5889  
5890  
5891  
5892  
5893  
5894  
5895  
5896  

PHA  
F636A400  
F810 65F0  
F81A 6404  
F81C 0200  
F81E 9A88  
F820 BDCC  
F822 32FF  
F824 D3E3  
F826 5833  
F826 5834  
F828 5835  
F828 5836  
F829 5837  
F829 5838  
F82A 5839  
F82A 5840  
F82C 5841  
F832 5842  
F833 5843  
F834 5844  
F835 5845  
F836 5846  
F837 5847  
F838 5848  
F839 5849  
F83C 5850  
F83F 5851  

--- INT 12 ---  
MEMORY_SIZE_DET  
004  THIS ROUTINE DETERMINES THE AMOUNT OF MEMORY IN THE SYSTEM  
005  AS REPRESENTED BY THE SWITCHES ON THE PLANAR. NOTE THAT THE  
006  SYSTEM MAY NOT BE ABLE TO USE 1/0 MEMORY UNLESS THERE IS A FULL  
007  COMPLEMENT OF 64K BYTES ON THE PLANAR.  
008  INPUT  
009  NO REGISTERS  
010  THE MEMORY_SIZE_VARIABLE IS SET DURING POWER ON DIAGNOSTICS  
011  ACCORDING TO THE FOLLOWING HARDWARE ASSUMPTIONS:  
012  PORT 68 BITS 3-2 = 00 = 16K BASE RAM  
013  01 = 32K BASE RAM  
014  10 = 64K BASE RAM  
015  PORT 62 BITS 3-0 INDICATE AMOUNT OF 1/0 RAM IN 32K INCREMENTS  
016  E.G., 0000 = NO RAM IN 1/0 CHANNEL  
017  0001 - 64K RAM IN 1/0 CHANNEL, ETC.  
018  OUTPUT  
019  (AX) = NUMBER OF CONTIGUOUS 1K BLOCKS OF MEMORY  
020  ASSUME CS:CODE, DS:DATA  
021  ORG 000AH  
MEMORY_SIZE_DET PROC  
STI  
PUSH DS  
CALL 01030H  
MOV AX,MEMORY_SIZE  
PUSH DS  
PUSH AX  
INT 21  
INT 15  
PUSH AX  
RET  
MEMORY_SIZE_DET ENDP  
--- INT 11 ---  
EQUIPMENT DETERMINATION  
THIS ROUTINE ATTEMPTS TO DETERMINE WHAT OPTIONAL DEVICES ARE ATTACHED TO THE SYSTEM.  
INPUT  
NO REGISTERS  
THE EQUIP_FLAG VARIABLE IS SET DURING THE POWER ON DIAGNOSTICS USING THE FOLLOWING HARDWARE ASSUMPTIONS:  
PORT 68 = LOW ORDER BYTE OF EQUIPMENT  
PORT 3F8 = INTERRUPT ID REGISTER OF 8550  
BITS 7-3 ARE ALWAYS 0  
PORT 378 = OUTPUT PORT OF PRINTER -- 855 PORT THAT CAN BE READ AS WELL AS WRITTEN  
OUTPUT  
A-71
(AX) IN SET: BIT SIGNIFICANT TO INDICATE ATTACHED I/O:

BIT 15,14 = NUMBER OF PRINTERS ATTACHED:

BIT 13 NOT USED:

BIT 12 = GAME I/O ATTACHED:

BIT 11,10,9 = NUMBER OF KGIO CARDS ATTACHED:

BIT 8 UNUSED:

BIT 7,6 = NUMBER OF DISKETTE DRIVES:

BIT 0-4 = INITIAL VIDEO MODE:

BIT 0 = 0 - GRAYED:

BIT 1 = 01 - BOXES SM USING COLOR CARD:

BIT 2 = 10 - BOXES SM USING COLOR CARD:

BIT 3 = 11 - BOXES SM USING SM CARD:

BIT 4 = 3.2 FLOOR RAM SIZE (00=16K, 01=32K, 10=64K, 11=128K):

BIT 5 = I/O NOT USED:

BIT 6 = 0 = TRK FROM DISKETTE -- THIS BIT INDICATES THAT:

BIT 7 = THERE ARE DISKETTE DRIVES ON THE SYSTEM:

BIT 9 = NO OTHER REGISTERS AFFECTED:

ASSUME CS:CODE, DS:DATA

EQUIPMENT PROC FAR

STI INTERRUPTS BACK ON

PUSH DS SAVE SEGMENT REGISTER

CALL DDS

MOV AX,VRG.FLAG:

GET THE CURRENT SETTINGS

PDP DS RECOVER SEGMENT

IRET RETURN TO CALLER

EQUIPMENT ENDP

; --- INT 1 ---

; DUMMY CASSETTE ID ROUTINE--RETURNS 'INVALID CMD' IF THE ROUTINE IS:

; EVER CALLED BY ACCIDENT (AH=666, CARRY FLAG=1):

; ---

ORG 0F889H

CASSETTE_ID PROC FAR

CARRY INDICATOR=1

MOV AX, 86H

POP E

CASSETTE_ID ENDP

; ---

; NON-HASIBLE INTERRUPT ROUTINE:

; THIS ROUTINE WILL PRINT A PARITY CHECK 1 OR 2 MESSAGE:

; AND ATTEMPT TO FIND THE STORAGE LOCATION CONTAINING THE:

; BAD PARITY. IF FOUND, THE SEGMENT ADDRESS WILL BE:

; PRINTED. IF NO PARITY ERROR CAN BE FOUND (INTERMITTANT:

; READ PROBLEM) ?????----ON THE ADDRESS THE ADDRESS:

; WOULD NORMALLY GO:

; IF ADDRESS IN ERROR IS IN THE I/O EXPANSION BOX, THE:

; ADDRESS WILL BE FOLLOWED BY A '(B)' IF IN SYSTEM UNIT,

; A '(B)' WILL FOLLOW THE ADDRESS:

ORG_INT PROC NEAR

ASSUME DS:DATA

PUSH AX SAVE ORIG CONTENTS OF AX

IN AL,PORT_C:

TEST AL,6CH:

JNZ H1:

JMP D14:

H1:

MOV DX:DATA:

MOV DS,DX:

MOV SI,OFFSET D1 ADDR OF ERROR MSG:

TEST AL,4OH:

JNZ D13:

DISPLAY ERROR MSG:

MOV SI,OFFSET D2 MUST BE PLAIN:

MOV AH,0 INIT AND SET MODE FOR VIDEO:

MOV AL,CFG_MODE:

CALL P.MSG PRINT ERROR MSG:

D13:

INT 10H:

CALL ERROR ID PROCEDURE

D14:

SEE IF LOCATION THAT CAUSED PARITY CHECK CAN BE FOUND

A-72 System BIOS
LOC OBJ   LINE   SOURCE
F80F 313031 5250   E0   DB  '101',13,10   SYSTEM BOARD ERROR
F902 00   5251   E1   DB  ' 201',13,10   MEMORY ERROR
F903 0A   5252   F2A   DB  'ROM',13,10   ROM CHECKSUM ERROR
F906 0D   5253   F3C   DB  '3001',13,10   EXPANSION IO BOX ERROR
F913 0D   5254   D1   DB  'PARITY CHECK 2',13,10
F916 0A   5255   D2   DB  'PARITY CHECK 1',13,10
F918 50415249545920 5256   D2A   DB  '?????',13,10
F933 0D   5257
F934 0A   5258
F935 383F3F3F3F 5259
F936 00   5260
F939 0A   5261

ASSUME DS:DATA
BLINK_INT PROC NEAR
F93C FB   5262
F93D 50   5263
F93E E461 5264   STI
F940 0A60 5265   PUSH AX   SAVE AX REG CONTENTS
F942 F400 5266   IN AL,PORT_B   READ CURRENT VAL OF PORT B
F944 2440 5267   MOV AH,AL
F946 2440 5268   NOT AL   FLIP ALL BITS
F948 4004F7 5269   AND AL,0000000B   MASK OUT OF ORIGINAL VAL
F94A 0407 5270   OR AH,AL   OR HEX CONTROL BIT IN
F94C E661 5271   MOV AL,01111110   OR HEX CONTROL BIT IN
F94E 8A00 5272   MOV PORT_B,AL
F950 E200 5273   OR AL,01111110   OR HEX CONTROL BIT IN
F952 5160 5274   MOV AX,AL
F954 CB   5275   POP AX   RESTORE AX REG
F955 CF   5276   IRET
5277   BLINK_INT ENDP
5278
5279

THIS ROUTINE CHECKSUMS OPTIONAL ROM MODULES AND:
IF CHECKSUM IS OK, CALLS INIT/TEST CODE IN MODULE

ROM_CHECK PROC NEAR
F95B 004000 5280   MOV AX,DATA   POINT ES TO DATA AREA
F95F 0EC0   5281   MOV ES,AX
F960 0A64   5282   MOV AL,10111110   GET LENGTH INDICATOR
F964 0A6202 5283   MOV CL,0FH   MULTIPLY BY $2
F969 03E0 5284   SHL AX,CL
F96D 80C8 5285   MOV CX,AX
F96F 51    5286   PUSH CX   SAVE COUNT
F971 0B6008 5287   MOV CX,A   ADJUST
F973 0B61E8 5288   SNAR AX,CL
F976 0B5200 5289   ADD DX,AX   SET POINTER TO NEXT MODULE
F979 59    5290   POP CX   RETRIEVE COUNT
F97C 7C1F00 5291   CALL ROM_CHECK_MNT   DO CHECKSUM
F97F 7C06 5292   JZ ROM_CHECK-1
F981 0B87ED 5293   CALL ROM_ERR   POST CHECKSUM ERROR
F984 0B1490 5294   JMP ROM_CHECK_END   AND EXIT
F987 0B30F2 5295   ROM_CHECK-1:
F98A 0B5100 5296   PUSH DX   SAVE POINTER
F98D 0B7C0677002308 5297   MOV EB:12_ROM_INIT,0030H   LOAD OFFSET
F990 0B5C163900 5298   MOV EB:12_ROM_SEG,03   LOAD SEGMENT
F994 0B12716700 5299   CALL EB:12_ROM_INIT   CALL INIT/TEST ROUTINE
F997 8A    5300   POP DX
5302   ROM_CHECK_END ENDP
5303
5304
5306
5307
5308
5309

RETURN TO CALLER

A-74 System BIOS
LOC OBJ       LINE       SOURCE

5310      I ---------- CONVERT AND PRINT ASCII CODE ---------- I
5311      I AL MUST CONTAIN NUMBER TO BE CONVERTED : I
5312      I AX AND BX DESTROYED. : I
5314      I ----------

FND    5315      XPC_BYTE PROC NEAR
FND    5316      PUSH AX
FND    5317      MOV CL,4
FND    5318      SHR AL,CL
FND    5319      CALL XLAT_PR
FND    5320      POP AX
FND    5321      AND AL,0FH
FND    5322      I SAVE FOR LOW NIBBLES DISPLAY I
FND    5323      I SHIFT COUNT I
FND    5324      I HYDRO SNAP I
FND    5325      I DO THE HIGH NIBBLES DISPLAY I
FND    5326      I RECOVER THE NIBBLES I
FND    5327      I ISOLATE TO LOW NIBBLE I
FND    5328      I FALL INTO LOW NIBBLE CONVERSION I
FND    5329      I CONVERT 00-0F TO ASCII CHARACTER I
FND    5330      I ADD FIRST CONVERSION FACTOR I
FND    5331      I ADJUST FOR NUMERIC AND ALPHANUM RANGES I
FND    5332      I ADD MORE AND ADJUST LOW NIBBLES I
FND    5333      I ADD HIGH NIBBLE TO ASCII RANGE I
FND    5334      I DISPLAY CHARACTER IN AL I
FND    5335      MOV AH,14
FND    5336      MOV BH,0
FND    5337      INT 10H
FND    5338      I CALL VIDEO_IO I
FND    5339      I PRINTING SOURCE TABLE I
FND    5340      I ENTRY REQUIREMENTS: I
FND    5341      I SI = OFFSET(ADDRESS) OF MESSAGE BUFFER I
FND    5342      I CX = MESSAGE BYTE COUNT I
FND    5343      I MAXIMUM MESSAGE LENGTH IS 36 CHARACTERS I
FND    5344      I ----------
FND    5345      E MSG PROC NEAR
FND    5346      MOV BP,SI
FND    5347      CALL P_MSG
FND    5348      PUSH DS
FND    5349      CALL OOS
FND    5350      MOV AL,BYTE PTR EQUIP_FLAG
FND    5351      I LOOP/HALT ON ERROR I
FND    5352      AND AL,01H
FND    5353      JNZ 012
FND    5354      I NO - RETURN I
FND    5355      I YES - HALT SYSTEM I
FND    5356      I Clr I
FND    5357      MOV AL,0FH
FND    5358      OUT CMD_PORT,AL
FND    5359      OUT CMD_PORT,AL
FND    5360      MOV AL,1000001B
FND    5361      I DISABLE KB I
FND    5362      OUT PORT_R,AL
FND    5363      MOV AL,0
FND    5364      OUT PORT_R,AL
FND    5365      MOV AL,MFR_ERR_FLAG
FND    5366      OUT PORT_R,AL
FND    5367      HALT
FND    5368      012:
FND    5369      POP DS
FND    5370      POP AX
FND    5371      I WRITE_MSG I
FND    5372      E_MSG ENDP
FND    5373      P.MSG PROC NEAR
FND    5374      I PUT CHAR IN AL I
FND    5375      I POINT TO NEXT CHAR I
FND    5376      I SAVE PRINT CHAR I
FND    5377      CALL MFR_MSG
FND    5378      CALL MFR_MSG
FND    5379      POP AX
FND    5380      CPL AL,10
FND    5381      I RECOVER PRINT CHAR I
FND    5382      I HAS IT LINE FEEDS I
FND    5383      I NO - KEEP PRINTING STRING I
FND    5384      E_MSG ENDP

5385      I ---------- INITIAL RELIABILITY TEST ---------- I
5386      I SUBROUTINES : I
5387      I ----------
5388      I ---------- ASSUME CS:CODE,DS:DATA I

Appendix A

System BIOS A-75
5391 | SUBROUTINES FOR POWER ON DIAGNOSTICS |
5392 | THIS PROCEDURE WILL ISSUE ONE LONG TONE (3 SECONDS) AND ONE OR |
5393 | MORE SHORT TONES (1 SECONDS) TO INDICATE A FAILURE ON THE PLANAER |
5394 | BOARD, A BAD RAM MODULE, OR A PROBLEM WITH THE CRU. |
5395 | ENTRY PARAMETERS: |
5396 | DH = NUMBER OF LONG TONES TO BEEP |
5397 | DL = NUMBER OF SHORT TONES TO BEEP |
5398 | ERR_BEEP PROC NEAR |
5399 | CALL ERR_BEEP |
5400 | CALL ERR_BEEP |
5401 | CALL ERR_BEEP |
5402 | CALL ERR_BEEP |
5403 | CALL ERR_BEEP |
5404 | CALL ERR_BEEP |
5405 | CALL ERR_BEEP |
5406 | CALL ERR_BEEP |
5407 | CALL ERR_BEEP |
5408 | CALL ERR_BEEP |
5409 | CALL ERR_BEEP |
5410 | CALL ERR_BEEP |
5411 | CALL ERR_BEEP |
5412 | CALL ERR_BEEP |
5413 | CALL ERR_BEEP |
5414 | CALL ERR_BEEP |
5415 | CALL ERR_BEEP |
5416 | CALL ERR_BEEP |
5417 | CALL ERR_BEEP |
5418 | CALL ERR_BEEP |
5419 | CALL ERR_BEEP |
5420 | CALL ERR_BEEP |
5421 | CALL ERR_BEEP |
5422 | CALL ERR_BEEP |
5423 | CALL ERR_BEEP |
5424 | CALL ERR_BEEP |
5425 | CALL ERR_BEEP |
5426 | CALL ERR_BEEP |
5427 | CALL ERR_BEEP |
5428 | CALL ERR_BEEP |
5429 | CALL ERR_BEEP |
5430 | CALL ERR_BEEP |
5431 | CALL ERR_BEEP |
5432 | CALL ERR_BEEP |
5433 | CALL ERR_BEEP |
5434 | CALL ERR_BEEP |
5435 | CALL ERR_BEEP |
5436 | CALL ERR_BEEP |
5437 | CALL ERR_BEEP |
5438 | CALL ERR_BEEP |
5439 | CALL ERR_BEEP |
5440 | CALL ERR_BEEP |
5441 | CALL ERR_BEEP |
5442 | CALL ERR_BEEP |
5443 | CALL ERR_BEEP |
5444 | CALL ERR_BEEP |
5445 | CALL ERR_BEEP |
5446 | CALL ERR_BEEP |
5447 | CALL ERR_BEEP |
5448 | CALL ERR_BEEP |
5449 | CALL ERR_BEEP |
5450 | CALL ERR_BEEP |
5451 | CALL ERR_BEEP |
5452 | CALL ERR_BEEP |
5453 | CALL ERR_BEEP |
5454 | CALL ERR_BEEP |
5455 | CALL ERR_BEEP |
5456 | CALL ERR_BEEP |
5457 | CALL ERR_BEEP |
5458 | CALL ERR_BEEP |
5459 | CALL ERR_BEEP |
5460 | CALL ERR_BEEP |
5461 | CALL ERR_BEEP |
5462 | CALL ERR_BEEP |
5463 | CALL ERR_BEEP |
5464 | CALL ERR_BEEP |
5465 | CALL ERR_BEEP |
5466 | CALL ERR_BEEP |
5467 | CALL ERR_BEEP |
5468 | CALL ERR_BEEP |
5469 | CALL ERR_BEEP |
5470 | CALL ERR_BEEP |
5471 | CALL ERR_BEEP |
5472 | CALL ERR_BEEP |
5473 | CALL ERR_BEEP |
5474 | CALL ERR_BEEP |
5475 | CALL ERR_BEEP |
5476 | CALL ERR_BEEP |
5477 | CALL ERR_BEEP |
5478 | CALL ERR_BEEP |
5479 | CALL ERR_BEEP |
5480 | CALL ERR_BEEP |
5481 | CALL ERR_BEEP |
5482 | CALL ERR_BEEP |
5483 | CALL ERR_BEEP |
5484 | CALL ERR_BEEP |
5485 | CALL ERR_BEEP |
5486 | CALL ERR_BEEP |
5487 | CALL ERR_BEEP |
5488 | CALL ERR_BEEP |
5489 | CALL ERR_BEEP |
5490 | CALL ERR_BEEP |
5491 | CALL ERR_BEEP |
5492 | CALL ERR_BEEP |
5493 | CALL ERR_BEEP |
5494 | CALL ERR_BEEP |
5495 | CALL ERR_BEEP |
5496 | CALL ERR_BEEP |
5497 | CALL ERR_BEEP |
5498 | CALL ERR_BEEP |
5499 | CALL ERR_BEEP |
5500 | CALL ERR_BEEP |
5501 | CALL ERR_BEEP |
5502 | CALL ERR_BEEP |
5503 | CALL ERR_BEEP |
5504 | CALL ERR_BEEP |
5505 | CALL ERR_BEEP |
5506 | CALL ERR_BEEP |
5507 | CALL ERR_BEEP |
5508 | CALL ERR_BEEP |
5509 | CALL ERR_BEEP |
5510 | CALL ERR_BEE

ORG 0FF3H

VECTOR_TABLE LABEL WORD  VECTOR TABLE FOR MOVE TO INTERRUPTS

FE5  65H  5467  MOV  OFFSET TIMER_INT  INTERRUPT 6
FE6  66H  5468  MOV  OFFSET KB_INT  INTERRUPT 9
FE7  67H  5469  MOV  OFFSET D11  INTERRUPT A
FE8  68H  546A  MOV  OFFSET D12  INTERRUPT B
FE9  69H  546B  MOV  OFFSET D13  INTERRUPT C
FEA  6AH  546C  MOV  OFFSET DISK_INT  INTERRUPT E
FEB  6BH  546D  MOV  OFFSET D14  INTERRUPT F
FEC  6CH  546E  MOV  OFFSET VIDEO_ID  INTERRUPT 10H
FED  6DH  546F  MOV  OFFSET EQUIPMENT  INTERRUPT 11H
FEF  6EH  5470  MOV  OFFSET MEMORY_SIZE_DET  INTERRUPT 12H
FF0  6FH  5471  MOV  OFFSET DISKETTE_ID  INTERRUPT 13H
FF1  70H  5472  MOV  OFFSET PP32K_ID  INTERRUPT 14H
FF2  71H  5473  MOV  OFFSET CASSETTE_ID  INTERRUPT 15H(FORMER CASSETTE IO)
FF3  72H  5474  MOV  OFFSET KEYBOARD_ID  INTERRUPT 16H
FF4  73H  5475  MOV  OFFSET PRINTER_ID  INTERRUPT 17H
FF5  74H  5476  MOV  OFFSET BOOTSTRAP  INTERRUPT 18H
FF6  75H  5477  MOV  TIME_OF_DAY  INTERRUPT 1AH -- TIME OF DAY
FF7  76H  5478  MOV  DUMMY_RETURN  INTERRUPT 1BH -- KEYBOARD BREAK ADDR
FF8  77H  5479  MOV  DUMMY_RETURN  INTERRUPT 1CH -- TIMER BREAK ADDR
FF9  78H  547A  MOV  VIDEO_PANMS  INTERRUPT 1DH -- VIDEO PARAMETERS
FFA  79H  547B  MOV  DUMMY_DISK  INTERRUPT 1EH -- DISK PANS
FFB  7AH  547C  MOV  0  INTERRUPT 1FH -- POINTER TO VIDEO EXT

A-80  System BIOS
LOC OBJ

LINE SOURCE

5775 | 1. TEMPORARY INTERRUPT SERVICE ROUTINE:
5776 | 1. THIS ROUTINE IS ALSO LEFT IN PLACE AFTER THE
5777 | POWER ON DIAGNOSTICS TO SERVICE UNUASED
5778 | INTERRUPT VECTORS. LOCATION 'INTR_FLAG' WILL
5779 | CONTAIN EITHER: 1. LEVEL OF HARDWARE INT. THAT
5780 | CAUSED CODE TO BE EXEC.
5781 | 2. 'OFF' FOR NON-HARDWARE INTERRUPTS THAT WAS
5782 | EXECUTED ACCIDENTLY:
5783 |
5784 | FF33  do PROC NEAR
5785 | FF33  IX
5786 | FF26  52
5787 | FF25  50
5788 | FF24  E030FB
5789 | FF24  B0BD
5790 | FF22  E620
5791 | FF22  90
5792 | FF22  E420
5793 | FF30  8A00
5794 | FF32  0A24
5795 | FF34  7504
5796 | FF36  B4FF
5797 | FF38  E0A
5798 | FF34  5000
5799 | FF3A  E421
5800 | FF3C  0A24
5801 | FF3E  E621
5802 | FF40  B200
5803 | FF42  E620
5804 | FF44  80166000
5805 | FF44  50
5806 | FF49  5A
5807 | FF4A  1F
5808 | FF4B  CF
5809 | FF4D  41
5810 | FF53  ORG OFFSH
5811 | FF53  CF
5812 | FF54  ORG OFFSH
5813 | FF54
5814 | FF54
5815 | FF54
5816 | INT 5  
5817 | --- ORG OFFSH
5818 | --- ORG OFFSH
5819 | --- ORG OFFSH
5820 |
5821 | --- ORG OFFSH
5822 | --- ORG OFFSH
5823 | --- ORG OFFSH
5824 | --- ORG OFFSH
5825 | --- ORG OFFSH
5826 | --- ORG OFFSH
5827 | --- ORG OFFSH
5828 | --- ORG OFFSH
5829 | --- ORG OFFSH
5830 | --- ORG OFFSH
5831 | --- ORG OFFSH
5832 | --- ORG OFFSH
5833 | --- ORG OFFSH
5834 | --- ORG OFFSH
5835 | --- ORG OFFSH
5836 | --- ORG OFFSH
5837 | --- ORG OFFSH
5838 | --- ORG OFFSH
5839 | --- ORG OFFSH
5840 | --- ORG OFFSH
5841 | --- ORG OFFSH
5842 | --- ORG OFFSH
5843 | --- ORG OFFSH
5844 | --- ORG OFFSH
5845 | --- ORG OFFSH
5846 | --- ORG OFFSH
5847 | --- ORG OFFSH
5848 | --- ORG OFFSH
5849 | --- ORG OFFSH
5850 | --- ORG OFFSH
5851 | --- ORG OFFSH
5852 | --- ORG OFFSH
5853 | --- ORG OFFSH
5854 | --- ORG OFFSH
5855 | --- ORG OFFSH
5856 | --- ORG OFFSH
5857 | --- ORG OFFSH
5858 | --- ORG OFFSH
5859 | --- ORG OFFSH
5860 | --- ORG OFFSH
5861 | --- ORG OFFSH
5862 | --- ORG OFFSH
5863 | --- ORG OFFSH
5864 | --- ORG OFFSH
5865 | --- ORG OFFSH
5866 | --- ORG OFFSH
5867 | --- ORG OFFSH
5868 | --- ORG OFFSH
5869 | --- ORG OFFSH
5870 | --- ORG OFFSH
5871 | --- ORG OFFSH
5872 | --- ORG OFFSH
5873 | --- ORG OFFSH
5874 | --- ORG OFFSH
5875 | --- ORG OFFSH
5876 | --- ORG OFFSH
5877 | --- ORG OFFSH
5878 | --- ORG OFFSH
5879 | --- ORG OFFSH
5880 | --- ORG OFFSH

System BIOS A-81
A-82 System BIOS
<table>
<thead>
<tr>
<th>LOC OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFD1 C017</td>
<td>5928</td>
<td>INT 17H</td>
</tr>
<tr>
<td>FFD1 32EH</td>
<td>5929</td>
<td>XOR AH, AH</td>
</tr>
<tr>
<td>FFD0 000D</td>
<td>5930</td>
<td>MOV AL, 1B8</td>
</tr>
<tr>
<td>FFD7 CD17</td>
<td>5931</td>
<td>INT 17H</td>
</tr>
<tr>
<td>FFD9 C3</td>
<td>5932</td>
<td>RET</td>
</tr>
<tr>
<td></td>
<td>5933</td>
<td>CR LF ENDP</td>
</tr>
<tr>
<td></td>
<td>5934</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5935</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5936</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5937</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5938</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5939</td>
<td></td>
</tr>
<tr>
<td>FFD4</td>
<td>5940</td>
<td>MOV AL, DM</td>
</tr>
<tr>
<td>FFD4 BAC6</td>
<td>5941</td>
<td>CALL XMC_BYTE</td>
</tr>
<tr>
<td>FFD4 BAC2</td>
<td>5942</td>
<td>MOV AL, DL</td>
</tr>
<tr>
<td>FFEE EA7FF9</td>
<td>5943</td>
<td>CALL XMC_BYTE</td>
</tr>
<tr>
<td>FFEE D030</td>
<td>5944</td>
<td>MOV AL, '0'</td>
</tr>
<tr>
<td>FFEE E93F9F</td>
<td>5945</td>
<td>CALL PRT_HEX</td>
</tr>
<tr>
<td>FFEE B020</td>
<td>5946</td>
<td>MOV AL, '*'</td>
</tr>
<tr>
<td>FFEE EA6EF9</td>
<td>5947</td>
<td>CALL PRT_HEX</td>
</tr>
<tr>
<td>FFEE C3</td>
<td>5948</td>
<td>RET</td>
</tr>
<tr>
<td></td>
<td>5949</td>
<td>PRT_SEG ENDP</td>
</tr>
<tr>
<td></td>
<td>5950</td>
<td>CODE ENDS</td>
</tr>
<tr>
<td></td>
<td>5951</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5952</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5953</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5954</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5955</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5956</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5957</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5958</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5959</td>
<td></td>
</tr>
<tr>
<td>0000 E95B0003F0</td>
<td>5960</td>
<td>JMP RESET</td>
</tr>
<tr>
<td>0005 3131F3036538</td>
<td>5961</td>
<td>DB '11/08/82'</td>
</tr>
<tr>
<td></td>
<td>5962</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>5963</td>
<td>VECTOR ENDS</td>
</tr>
<tr>
<td></td>
<td>5964</td>
<td>END</td>
</tr>
</tbody>
</table>

Appendix A

System BIOS  A-83
$TITLE(FIXED DISK BIOS FOR IBM DISK CONTROLLER)

-- INT 13 --

* FIXED DISK I/O INTERFACE

* THIS INTERFACE PROVIDES ACCESS TO 5 1/4" FIXED DISKS

* THROUGH THE IBM FIXED DISK CONTROLLER.

The BIOS routines are meant to be accessed through:

- SOFTWARE INTERRUPTS ONLY.
- Any addresses present in

The listings are included only for completeness,

Not for Reference.

Applications which reference

Absolute addresses within the code segment

Violate the structure and design of BIOS.

---

INPUT (AH = HEX VALUE)

(AH)=00 RESET DISK (DL = 08H, DH = / DISKETTE)

(AH)=01 READ THE STATUS OF THE LAST DISK OPERATION INTO (AL)

NOTE: DL < 80H = DISKETTE

DL > 80H = DISK

(AH)=02 READ THE DESIRED SECTORS INTO MEMORY

(AH)=03 WRITE THE DESIRED SECTORS FROM MEMORY

(AH)=04 VERIFY THE DESIRED SECTORS

(AH)=05 FORMAT THE DESIRED TRACK

(AH)=06 FORMAT THE DESIRED TRACK AND SET BAD SECTOR FLAGS

(AH)=07 FORMAT THE DRIVE STARTING AT THE DESIRED TRACK

(AH)=08 RETURN THE CURRENT DRIVE PARAMETERS

(AH)=09 INITIALIZE DRIVE PAIR CHARACTERISTICS

INTERUPT 41 POINTS TO DATA BLOCK

(AH)=0A READ LONG

(AH)=0B WRITE LONG

NOTE: READ AND WRITE LONG ENCOMPASS S12 + 4 BYTES ECC

(AH)=0C SEEK

(AH)=0D ALTERNATE DISK RESET (SEE DL)

(AH)=0E READ SECTOR BUFFER

(AH)=0F WRITE SECTOR BUFFER,

(RECOMMENDED PRACTICE BEFORE FORMATTING)

(AH)=10 TEST DRIVE READY

(AH)=11 RECALIBRATE

(AH)=12 CONTROLLER RAM DIAGNOSTIC

(AH)=13 DRIVE DIAGNOSTIC

(AH)=14 CONTROLLER INTERNAL DIAGNOSTIC

---

REGISTERS USED FOR FIXED DISK OPERATIONS

- (DL) = DRIVE NUMBER (00H-07H FOR DISK, VALUE CHECKED)
- (DH) = HEAD NUMBER (0-7 ALLOWED, NOT VALUE CHECKED)
- (CH) = CYLINDER NUMBER (0-1023), NOT VALUE CHECKED [SEE CL]
- (CL) = SECTOR NUMBER (1-17), NOT VALUE CHECKED

NOTE: HIGH 2 BITS OF CYLINDER NUMBER ARE PLACED

IN THE HIGH 2 BITS OF THE CL REGISTER

(10 BITS TOTAL)

- (AL) = NUMBER OF SECTORS (MAXIMUM POSSIBLE RANGE 1-60H)

- FOR READ/WRITE LONG 1-7FH

- INTERLEAVE VALUE FOR FORMAT 1-160

- (ES:BX) = ADDRESS OF BUFFER FOR READS AND WRITES,

- (NOT REQUIRED FOR VERIFY)

---

OUTPUT

(AH) = STATUS OF CURRENT OPERATION

STATUS BITS ARE DEFINED IN THE ENTRIES BELOW

- CY = 0 SUCCESSFUL OPERATION (AH=0 ON RETURN)

- CY = 1 FAILED OPERATION (AH HAS ERROR REASON)

NOTE: ERROR 11H INDICATES THAT THE DATA READ HAD A RECOVERABLE

ERROR WHICH WAS CORRECTED BY THE ECC ALGORITHM. THE DATA

IS PROBABLY GOOD, HOWEVER THE BIOS ROUTINE INDICATES AN

ERROR TO ALLOW THE CONTROLLING PROGRAM A CHANCE TO DECIDE

FOR ITSELF. THE ERROR MAY NOT RECUR IF THE DATA IS

---

A-84 Fixed Disk BIOS
SRE_DEFAULT EQU 0F7H               ; SENSE OPERATION FAILED
UNDEF_ERR EQU 080H                ; UNDEFINED ERROR OCCURRED
TIME_OUT EQU 05H                  ; ATTACHMENT FAILED TO RESPOND
BAD_SEEK EQU 40H                  ; SEEK OPERATION FAILED
BAD_CHKLR EQU 02H                 ; CONTROLLER HAS FAILED
DATA_CORRECTED EQU 11H           ; ECC CORRECTED DATA ERROR
BAD_ECC EQU 18H                   ; BAD ECC ON DISK READ
BAD_TRACK EQU 08H                 ; BAD TRACK FLAG DETECTED
DNA_BOUNDARY EQU 09H              ; ATTEMPT TO DMA ACROSS 64K BOUNDARY
DISK_FAILED EQU 07H               ; DRIVE PARAMETER ACTIVITY FAILED
BAD_RESET EQU 09H                 ; RESET FAILED
RECORD_NOT_RD EQU 04H             ; REQUESTED SECTOR NOT FOUND
BAD_ADDR MARK EQU 02H             ; ADDRESS MARK NOT FOUND
BAD_CMD EQU 01H                   ; BAD COMMAND PASSED TO DISK I/O

----------

DUMMY SEGMENT AT 0
DUMMY SEGMENT AT 40H
DUMMY END

DATA SEGMENT AT 40H

CODE SEGMENT
A-86  Fixed Disk BIOS
Appendix A

LOC OBJ
LINE
SOURCE
0027 PA
0028 A14COO
0029 A30091
0030 A14COO
0031 A30781
0032 C704C4C05602
0033 C004E4E0
0034 B64007
0035 A33400
0044 C0E34E00
0045 C75440800601
0046 C146E600
0047 C0D4001E703
0056 B004601
005C FB
0060 28
0064 ASSUME DS:DATA
0065 MOV AX,DATA
0066 MOV DS:AX
0067 MOV DS:DATA
0068 MOV AX,DATA
0069 MOV AX,DATA
006A MOV AX,DATA
006B MOV AX,DATA
006C MOV AX,DATA
006D MOV AX,DATA
006E MOV AX,DATA
006F MOV AX,DATA
0070 MOV AX,DATA
0071 MOV AX,DATA
0072 MOV AX,DATA
0073 MOV AX,DATA
0074 MOV AX,DATA
0075 MOV AX,DATA
0076 MOV AX,DATA
0077 MOV AX,DATA
0078 MOV AX,DATA
0079 MOV AX,DATA
007A MOV AX,DATA
007B MOV AX,DATA
007C MOV AX,DATA
007D MOV AX,DATA
007E MOV AX,DATA
007F MOV AX,DATA
0080 MOV AX,DATA
0081 MOV AX,DATA
0082 MOV AX,DATA
0083 MOV AX,DATA
0084 MOV AX,DATA
0085 MOV AX,DATA
0086 MOV AX,DATA
0087 MOV AX,DATA
0088 MOV AX,DATA
0089 MOV AX,DATA
008A MOV AX,DATA
008B MOV AX,DATA
008C MOV AX,DATA
008D MOV AX,DATA
008E MOV AX,DATA
008F MOV AX,DATA
0090 MOV AX,DATA
0091 MOV AX,DATA
0092 MOV AX,DATA
0093 MOV AX,DATA
0094 MOV AX,DATA
0095 MOV AX,DATA
0096 MOV AX,DATA
0097 MOV AX,DATA
0098 MOV AX,DATA
0099 MOV AX,DATA
009A MOV AX,DATA
009B MOV AX,DATA
009C MOV AX,DATA
009D MOV AX,DATA
009E MOV AX,DATA
009F MOV AX,DATA
00A0 MOV AX,DATA
00A1 MOV AX,DATA
00A2 MOV AX,DATA
00A3 MOV AX,DATA
00A4 MOV AX,DATA
00A5 MOV AX,DATA
00A6 MOV AX,DATA
00A7 MOV AX,DATA
00A8 MOV AX,DATA
00A9 MOV AX,DATA
00AA MOV AX,DATA
00AB MOV AX,DATA
00AC MOV AX,DATA
00AD MOV AX,DATA
00AE MOV AX,DATA
00AF MOV AX,DATA
00B0 MOV AX,DATA
00B1 MOV AX,DATA
00B2 MOV AX,DATA
00B3 MOV AX,DATA
00B4 MOV AX,DATA
00B5 MOV AX,DATA
00B6 MOV AX,DATA
00B7 MOV AX,DATA
00B8 MOV AX,DATA
00B9 MOV AX,DATA
00BA MOV AX,DATA
00BB MOV AX,DATA
00BC MOV AX,DATA
00BD MOV AX,DATA
00BE MOV AX,DATA
00BF MOV AX,DATA
00C0 MOV AX,DATA
00C1 MOV AX,DATA
00C2 MOV AX,DATA
00C3 MOV AX,DATA
00C4 MOV AX,DATA
00C5 MOV AX,DATA
00C6 MOV AX,DATA
00C7 MOV AX,DATA
00C8 MOV AX,DATA
00C9 MOV AX,DATA
00CA MOV AX,DATA
00CB MOV AX,DATA
00CC MOV AX,DATA
00CD MOV AX,DATA
00CE MOV AX,DATA
00CF MOV AX,DATA
00D0 MOV AX,DATA
00D1 MOV AX,DATA
00D2 MOV AX,DATA
00D3 MOV AX,DATA
00D4 MOV AX,DATA
00D5 MOV AX,DATA
00D6 MOV AX,DATA
00D7 MOV AX,DATA
00D8 MOV AX,DATA
00D9 MOV AX,DATA
00DA MOV AX,DATA
00DB MOV AX,DATA
00DC MOV AX,DATA
00DD MOV AX,DATA
00DE MOV AX,DATA
00DF MOV AX,DATA
00E0 MOV AX,DATA
00E1 MOV AX,DATA
00E2 MOV AX,DATA
00E3 MOV AX,DATA
00E4 MOV AX,DATA
00E5 MOV AX,DATA
00E6 MOV AX,DATA
00E7 MOV AX,DATA
00E8 MOV AX,DATA
00E9 MOV AX,DATA
00EA MOV AX,DATA
00EB MOV AX,DATA
00EC MOV AX,DATA
00ED MOV AX,DATA
00EE MOV AX,DATA
00EF MOV AX,DATA
00F0 MOV AX,DATA
00F1 MOV AX,DATA
00F2 MOV AX,DATA
00F3 MOV AX,DATA
00F4 MOV AX,DATA
00F5 MOV AX,DATA
00F6 MOV AX,DATA
00F7 MOV AX,DATA
00F8 MOV AX,DATA
00F9 MOV AX,DATA
00FA MOV AX,DATA
00FB MOV AX,DATA
00FC MOV AX,DATA
00FD MOV AX,DATA
00FE MOV AX,DATA
00FF MOV AX,DATA

Fixed Disk BIOS  A-87
LOC OBJ | LINE | SOURCE
---|---|---
0055 06CH | 305 | MOV EB,AX | ; SET SEGMENT
0057 260B | 306 | SUB DX,DX | ;
0059 D8D00F | 307 | MOV AX,0000H | ; WRITE Sector BUFFER
0061 CD13 | 308 | INT 13H | ;
0063 7252 | 309 | JC ERROR_EX | ;
130 | 310 | INC NP_NUM | ; DRIVE ZERO RESPONDED
0074 BA192 | 311 | MOV DX,215H | ; EXPANSION BOX
0077 B000 | 312 | MOV AL,0 | ;
007F EE | 313 | OUT DX,AL | ; TURN BOX OFF
0082 BA193 | 314 | MOV DX,321H | ; TEST IF CONTROLLER
0082 EC | 315 | IN AL,DX | ; ... IS IN THE SYSTEM UNIT
0082 E40F | 316 | AND AL,0FH | ;
0083 3C0F | 317 | CMP AL,0FH | ;
0083 7496 | 318 | JE BOX_ON | ;
0083 CB40 | 319 | MOV TIMER_LOW,420D | ; CONTROLLER IS IN SYSTEM UNIT
0100 420D | 320 | BOX_ON: | ;
0103 A192 | 321 | MOV DX,215H | ; EXPANSION BOX
0106 B0FF | 322 | MOV AL,0FH | ;
010F EE | 323 | OUT DX,AL | ; TURN BOX ON
130 | 324 | INC NP_NUM | ;
0118 D910 | 325 | MOV CH,1 | ; ATTEMPT NEXT DRIVES
011B DA00 | 326 | MOV DX,001H | ;
011E 28C | 327 | SUB AX,AX | ; RESET
011E CD13 | 328 | INT 13H | ;
011A 7240 | 329 | JC LOOP_DONE | ; RECAL
011C B001 | 330 | MOV AX,01100H | ;
011F CD13 | 331 | INT 13H | ;
0121 720B | 332 | JNC PS | ;
0123 A1C0 | 333 | MOV AX,TIMER_LOW | ;
0126 3C8E01 | 334 | CMP AX,144D | ; 12 SECONDS
0129 7EED | 335 | JB PS | ;
012B E8F90 | 336 | JMP LOOP_DONE | ;
012E 00009 | 337 | MOV AX,0900H | ; INITIALIZE CHARACTERISTICS
0131 CD13 | 338 | INT 13H | ;
0133 7227 | 339 | JC LOOP_DONE | ;
0135 FE07500 | 340 | INC HF_NUM | ; TALLY ANOTHER DRIVE
0139 81FA100 | 341 | CMP DX,100H + S_MAX_FILE - 1 | ;
013D 7310 | 342 | JAE LOOP_DONE | ;
013F 40 | 343 | INC DX | ;
0140 8004 | 344 | JMP PS | ;
0144 39 | 345 | STC | ;
359 | 346 | LOOP DONE: | ;
0142 ERROR_EX: | 350 | MOV BP,0FH | ; LOOP ERROR FLAG
0145 0F00 | 351 | MOV BP,0FH | ;
0148 2BC0 | 352 | SUB AX,AX | ;
014E 0F0 | 353 | MOV DI,AX | ;
014F B06009 | 354 | MOV CH,FI7L | ; MESSAGE CHARACTER COUNT
0150 B700 | 355 | MOV BX,0 | ; PAGE ZERO
014F OUT_CH: | 356 | MOV AL,CS:FI7FSI | ; GET BYTE
0154 B40E | 357 | MOV AM,140 | ; VIDEO OUT
0156 CD10 | 358 | INT 10H | ; DISPLAY CHARACTER
0158 44 | 359 | INC SI | ; NEXT CHAR
0159 E9F4 | 360 | LOOP OUT_CH | ; DO MORE
015B 89 | 361 | STC | ;
015C LOOP_DONE: | 362 | MOVDX,0 | ;
015F FA | 363 | CLI | ;
0160 E621 | 364 | IN AL,02IH | ; BE SURE TIMER IS DISABLED
0165 CD0 | 365 | OR AL,01H | ;
0166 E621 | 366 | OUT 02H,AL | ;
0167 FB | 367 | STI | ;
0168 E859D | 368 | CALL DISBL | ;
016C CB | 369 | RET | ;
016D 3173031 | 370 | FIT DB '1701','00H','00H' | ;
016C 80 | 371 | MOV DX,DSBL | ;
0006 | 372 | F17L EQU S-F17 | ;
0168 0A | 373 | MOV AX,00H | ;
0168 E7 | 374 | MOV DI,00H | ;
0168 E7 | 375 | MOV AX,00H | ;
0168 E7 | 376 | MOV DX,00H | ;
0168 E7 | 377 | MOV AX,00H | ;
0168 E7 | 378 | MOV AX,00H | ;
0168 E7 | 379 | MOV AX,00H | ;

A-88 Fixed Disk BIOS
LOC OBJ
LINE SOURCE
0179 FB 380 CLC
0179 B90001 381 MVI CH,0100H
0179 382 L6:
0179 E5706 383 MVI CH,0100H
0178 384 L6:
0178 E5000E 385 CALL PORT_1
0178 386 OUT DX,AL
0178 EC 387 CALL PORT_1
0178 E50006 388 IN AL,DX
0178 2602 389 AND AL,E
0178 38A 390 JZ R5
0178 7403 391 LOOP L6
0182 F9 392 R5:
0185 393 POP DX
0185 394 POP CX
0185 C3 395 RET
396
397 MD_RESET_1 ENDP
398
399 DISK_SETUP ENDP
399
400-inter INT 19-------------------------
401  ; INTERRUPT 19 BOOT STRAP LOADER
402  ;
403  ; - THE FIXED DISK BIOS REPLACES THE INTERRUPT 19
404  ; BOOT STRAP VECTOR WITH A POINTER TO THIS ROUTINE
405  ; - RESET THE DEFAULT DISK AND DISKETTE PARAMETER VECTORS
406  ; - THE DISK BLOCK TO BE READ IN WILL BE ATTEMPTED FROM
407  ; CYLINDER 0 SECTOR 1 OF THE DEVICE.
408  ; - THE BOOTSTRAP SEQUENCE IS:
409  ; > ATTEMPT TO LOAD FROM THE DISKETTE INTO THE BOOT
410  ; LOCATION (0000:7C00) AND TRANSFER CONTROL THERE
411  ; > IF THE DISKETTE FAILS THE FIXED DISK IS TRIED FOR A
412  ; VALID BOOT BLOCK. A VALID BOOT BLOCK ON THE
413  ; FIXED DISK CONSISTS OF THE BYTES 03H 03H AS THE
414  ; LAST TWO BYTES OF THE BLOCK
415  ; > IF THE ABOVE FAILS CONTROL IS PASSED TO RESIDENT BASIC
416  ;
417  ;----------------------------------------
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
448
449
450
451
452
453
454
455
456

BOOTSTRAP:
ASSUME DS:dummy, ES: dummy
MVI AX,AX
MVI DS,AX
I ESTABLISH SEGMENT

RESET PARAMETER VECTORS

CLI
MVI WORD PTR NF_TBL_VEC, OFFSET FB_TBL
MVI WORD PTR NF_TBL_VEC+2, CS
MVI WORD PTR DISKETTE_PARM, OFFSET DISKETTE_TBL
MVI WORD PTR DISKETTE_PARM+4, CS
STI

ATTEMPT BOOTSTRAP FROM DISKETTE

MVI CX,3
I SET RETRY COUNT
I IPL_SYSTEM
I SAVE RETRY COUNT
I DRIVE ZERO
I RESET THE DISKETTE
I IF ERROR, TRY AGAIN
I READ IN THE SINGLE SECTOR

ADD DX,DX
I ESTABLISH SEGMENT

MVI CX,1
I SECTOR 1, TRACK 8
I FILE ID CALL
I RECOVER RETRY COUNT
I CP SET BY UNSUCCESSFUL READ
I IF TIME OUT, NO RETRY
I TRY FIXED DISK
I DO IT FOR RETRY TIMES
I UNABLE TO IPL FROM THE DISKETTE
I IPL WAS SUCCESSFUL
0100 1A007C0000 457  JMP  BOOT_LOCM
0100 1A007C0000 458  
0100 1A007C0000 459  I------- ATTEMPT BOOTSTRAP FROM FIXED DISK
0100 1A007C0000 460  
0100 1A007C0000 461  H5:
0100 1A007C0000 462  SUB  AX,AX
0100 1A007C0000 463  SUB  DX,DX
0100 1A007C0000 464  INT  13H
0100 1A007C0000 465  MOV  CX,3
0100 1A007C0000 466  H6:
0100 1A007C0000 467  PUSH  CX
0100 1A007C0000 468  MOV  DX,088H
0100 1A007C0000 469  MOV  AX,0AH
0100 1A007C0000 470  INT  13H
0100 1A007C0000 471  JC  H7
0100 1A007C0000 472  MOV  AX,0200H
0100 1A007C0000 473  SUB  DX,DX
0100 1A007C0000 474  MOV  ES,DX
0100 1A007C0000 475  MOV  BX,OFFSET BOOT_LOCM
0100 1A007C0000 476  MOV  DX,088H
0100 1A007C0000 477  MOV  CX,1
0100 1A007C0000 478  INT  13H
0100 1A007C0000 479  H7:
0100 1A007C0000 480  MOV  CX
0100 1A007C0000 481  JC  H6
0100 1A007C0000 482  MOV  AX,0400H
0100 1A007C0000 483  JMP  AX,0A45H
0100 1A007C0000 484  H8:
0100 1A007C0000 485  LOOP  H6
0100 1A007C0000 486  
0100 1A007C0000 487  I------- UNABLE TO IPL FROM THE DISKETTE ON FIXED DISK
0100 1A007C0000 488  
0100 1A007C0000 489  INT  16H
0100 1A007C0000 490  RESIDENT BASIC
0100 1A007C0000 491  DISKETTE_TBL:
0100 1A007C0000 492  
0100 1A007C0000 493  DB  11001111B
0100 1A007C0000 494  DB  2
0100 1A007C0000 495  DB  256H
0100 1A007C0000 496  DB  2
0100 1A007C0000 497  DB  8
0100 1A007C0000 498  DB  02AH
0100 1A007C0000 499  DB  0FH
0100 1A007C0000 500  DB  05H
0100 1A007C0000 501  DB  06H
0100 1A007C0000 502  DB  25
0100 1A007C0000 503  DB  4
0100 1A007C0000 504  
0100 1A007C0000 505  I------- MAKE SURE THAT ALL HOUSEKEEPING IS DONE BEFORE EXIT
0100 1A007C0000 506  
0100 1A007C0000 507  OCSL  PROC  HEAR
0100 1A007C0000 508  
0100 1A007C0000 509  DB  0A0H
0100 1A007C0000 510  MOV  AX,DATA
0100 1A007C0000 511  MOV  AX,DATA
0100 1A007C0000 512  
0100 1A007C0000 513  MOV  AH,PORT_OFF
0100 1A007C0000 514  PUSK  AX
0100 1A007C0000 515  
0100 1A007C0000 516  MOV  PORT_OFF,OH
0100 1A007C0000 517  CALL  PORT_3
0100 1A007C0000 518  DB  110H
0100 1A007C0000 519  OUT  DX,AL
0100 1A007C0000 520  MOV  PORT_OFF,8H
0100 1A007C0000 521  CALL  PORT_3
0100 1A007C0000 522  DB  0AH
0100 1A007C0000 523  OUT  DX,AL
0100 1A007C0000 524  MOV  PORT_OFF,0H
0100 1A007C0000 525  CALL  PORT_3
0100 1A007C0000 526  DB  01H
0100 1A007C0000 527  OUT  DX,AL
0100 1A007C0000 528  MOV  PORT_OFF,0CH
0100 1A007C0000 529  CALL  PORT_3
0100 1A007C0000 530  DB  0AH
0100 1A007C0000 531  OUT  DX,AL
0100 1A007C0000 532  MOV  AL,07H
0100 1A007C0000 533  OUT  DMA+10H,AL

A-90  Fixed Disk BIOS
LOC OBJ  | LINE | SOURCE
------- | ---- | ----
0247 FA | 534  | CLI  | DISABLE INTERRUPTS
0248 4941 | 535  | IN   | AL, DIH
0249 3C20 | 536  | OR   | AL, DIH
024C E621 | 537  | OUT  | 02H, AL
0250 7B9F | 538  | STI  | ENABLE INTERRUPTS
0251 5B80 | 539  | POP  | AX
0252 8567700 | 540  | MOV  | PORT_OFFSET, AH
0254 1F | 541  | POP  | DS
0255 C3 | 542  | RET  | RESTORE SEGMENT
0256 | 543  | DBL  | ENDP
       | 544  |      |--------------------------
0256 | 545  | DISK_IO PROC FAR
0258 80FA00 | 546  | A8H  | DISK_IO:NOTHING, ES:NOTHING
0259 7305 | 547  | JAE  | MISMATCH_DISK
025D C400 | 548  | INT  | 40H
025E | 549  | RET  | 2
          | 550  |      | BACK TO CALLER
0260 | 551  | HARD_DISK: | TEST FOR FIXED DISK DRIVE
0261 6A94 | 552  | CMP  | DL, 40H
0263 7699 | 553  | JNZ  | YES, HANDLE HERE
0265 CD9A | 554  | JMP  | DISKETTE_HANDLER
0267 | 555  | RET  | 2
0269 | 556  | CMP  | AH, 0B
026A 4941 | 557  | OR   | AH, AH
026C 5B80 | 558  | INT  | 40H
026D 8FFA | 559  | SUB  | AH, AH
026F 77EF | 560  | JMP  | RET_1
0271 7305 | 561  | CMP  | DL, (40H + D_MAX_FILE - 1)
0273 91A01 | 562  | JA   | RET_1
0275 | 563  | CMP  | AH, 0B
0276 | 564  | JMP  | GET_PARAM
0276 | 565  | JNZ  | A2
0277 | 566  | PUSH  | DX
0278 | 567  | PUSH  | CX
0279 | 568  | PUSH  | DX
027A | 569  | PUSH  | ES
027B | 570  | PUSH  | SI
027C | 571  | PUSH  | DI
027D | 572  | CALL  | DISK_IO_CONT
027E | 573  | PUSH  | AX
027F | 574  | CALL  | DBL
0280 69400 | 575  | MOV  | AX, DATA
0282 6600 | 576  | MOV  | DS, AX
0284 5B80 | 577  | MOV  | AH, DI
0285 | 578  | MOV  | AH, DISK_STATUS
0286 09C001 | 579  | CMP  | AH, 1
0287 7B9F | 580  | CYC  | SET THE CARRY FLAG TO INDICATE
0289 | 581  | POP  | DI
028B | 582  | POP  | SI
028C | 583  | POP  | ES
028D | 584  | POP  | DS
028E | 585  | POP  | DX
028F | 586  | POP  | CX
0290 | 587  | POP  | BX
0291 | 588  | RET  | 2
0292 | 589  | DISK_IO ENDP
0293 | 590  |      |--------------------------
029C | 591  | M1 LABEL WORD | FUNCTION TRANSFER TABLE
029D 3A03 | 592  | DM  | DISK_RESET
029E 4003 | 593  | DM  | RETURN_STATUS
02A0 5603 | 594  | DM  | DISK_READ
02A2 6003 | 595  | DM  | DISK_WRITE
02A4 6A03 | 596  | DM  | DISK_VERF
02A6 7203 | 597  | DM  | FNT_TRK
02A8 7903 | 598  | DM  | FNT_BAD
02AA 8003 | 599  | DM  | FNT_DRV
02AC 3003 | 600  | DM  | BAD_COMMAND
02AE 2704 | 601  | DM  | INIT_DRV
02B0 0704 | 602  | DM  | RG_LONG
02B2 0D04 | 603  | DM  | WR_LONG

Appendix A

Fixed Disk BIOS  A-91
A-92  Fixed Disk BIOS
LOC OBJ
LINE
SOURCE

033A 4E504
033B EE
033C E8F04
033F EC
0340 2402
0342 7406
0344 C046740000
0349 C3
034A EHAA0
034D 807
0350 C606740000
0355 C3
0356 0047
0358 C606420006
0360 2385
0362 C60662000A
0367 E9B001
0368 0048
036A C606420005
036F E9C001
0372 C6064200006
0377 E9C0
0379 C6064200097
037E E9D0
0380 C606420004
0385 00
0385 A24400
0388 2400
038A A24400
0390 E9A401

0388 DISK_RESET PROC NEAR
0389 CALL PORT_1
0390 ; RESET PORT
0391 OUT DX, AL
0392 ; ISSUE RESET
0393 CALL PORT_1
0394 ; CONTROLLER HARDWARE STATUS
0395 IN AL, DX
0396 ; SET STATUS
0397 AND AL, 2
0398 ; ERROR BIT
0399 JE DRI
0400 MOV DISK_STATUS,BAD_RESET
0401 DRI:
0402 JMP INIT_DRV
0403 ; SET THE DRIVE PARAMETERS
0404 DISK_RESET ENDP

0408 RETURN_STATUS PROC NEAR
0409 MOV AL,DISK_STATUS
0410 ; OBTAIN PREVIOUS STATUS
0411 MOV DISK_STATUS,D
0412 ; RESET STATUS
0413 RET
0414 RETURN_STATUS ENDP

0418 DISK_READ PROC NEAR
0419 MOV AL,DMA_READ
0420 ; MODE BYTE FOR DMA READ
0421 MOV CHD_BLOCK+8,READ_CND
0422 JMP DMA_DPH
0423 DISK_READ ENDP

0426 DISK_WRITE PROC NEAR
0427 MOV AL,DMA_WRITE
0428 ; MODE BYTE FOR DMA WRITE
0429 MOV CHD_BLOCK+8,WRITE_CND
0430 JMP DMA_DPH
0431 DISK_WRITE ENDP

0436 DISK_VERIFY PROC NEAR
0437 MOV CHD_BLOCK+8,CHK_TRK_CND
0438 JMP HDD_DPH
0439 DISK_VERIFY ENDP

0441 DISK_FORMAT PROC NEAR
0442 MOV CHD_BLOCK+8,FORMAT_CND
0443 JMP HDD_DPH
0444 DISK_FORMAT ENDP

0449 FMT_TRK PROC NEAR
0450 MOV CHD_BLOCK,FMTTRK_CND
0451 JMP SHORT FMT_CONT
0452 FMT_TRK ENDP

0457 FMT_BAD PROC NEAR
0458 MOV CHD_BLOCK,FMBAD_CND
0459 JMP SHORT FMT_CONT
0460 FMT_BAD ENDP

0465 FMT_DRV PROC NEAR
0466 MOV CHD_BLOCK,FMTDRV_CND
0467 JMP SHORT FMT_CONT
0468 FMT_DRV ENDP

0473 FMT_CONT:
0474 MOV AL,CMD_BLOCK+2
0475 MOV AL,110000000B
0476 AND AL
0477 MOV AL,CMD_BLOCK+2,AL
0478 MOV HDD_DPH

Appendix A

Fixed Disk BIOS A-93
LOC OBJ

764 1------------------------------------------
1
765 1 GET PARAMETERS (AH = 0)

766 1------------------------------------------

0300 768 GET_PWR_M
0301 769 LABEL NEAR
0302 769 GET_PWR_M
0303 769 PROC FAR
0304 770 GET_DRIVE PARAMETERS
0305 770 PUSH DS
0306 770 SAVE REGISTERS
0307 770 PUSH ES
0308 770 PUSH BX

0309 770 ASSURE DS:DUMMY
0310 770 SUB AX,AX
0311 770 ENSURE AX,AX
0312 770 LES BX,‘CF_TBL_VEC’
0313 770 ASSURE DS:DATA
0314 770 MOV AX,DATA
0315 770 MOV DS:AX
0316 770 ENSURE SEGMENT
0317 770 SUB DL,50H
0318 770 CMP DL,MAX_FILE
0319 770 JAE 6A
0320 770 CALL SETUP_A
0321 770 CALL SND_OFFS
0322 770 JC 6A
0323 770 ADD BX,AX
0324 770 MAX NUMBER OF CYLINDERS
0325 770 SUB AX,2
0326 770 ADJUST FOR O-N
0327 770 AND AX,030H
0328 770 HIGH TWO BITS OF CYL
0329 770 SHR AX,1
0330 770 SHR AX,1
0331 770 OR AL,01H
0332 770 SECTORS
0333 770 MOV CL,AL
0334 770 MOV AH,ES:BX[2I]
0335 770 DH
0336 770 O-H RANGE
0337 770 MOV DL,MAX_NUM
0338 770 DRIVE COUNT
0339 770 MOV AX,AX
0340 770 RESTORE REGISTERS
0341 770 POP DX
0342 770 POP ES
0343 770 POP DS
0344 770 RET 2
0345 770 GA:
0346 770 MOV DISK_STATUS,INIT_FAIL
0347 770 OPERATION FAILED
0348 770 MOV AH,INIT_FAIL
0349 770 SUB AL,AL
0350 770 SUB DX,DX
0351 770 SUB CX,CX
0352 770 STC
0353 770 SET ERROR FLAG
0354 770 JP MP 6G
0355 770 GET_PWR_M
0356 770 ENDP
0357 770 GET_PWR_M
0358 770 ENDP

781 1------------------------------------------
1
782 1 INITIALIZE DRIVE CHARACTERISTICS

783 1------------------------------------------

0100 820

A-94 Fixed Disk BIOS
LOC OBJ | LINE | SOURCE
041A 8201 | 919 | DW O360D
041C 9000 | 920 | DW O0000D
041E 00 | 921 | DB OSH
041F 05 | 922 | DB OSH | 1 STANDARD
0420 0C | 923 | DB OCH | 1 FORMAT DRIVE
0421 04 | 924 | DB OAH | 1 CHECK DRIVE
0422 28 | 925 | DB OZHN
0423 00000000 | 926 | DB 0.0.0.0
0427 | 927 | INIT_DRV PROC NEAR
0427 | 928 | ----- DO DRIVE ZERO
0427 C6A4D2000C | 929 | MOV CMD_BLOCK+0.INIT_DRV_CMD
042C C6A4D50000 | 930 | MOV CMD_BLOCK+1.0
0431 E01000 | 931 | CALL INIT_DRV_R
0436 720D | 932 | JC INIT_DRV_OUT
0436 | 933 | ----- DO DRIVE ONE
0436 C6A4D2000C | 934 | MOV CMD_BLOCK+0.INIT_DRV_CMD
043B C6A4D50020 | 935 | MOV CMD_BLOCK+1.01000000B
0440 E80100 | 936 | CALL INIT_DRV_R
0443 | 937 | INIT_DRV_OUT:
0443 C3 | 938 | RET
0444 | 939 | INIT_DRV ENDP
0444 | 940 | INIT_DRV_R PROC NEAR
0444 | 941 | Assume ES:CODE
0446 E81901 | 942 | SUB AX,AL
0449 73DB | 943 | CALL COMMAND | 1 ISSUE THE COMMAND
044B 7301 | 944 | JC B1
044B C3 | 945 | RET
044C | 946 | BI:
044C 1E | 947 | Push DS | 1 SAVE SEGMENT
044D 8BC9 | 948 | Assume DS:SSDATA
044F 8ED6 | 949 | SUB AX,AX
0451 C4894901 | 950 | MOV DS,AX | 1 ESTABLISH SEGMENT
0455 1F | 951 | LES AX,FP,VEC | 1 RESTORE SEGMENT
0456 E83403 | 952 | MOV DS,OFFS
0459 7257 | 953 | JC B3
045B 0320 | 954 | ADD BX,AX
0463 | 955 | ----- SEND DRIVE PARAMETERS MOST SIGNIFICANT BYTE FIRST
0463 D9100 | 956 | MOV DI.1
0466 E0F700 | 957 | CALL INIT_DRV_R
0466 724D | 958 | JC B3
0466 | 959 | MOV DI.0
0466 030700 | 960 | CALL INIT_DRV_R
0466 724D | 961 | JC B3
0466 | 962 | MOV DI.2
0466 030700 | 963 | CALL INIT_DRV_R
0467 7230 | 964 | JC B3
0467 | 965 | MOV DI.4
0467 E04700 | 966 | CALL INIT_DRV_R
0467 7235 | 967 | JC B3
0467 | 968 | MOV DI.3
0468 E03F00 | 969 | CALL INIT_DRV_R
0468 722D | 96A | JC B3
0468 | 96B | MOV DI.6
0468 E03700 | 96C | CALL INIT_DRV_R
0468 721D | 96D | JC B3
0468 | 96E | MOV DI.5
046B E05F00 | 96F | CALL INIT_DRV_R
0470 721D | 970 | JC B3
0470 E02700 | 971 | CALL INIT_DRV_R
0470 | 972 | MOV DI.7
0470 E02700 | 973 | CALL INIT_DRV_R

A-96  Fixed Disk BIOS
LOC OBJ  |  LINE  |  SOURCE

0479 T21S  |  996   |  JC  B3
0490 8F0000  |  997   | MOV D1,0 : DRIVE STEP OPTION
04A0 26A01  |  998   | MOV AL,ES:IBX + DII
04A3 A7600  |  999   | MOV CONTROL_BYTE,AL
1000         |  1000  |
0466 2BC0F  |  1001  | SUB CX,CX
0468         |  1002  |
046A E0302  |  1003  | BS:
046B E5C  |  1004  | CALL PORT_1
046B 1F  |  1005  | IN AL,DX
046C 1002  |  1006  | TEST AL,AL\_INDEX : STATUS INPUT MODE
046E 7559  |  1007  | JNZ D6
046E 82F6  |  1008  | LOOP BS
0472         |  1009  | BS:
0472 C667400007  |  1010  | MOV DISK\_STATUS,INIT\_FAIL : OPERATION FAILED
0487 F9  |  1011  | STC
048B C3  |  1012  | RET
1013         |  1013  |
0489         |  1014  | D6:
0489 88902  |  1015  | CALL PORT_0
048C EC  |  1016  | IN AL,DX
049D 2402  |  1017  | AND AL,2 : MASK ERROR BIT
049F 7551  |  1018  | JNZ B3
04C1 C3  |  1019  | RET
1020         |  1020  | ASSUME ES=NULLPTR
1021         |  1021  | INIT\_DVR\_P ENDP
1022         |  1022  |
1023         |  1023  | SEND THE BYTE OUT TO THE CONTROLLER
1024         |  1024  |
04C2         |  1025  | INIT\_DVR\_S PROC NEAR
04C2 E0C501  |  1026  | CALL EOF\_WAIT\_RES
04C5 7207  |  1027  | JC D1
04C7 8A702  |  1028  | CALL PORT_0
04CD 26A001  |  1029  | MOV AL,ES:IBX + DII
04CE 60  |  1030  | OUT DX,AL
04CE         |  1031  | D1:
04CE C3  |  1032  | RET
1033         |  1033  | INIT\_DVR\_S ENDP
1034         |  1034  |
1035         |  1035  |-------------------------------------------
1036         |  1036  | READ LONG (AH = 0AH):
1037         |  1037  |-------------------------------------------
1038         |
04CF         |  1039  | RD\_LONG PROC NEAR
04CF E81900  |  1040  | CALL CHK\_LONG
04D2 7260  |  1041  | JC G0
04D4 C6604205E  |  1042  | MOV CMD\_BLOCK+0,RD\_LONG\_CMD
04D9 8047  |  1043  | MOV AL,DMA\_READ
04DB 8E6A  |  1044  | JMP SHORT DMA\_DYN
04DE         |  1045  | RD\_LONG ENDP
1046         |
1047         |  1046  |-------------------------------------------
1048         |  1047  | WRITE LONG (AH = 0BH):
1049         |  1048  |-------------------------------------------
1050         |
04D2         |  1051  | WR\_LONG PROC NEAR
04D2 E0B000  |  1052  | CALL CHK\_LONG
04D2 725D  |  1053  | JC G0
04E2 C6604205E  |  1054  | MOV CMD\_BLOCK+0,WR\_LONG\_CMD
04E7 804D  |  1055  | MOV AL,DMA\_WRITE
04E7 8B5A  |  1056  | JMP SHORT DMA\_DYN
1057         |  1057  | WR\_LONG ENDP
1058         |
04E2         |  1059  | CHK\_LONG PROC NEAR
04EB A9600  |  1060  | MOV AL,CMD\_BLOCK+4
04EE 3000  |  1061  | CMP AL,000H
04F0 FA  |  1062  | CMC
04F1 C3  |  1063  | RET
1064         |  1064  | CHK\_LONG ENDP
1065         |
1066         |  1065  |-------------------------------------------
1067         |  1066  | SEEK (AH = 0CH):
1068         |  1067  |-------------------------------------------
1069         |
04F2         |  1070  | DISK\_SEEK PROC NEAR
04F2 C66042000B  |  1071  | MOV CMD\_BLOCK:SEEK\_CMD
04F7 8B3D  |  1072  | JMP SHORT DMA\_DYN

---
Appendix A

Fixed Disk BIOS  A-97
A-98  Fixed Disk BIOS
LOC OBJ | LINE | SOURCE
----------|-----|-----------------------
0537 | 1150 | DB: MOV DISK_STATUS,DMA_BOUNDARY
0544 | C3 | RET
0545 | 1153 | DMA_DPH:
0545 | E05791 | 1154 | CALL DMA_SETUP | I SET UP FOR DMA OPERATION
0546 | 72F5 | 1155 | JC GB
0546 | B023 | 1156 | MOV AL,03H
0546 | E81300 | 1157 | CALL COMMAND | I ISSUE THE COMMAND
054F | 720D | 1158 | JC G11
0551 | B023 | 1159 | MOV AL,03H
0553 | E99A | 1160 | OUT DMA+10,AL | I INITIALIZE THE DISK CHANNEL
0559 | 1161 | 63: MOV AL,02H
055B | 9A01 | 1162 | IN AL,02H
055F | 2A9F | 1163 | AND AL,00FH
0559 | E821 | 1164 | OUT 02H,AL
0560 | E8AA01 | 1165 | CALL WAIT_INT
0565 | 1166 | GI1: MOV AL,02H
056E | E93800 | 1167 | CALL ERROR_CHK
0561 | C3 | RET
0569 | 1169 | -----------------------------------------------
0570 | 1170 | COMMAND PROC NEAR
057A | E420B | 1171 | MOV ST.OFFSET CMD_BLOCK
057B | E1002 | 1172 | CALL PORT_2
057C | EE | 1173 | OUT DX,AL
057E | E91C02 | 1174 | CALL PORT_3 | I CONTROLLER SELECT PULSE
057F | EEC | 1175 | OUT DX,AL
0581 | 28C9 | 1176 | SUB CX,CX | I WAIT COUNT
057F | E90C02 | 1177 | CALL PORT_1
0577 | 1178 | WAIT_BUSY:
057E | EE | 1179 | IN AL,DX | I GET STATUS
0581 | 3A0F | 1180 | AND AL,0FH
0581 | 5C0D | 1181 | CMP AL,RI_BUSY OR RI_BUS OR RI_REQ
0577 | 7409 | 1182 | JE C1
0577 | E2F7 | 1183 | LOOP WAIT_BUSY
057B | C60704000 | 1184 | MOV DISK_STATUS,TIME_OUT
0580 | F9 | 1185 | STC
0581 | C3 | 1186 | RET | I ERROR RETURN
0582 | 1189 | C1:
0582 | FC | 1190 | CLD
0583 | 90600 | 1191 | MOV CX,0 | I BYTE COUNT
0584 | 1990 | 1192 | CHS:
0586 | E8801 | 1193 | CALL PORT_0
0586 | AC | 1194 | LOOP
0588 | EE | 1195 | OUT DX,AL | I GET THE NEXT COMMAND BYTE
0588 | E2F9 | 1196 | LOOP CHS | I DO MORE
058B | 1197 | LOOP:
058D | 88E01 | 1198 | CALL PORT_1 | I STATUS
058F | EC | 1199 | IN AL,DX
0591 | A001 | 1200 | TEST AL,RI_REQ
0593 | 7420 | 1201 | JE CH7
0595 | C607040020 | 1202 | MOV DISK_STATUS,BAD_CYLDR
0596 | F9 | 1203 | STC
059D | 1204 | CH7:
0598 | C3 | 1205 | RET
059D | 1211 | COMMAND ENDP
059D | 1212 | -----------------------------------------------
059D | 1213 | SENSE STATUS BYTES
059D | 1214 | -----------------------------------------------
059D | 1215 | BYTE 9
059D | 1216 | -----------------------------------------------
059D | 1217 | BYTE 6
059D | 1218 | -----------------------------------------------
059D | 1219 | BYTE 5
059D | 1220 | -----------------------------------------------
059D | 1221 | BYTE 4
059D | 1222 | -----------------------------------------------
059D | 1223 | BYTE 3
059D | 1224 | -----------------------------------------------
059D | 1225 | BYTE 2
059D | 1226 | -----------------------------------------------
059D | 1227 | BYTE 1
059D | 1228 | -----------------------------------------------
059D | 1229 | BYTE 0

Fixed Disk BIOS  A-99
A-100  Fixed Disk BIOS
LOC  OBJ  1304  TYPE_TBL EQU 6-TYPE_TBL
0057  1305  TYPE_TBL LABEL BYTE
0067 290010  1306  DB BAD_CHKHL,BAD_CHKHR,BAD_ECC
0063  1307  TYPE_LEN EQU 6-TYPE_TABLE
0108  i------- TYPE 0 ERROR
0110
0114  TYPE_0:
0118  080206  MOV BX,OFFSET TYPE_TBL
0122  010C  CMP AL,TYPE_LEN ; CHECK IF ERROR IS DEFINED
0126  011F 7653  JAE UNDEF_ERR_L
0130  021E 2077  XLAT CS:TYPE_TBL ; TABLE LOOKUP
0134  0232 A27400  MOV DISK_STATUS,AL ; SET ERROR CODE
0138  0236 C3  RET
0139
0141  i------- TYPE 1 ERROR
0142
0146  TYPE_1:
0150  0278 800006  MOV BX,OFFSET TYPE_TBL
0154  02A0  0C8C  MOV CX,AX
0158  02C2 7954  CMP AL,TYPE_LEN ; CHECK IF ERROR IS DEFINED
0162  02E0 2077  JAE UNDEF_ERR_L
0166  031E 2077  XLAT CS:TYPE_TBL ; TABLE LOOKUP
0170  0332 A27400  MOV DISK_STATUS,AL ; SET ERROR CODE
0174  0336 8E1080  AND CL,80H ; CORRECTED ECC
0178  0358 8FF080  CMP CL,80H
0182  0370 75E8  JNZ 630
0183
0185  i------- OBTAIN ECC ERROR BURST LENGTH
0186  03CD 606460800D  MOV CHS_BLOCK+4,RO_ECC_CNT
0190  0422 6AC0  MOV SUB AL,AL
0194  0444 81B9F7  CALL COMMAND
0198  0467 721E  JC 630
019A  0499 E3E000  CALL NO_WAIT_REQ
019C  04C4 7219  JC 630
019E  04E2 E82018  CALL PORT 0
01A0  0515 EC  IN AL,DX
01A2  0522 8AC6  MOV CL,AL
01A4  0548 8E3010  CALL NO_WAIT_REQ
01A6  0567 7218  JC 630
01A8  0589 81B501  CALL PORT 0
01AA  05AC EC  IN AL,DX
01AC  05D0 4001  TEST AL,81H
01AA  05F7 7400  JC 630
01A0  0611 C606740028  MOV DISK_STATUS,BAD_CHKLR
01A4  0666 F0  BTCH
01A6  0667 68C1  MOV AL,CL
01A8  0669 C3  RET
01A9
01B0  i------- TYPE 2 ERROR
01B1
01B5  TYPE_2:
01B9  066A 801506  MOV BX,OFFSET TYPE_TBL
01BE  066D 3C08  CMP AL,TYPE_LEN ; CHECK IF ERROR IS DEFINED
01BF  066F 7313  JAE UNDEF_ERR_L
01C0  0671 2077  XLAT CS:TYPE_TBL ; TABLE LOOKUP
01C4  0675 A27400  MOV DISK_STATUS,AL ; SET ERROR CODE
01C8  0676 C3  RET
01C9
01DA  i------- TYPE 3 ERROR
01DC
01DF  TYPE_3:
01E1  0677 081706  MOV BX,OFFSET TYPE_TBL
01E6  067A 3C03  CMP AL,TYPE_LEN
01E7  067C 7306  JAE UNDEF_ERR_L
01E8  067E 2077  XLAT CS:TYPE_TBL
01E9  0680 A27400  MOV DISK_STATUS,AL
01EA  0683 C3  RET
01EB
01F0  UNDEF_ERR_L:
01F4  0690 C6067400BB  MOV DISK_STATUS,UNDEF_ERR
01F8  0699 C3  RET
01FA
0200  NO_WAIT_REQ PROC NEAR
0204  06A8 31  PUSH CX
0205
0206

Appendix A

Fixed Disk BIOS  A-101
LOC OBJ  
LINE  
SOURCE

068D DBC9  
068D EEEE00  
0690  
0690 EC  
0691 A001  
0695 700B  
0697 C06740000  
069C F9  
069D  
069D 59  
069E C3  
06A0  
06A0 8B  
06A0 A04600  
06A3 3C01  
06A5 60  
06A6 7002  
06A8 F9  
06A9 C5  
06AA  
06AA 81  
06AB FA  
06AC 600C  
06AE 50  
06AF 60  
06B0 E00B  
06B2 BCC0  
06B4 B104  
06B6 D3C0  
06B8 BAE8  
06B9 24F0  
06BC 33C3  
06BE 7002  
06BF FE05  
06C0  
06C2  
06C2 50  
06C3 E006  
06C5 6ACC  
06C7 E006  
06C9 6ACC  
06CB 10F0  
06CD 1082  
06D0  
06D0 8400D  
06D2 0000  
06D4 6E00  
06D6 0000  
06DF  
06E0  
06E0 A04600  
06E0 8C15  
06E0 7407  
06E2 3C06  
06E4 7403  
06E5 50  
06E7 E011  
06E9  
06EA  
06F0 53  

0D81  SUB CX,CX  
0D82  CALL PORT_1  
0D83  LI:  
0D84  IN AL,DX  
0D85  TEST AL,RI_REQ  
0D86  JNZ L2  
0D87  LOOP L1  
0D88  MOV DISK_STATUS,TIME_OUT  
0D89  STC  
0D8A  L2:  
0D8B  POP CX  
0D8C  RET  
0D8D  NO_WAIT_REQ ENDP  
0D8E  ID:  
0D8F  DNA_SETUP  
0D90  THIS ROUTINE SETS UP FOR DMA OPERATIONS.  
0D91  INPUT  
0D92  (AL) = MODE BYTE FOR THE DMA  
0D93  (ES:BX) = ADDRESS TO READ/WRITE THE DATA  
0D94  OUTPUT  
0D95  (AX) DESTROYED  
0D96  ID:  
0D9F  DNA_SETUuP  
0DA0  PROC NEAR  
0DA1  PUSH AX  
0DA2  MDV AL,CMD_BLOCK+4  
0DA3  CMP AL,01H  
0DA4  JBL COUNT OUT OF RANGE  
0DA5  POP AX  
0DA6  JB J1  
0DA7  STC  
0DA8  RET  
0DA9  J3:  
0DAA  J3:  
0DAB  PUSH CX  
0DAD  CLI  
0DAE  OUT DMA+12,AL  
0DF0  JPL FIRST/LAST F/P  
0DF1  PUSH AX  
0DF2  MDV AL,CMD_BLOCK+4  
0DF3  OUT DMA+11,AL  
0DF4  OUTPUT THE MODE BYTE  
0DF5  MDV AX,ES  
0DF6  GET THE ES VALUE  
0DF7  MDV CL,AL  
0DF8  SHIFT COUNT  
0DF9  RCL AX,CL  
0DAE  ROTATE LEFT  
0DFB  MDV CH,CL  
0DFC  GET HIGHEST NUMBLE OF ES TO CH  
0DFD  AND AL,BFM  
0DFE  ZERO THE LOW WORD OF COMMAND  
0E00  ADD AL,ES  
0E01  TEST FOR CARRY FROM ADDITION  
0E02  JNC J33  
0E03  J33:  
0E04  INE CH  
0E05  CARRY MEANS HIGH 4 BITS MUST BE INC  
0E06  J33:  
0E07  PUSH AX  
0E08  SAVE START ADDRESS  
0E09  OUT DMA+6,AL  
0E0A  OUTPUT LOW ADDRESS  
0E0B  MDV AL,ALH  
0E0C  GET ADDRESS  
0E0D  OUT DMA+5,AL  
0E0E  OUTPUT HIGH ADDRESS  
0E0F  MDV AL,CH  
0E10  LOC  
0E11  AND AL,BFM  
0E12  GET HIGH 4 BITS  
0E13  OUT DMA,HIGH,AL  
0E14  OUTPUT THE HIGH 4 BITS TO PAGE REG  
0E15  1----- DETERMINE COUNT  
0E16  1----- HANDLE READ AND WRITE LONG (5160 BYTE BLOCKS)  
0E17  1-----  
0E18  1----- SAVE REGISTER  
0E19  1----- GET COMMAND  
0E1A  1-----  
0E1B  1----- RESTORE REGISTER  
0E1C  1----- RESTORE REGISTER  
0E1D  1-----  
0E1E  1----- ONE BLOCK (512) PLUS 4 BYTES ECC  
0E1F  1-----  

A-102  Fixed Disk BIOS
LOC   OBJ   LINE   SOURCE

065A  ZAFF   1458   SUB   BL,BH
065B  BAI64600  1459   MOV   BL,CMP_BLOCK+4
065C  52   1460   PUSH   BX
065D  7F73   1461   MUL   BX
065E  5A   1462   POP   BX
065F  5B   1463   POP   BX
0660  4B   1464   DEC   AX
0661  5A   1465   ;JMP
0662  5A   1466
0663  5A   1467   PUSH   AX
0664  E607   1468   OUT   DMA+7,AL
0665  94   1469   MOV   AL,AH
0666  E607   1470   OUT   DMA+7,AL
0701  FB   1471   STI   ;INTERRUPTS BACK ON
0702  59   1472   POP   CX
0703  5A   1473   POP   AX
0704  03C1   1474   ADD   AX,CX
0705  59   1475   POP   CX
0706  C3   1476   RET
0707  B8   1477   ;RETURN TO CALLER, CPU SET BY ABOVE IF ERROR
0708  39   1478   ;DMA_SETUP
0709  3C   1479
070A  5B   1480   ;--------------------------
070B  00   1481   ;      WAIT_INT
070C  59   1482   ;      THIS ROUTINE WAITS FOR THE FIXED DISK
070D  5B   1483   ;      CONTROLLER TO SIGNAL THAT AN INTERRUPT
070E  60   1484   ;      HAS OCCURRED.
070F  30   1485
0710  2C0   1486   ;PROE_NEAR
0711  53   1487   STI   ;TURN ON INTERRUPTS
0712  50   1488   PUSH   BX
0713  50   1489   PUSH   CX
0714  50   1490   PUSH   SI
0715  50   1491   PUSH   DS
0716  53   1492   ASSUME   DS:0000
0717  0600   1493   MOV   DS,AX
0718  2B00   1494   Establish   SEGMENT
0719  C6364400  1495   LES   SI,FS_TLB_VEC
071A  59   1496   ASSUME   DS:D ata
071B  5B   1497   POP   DS
071C  C9   1498
071D  1F   1499   ;SET TIMEOUT VALUES
071E  32   1500
071F  2AFF   1501   SUB   BH,BH
0720  6645C009  1502   MOV   BL,BYTE_PTR BP,E8131110H
0721  60442000  1503   MOV   AH,CMP_BLOCK
0722  65FC04  1504   CMP   AH,FATCHED
0723  7504   1505   JNZ   NS
0724  6645C0A  1506   MOV   BL,BYTE_PTR BP,E8131110H
0725  7504   1507   JNZ   NS
0726  E009   1508   CMP   AH,DISK_DRV_CND
0727  7500   1509   JNZ   NS
0728  6645C08  1510   MOV   BL,BYTE_PTR BP,E8131110H
0729  751F   1511   JNZ   NS
072A  2B09   1512   SUB   CX,CL
072B  C9   1513
072C  1F   1514   ;WAIT FOR INTERRUPT
072D  32   1515
072E  0C46460  1516   CALL   PORT_1
072F  3C   1517   IN   AL,DX
0730  2420   1518   AND   AL,020H
0731  3C00   1519   CMP   AL,020H
0732  740A   1520   JZ   NZ
0733  E044   1521   LOOP   NL
0734  4B   1522   DEC   BX
0735  7F14   1523   JNZ   NL
0736  C667640008  1524   MOV   DISK_STATUS,TIME_OUT
0737  2B09   1525
0738  0C46460  1526   CALL   PORT_2
0739  3C   1527   IN   AL,DX
073A  2402   1528   AND   AL,2
073B  80067400  1529   OR   DISK_STATUS,AL
073C  E53000  1530   CALL   PORT_3
073D  3C00   1531   XOR   AL,AL
073E  EE   1532   OUT   BX,AL
073F  5E   1533
0740  59   1534   POP   SI

Appendix A

Fixed Disk BIOS  A-103
LOC OBJ LINE SOURCE

075C 97 1535 POP ES
075D 59 1536 POP CX
075E 5B 1537 POP DX
075F C3 1538 RET
1539 Halt_INT
1540 ENDP
075A 1541 x x x x
075B 80 1542 x x x x
0761 000A 1543 MOV AL,00H
0763 6E20 1544 OUT INT_CTL_PORT,AL
0765 0007 1545 MOV AL,07H
0767 6E0A 1546 OUT DMA+10H,AL
0769 6E11 1547 IN AL,021H
076A 0210 1548 DX AL,021H
076B 6E11 1549 OUT DX,AL
076F 50 1550 POP AX
0770 CF 1551 IRET
1552 x x x x
1553 x x x x
1554 1 PORTS
1555 1 GENERATE PROPER PORT VALUE
1556 1 BASED ON THE PORT OFFSET
1557 1 x x x x
1558 1 x x x x
1559 1 x x x x
1560 1 x x x x
1561 1 x x x x
1562 1 x x x x
1563 1 x x x x
1564 1 x x x x
1565 1 x x x x
1566 1 x x x x
1567 1 x x x x
1568 1 x x x x
1569 1 x x x x
1570 1 x x x x
1571 + x x x x
1572 1 x x x x
1573 1 x x x x
1574 1 x x x x
1575 1 x x x x
1576 1 x x x x
1577 1 x x x x
1578 1 x x x x
1579 1 x x x x
1580 1 x x x x
1581 1 x x x x
1582 1 x x x x
1583 1 x x x x
1584 1 x x x x
1585 1 x x x x
1586 1 x x x x
1587 1 x x x x
1588 1 x x x x
1589 1 x x x x
1590 1 x x x x
1591 1 x x x x
1592 1 x x x x
1593 1 x x x x
1594 1 x x x x
1595 1 x x x x
1596 1 x x x x
1597 1 x x x x
1598 1 x x x x
1599 1 x x x x
1600 1 x x x x
1601 1 x x x x
1602 1 x x x x
1603 1 x x x x
1604 1 x x x x
1605 1 x x x x
1606 1 x x x x
1607 1 x x x x
1608 1 x x x x
1609 1 x x x x
1610 1 x x x x
1611 1 x x x x

A-104 Fixed Disk BIOS
<table>
<thead>
<tr>
<th>LOC OBJ</th>
<th>LINE</th>
<th>SOURCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>07AC D220</td>
<td>1612</td>
<td>SHL AL,CL ; ADJUST</td>
</tr>
<tr>
<td>07AE 2A44</td>
<td>1613</td>
<td>SUB AH,AN</td>
</tr>
<tr>
<td>07B0 C3</td>
<td>1614</td>
<td>RET</td>
</tr>
<tr>
<td>07B1</td>
<td>1615</td>
<td>SJZ_OFFS_END:</td>
</tr>
<tr>
<td>07B1 F9</td>
<td>1616</td>
<td>STC</td>
</tr>
<tr>
<td>07B2 C5</td>
<td>1617</td>
<td>RET</td>
</tr>
<tr>
<td>07B5 30302F931362F50 32</td>
<td>1618</td>
<td>SJZ_OFFS ENDP</td>
</tr>
<tr>
<td>07BB</td>
<td>1620</td>
<td>DB '08/16/82' ; RELEASE MARKER</td>
</tr>
<tr>
<td>----</td>
<td>1621</td>
<td></td>
</tr>
<tr>
<td>07BB</td>
<td>1622</td>
<td>END_ADDRESS LABEL BYTE</td>
</tr>
<tr>
<td>----</td>
<td>1623</td>
<td>CODE ENDS</td>
</tr>
<tr>
<td>1624</td>
<td>END</td>
<td></td>
</tr>
</tbody>
</table>
Notes:
### 8088 Register Model

<table>
<thead>
<tr>
<th>AX:</th>
<th>AH</th>
<th>AL</th>
<th>Accumulator</th>
</tr>
</thead>
<tbody>
<tr>
<td>BX:</td>
<td>BH</td>
<td>BL</td>
<td>Base</td>
</tr>
<tr>
<td>CX:</td>
<td>CH</td>
<td>CL</td>
<td>Count</td>
</tr>
<tr>
<td>DX:</td>
<td>DH</td>
<td>DL</td>
<td>Data</td>
</tr>
<tr>
<td></td>
<td>SP</td>
<td></td>
<td>Stack Pointer</td>
</tr>
<tr>
<td></td>
<td>BP</td>
<td></td>
<td>Base Pointer</td>
</tr>
<tr>
<td></td>
<td>SI</td>
<td></td>
<td>Source Index</td>
</tr>
<tr>
<td></td>
<td>DI</td>
<td></td>
<td>Destination Index</td>
</tr>
</tbody>
</table>

### General Register File

### Segment Register File

### Instruction Pointer

<table>
<thead>
<tr>
<th>IP</th>
<th>FALGSH</th>
<th>FALGSL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ES</td>
<td></td>
</tr>
</tbody>
</table>

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

\[
\begin{array}{cccccccccccccccc}
15 & X & X & X & X & OF & DF & IF & TF & SF & ZF & X & AF & X & PF & X & CF \\
\end{array}
\]

$X = \text{Don't Care}$

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

### 8080 Flags

### 8088 Flags

**DF**: Direction Flag (Strings)

**IF**: Interrupt Enable Flag

**OF**: Overflow Flag ($CF \oplus SF$)

**TF**: Trap - Single Step Flag

---

**B-2 8088 Instruction Reference**
# Operand Summary

"reg" field Bit Assignments:

<table>
<thead>
<tr>
<th>16-Bit (w=1)</th>
<th>8-Bit (w=0)</th>
<th>Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>000 AX</td>
<td>000 AL</td>
<td>00 ES</td>
</tr>
<tr>
<td>001 CX</td>
<td>001 CL</td>
<td>01 CS</td>
</tr>
<tr>
<td>010 DX</td>
<td>010 DL</td>
<td>10 SS</td>
</tr>
<tr>
<td>011 BX</td>
<td>011 BL</td>
<td>11 DS</td>
</tr>
<tr>
<td>100 SP</td>
<td>100 AH</td>
<td></td>
</tr>
<tr>
<td>101 BP</td>
<td>101 CH</td>
<td></td>
</tr>
<tr>
<td>110 SI</td>
<td>110 DH</td>
<td></td>
</tr>
<tr>
<td>111 DI</td>
<td>111 BH</td>
<td></td>
</tr>
</tbody>
</table>

# Second Instruction Byte Summary

<table>
<thead>
<tr>
<th>mod</th>
<th>xxx</th>
<th>r/m</th>
</tr>
</thead>
</table>

**mod** | **Displacement** |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>DISP = 0*, disp-low and disp-high are absent</td>
</tr>
<tr>
<td>01</td>
<td>DISP = disp-low sign-extended to 16-bits, disp-high is absent</td>
</tr>
<tr>
<td>10</td>
<td>DISP = disp-high: disp-low</td>
</tr>
<tr>
<td>11</td>
<td>r/m is treated as a &quot;reg&quot; field</td>
</tr>
</tbody>
</table>

**r/m** | **Operand Address** |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>(BX) + (SI) + DISP</td>
</tr>
<tr>
<td>001</td>
<td>(BX) + (DI) + DISP</td>
</tr>
<tr>
<td>010</td>
<td>(BP) + (SI) + DISP</td>
</tr>
<tr>
<td>011</td>
<td>(BP) + (DI) + DISP</td>
</tr>
<tr>
<td>100</td>
<td>(SI) + DISP</td>
</tr>
<tr>
<td>101</td>
<td>(DI) + DISP</td>
</tr>
<tr>
<td>110</td>
<td>(BP) + DISP*</td>
</tr>
<tr>
<td>111</td>
<td>(BX) + DISP</td>
</tr>
</tbody>
</table>

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

Segment Override Prefix

Use of Segment Override

<table>
<thead>
<tr>
<th>Operand Register</th>
<th>Default</th>
<th>With Override Prefix</th>
</tr>
</thead>
<tbody>
<tr>
<td>IP (Code Address)</td>
<td>CS</td>
<td>Never</td>
</tr>
<tr>
<td>SP (Stack Address)</td>
<td>SS</td>
<td>Never</td>
</tr>
<tr>
<td>BP (Stack Address or Stack Marker)</td>
<td>SS</td>
<td>BP + DS or ES, or CS</td>
</tr>
<tr>
<td>SI or DI (not including strings)</td>
<td>DS</td>
<td>ES, SS, or CS</td>
</tr>
<tr>
<td>SI (Implicit Source Address for Strings)</td>
<td>DS</td>
<td>ES, SS, or CS</td>
</tr>
<tr>
<td>DI (Implicit Destination Address for Strings)</td>
<td>ES</td>
<td>Never</td>
</tr>
</tbody>
</table>

B-4 8088 Instruction Reference
### Data Transfer

**MOV** = Move
Register/memory to/from register

<table>
<thead>
<tr>
<th>100010 d w</th>
<th>mod</th>
<th>reg</th>
<th>r/m</th>
</tr>
</thead>
</table>

**Immediate to register/memory**

<table>
<thead>
<tr>
<th>110011 w</th>
<th>mod</th>
<th>000 r/m</th>
<th>data</th>
<th>data if w=1</th>
</tr>
</thead>
</table>

**Immediate to register**

<table>
<thead>
<tr>
<th>1011 w</th>
<th>reg</th>
<th>data</th>
<th>data if w=1</th>
</tr>
</thead>
</table>

**Memory to accumulator**

<table>
<thead>
<tr>
<th>1010000 w</th>
<th>addr-low</th>
<th>addr-high</th>
</tr>
</thead>
</table>

**Accumulator to memory**

<table>
<thead>
<tr>
<th>1010001 w</th>
<th>addr-low</th>
<th>addr-high</th>
</tr>
</thead>
</table>

**Register/memory to segment register**

<table>
<thead>
<tr>
<th>10001110</th>
<th>mod</th>
<th>0 reg</th>
<th>r/m</th>
</tr>
</thead>
</table>

**Segment register to register/memory**

<table>
<thead>
<tr>
<th>10001100</th>
<th>mod</th>
<th>0 reg</th>
<th>r/m</th>
</tr>
</thead>
</table>

**PUSH** = Push
Register/memory

<table>
<thead>
<tr>
<th>11111111 w</th>
<th>mod</th>
<th>110 r/m</th>
</tr>
</thead>
</table>

**Register**

<table>
<thead>
<tr>
<th>01010 reg</th>
</tr>
</thead>
</table>

**Segment register**

<table>
<thead>
<tr>
<th>000 reg 110</th>
</tr>
</thead>
</table>

**POP** = Pop
Register/memory

<table>
<thead>
<tr>
<th>1000111 mod</th>
<th>000 r/m</th>
</tr>
</thead>
</table>

**Register**

<table>
<thead>
<tr>
<th>01011 reg</th>
</tr>
</thead>
</table>

**Segment register**

<table>
<thead>
<tr>
<th>000 reg 111</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instruction</td>
</tr>
<tr>
<td>-------------</td>
</tr>
<tr>
<td>XCHG</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>IN</td>
</tr>
<tr>
<td>OUT</td>
</tr>
<tr>
<td>Variable port (DX)</td>
</tr>
<tr>
<td>OUT</td>
</tr>
<tr>
<td>Variable port (DX)</td>
</tr>
<tr>
<td>XLAT</td>
</tr>
<tr>
<td>LEA</td>
</tr>
<tr>
<td>LDS</td>
</tr>
<tr>
<td>LES</td>
</tr>
<tr>
<td>LAHF</td>
</tr>
<tr>
<td>SAHF</td>
</tr>
<tr>
<td>PUSHF</td>
</tr>
<tr>
<td>POPF</td>
</tr>
</tbody>
</table>
Arithmetic

***ADD*** = Add
Register/memory with register to either

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

Immediate to register/memory

```
1 0 0 0 0 0 s w  mod 0 0 0 r/m  data  data if s:w=01
```

Immediate to accumulator

```
0 0 0 0 0 1 0 w  data  data if w=1
```

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

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

Immediate to register/memory

```
1 0 0 0 0 0 s w  mod 0 1 0 r/m  data  data if s:w=01
```

Immediate to accumulator

```
0 0 0 1 0 1 0 w  data  data if w=1
```

***INC*** = Increment
Register/memory

```
1 1 1 1 1 1 1 w  mod 0 0 0 r/m
```

Register

```
0 1 0 0 0 reg
```

***AAA*** = ASCII adjust for add

```
0 0 1 1 0 1 1 1
```

***DAA*** = Decimal adjust for add

```
0 0 1 0 0 1 1 1
```

***SUB*** = Subtract
Register/memory and register to either

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

Immediate from register/memory

```
1 0 0 0 0 0 s w  mod 1 0 1 r/m  data  data if s:w=01
```

Immediate from accumulator

```
0 0 1 0 1 1 0 w  data  data if w=1
```
<table>
<thead>
<tr>
<th>Opcode</th>
<th>Mod</th>
<th>Reg</th>
<th>R/M</th>
<th>Immediate</th>
</tr>
</thead>
<tbody>
<tr>
<td>SBB</td>
<td>00</td>
<td>11</td>
<td>0 d w</td>
<td>mod reg r/m</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Immediate from register/memory</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>00</td>
<td>0 0 s w</td>
<td>mod 0 1 r/m</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Immediate if s:w=01</td>
</tr>
<tr>
<td></td>
<td>00</td>
<td>11</td>
<td>1 0 w</td>
<td>data</td>
</tr>
<tr>
<td>DEC</td>
<td>11</td>
<td>11</td>
<td>1 1 1 1 w</td>
<td>mod 0 0 r/m</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Register</td>
</tr>
<tr>
<td></td>
<td>01</td>
<td>00</td>
<td>1 reg</td>
<td></td>
</tr>
<tr>
<td>NEG</td>
<td>11</td>
<td>11</td>
<td>0 1 1 w</td>
<td>mod 0 1 r/m</td>
</tr>
<tr>
<td>CMP</td>
<td>00</td>
<td>11</td>
<td>1 1 0 d w</td>
<td>mod reg r/m</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Immediate with register/memory</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>00</td>
<td>0 0 s w</td>
<td>mod 1 1 r/m</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Immediate if s:w=01</td>
</tr>
<tr>
<td></td>
<td>00</td>
<td>11</td>
<td>1 1 0 w</td>
<td>data</td>
</tr>
<tr>
<td>AAS</td>
<td>00</td>
<td>11</td>
<td>1 1 1 1</td>
<td></td>
</tr>
<tr>
<td>DAS</td>
<td>00</td>
<td>10</td>
<td>1 1 1 1</td>
<td></td>
</tr>
<tr>
<td>MUL</td>
<td>11</td>
<td>11</td>
<td>0 1 1 w</td>
<td>mod 1 0 0 r/m</td>
</tr>
<tr>
<td>IMUL</td>
<td>11</td>
<td>11</td>
<td>0 1 1 w</td>
<td>mod 1 0 1 r/m</td>
</tr>
<tr>
<td>AAM</td>
<td>11</td>
<td>01</td>
<td>0 1 0 0 0 0 0 1 0 1 0</td>
<td></td>
</tr>
<tr>
<td>DIV</td>
<td>11</td>
<td>11</td>
<td>0 1 1 w</td>
<td>mod 1 1 0 r/m</td>
</tr>
</tbody>
</table>

**B-8 8088 Instruction Reference**
IDIV = Integer divide (signed)

```
1 1 1 1 0 1 1 w  mod 1 1 1 r/m
```

AAD = ASCII adjust for divide

```
1 1 0 1 0 1 0 1 0 0 0 1 0 1 0
```

CBW = Convert byte to word

```
1 0 0 1 1 0 0 0
```

CWD = Convert word to double word

```
1 0 0 1 1 0 0 1
```

Logic

NOT = Invert

```
1 1 1 1 0 1 1 w  mod 0 1 0 r/m
```

SHL/SAL = Shift logical/arithmetic left

```
1 1 0 1 0 0 v w  mod 1 0 0 r/m
```

SHR = Shift logical right

```
1 1 0 1 0 0 v w  mod 1 0 1 r/m
```

SAR = Shift arithmetic right

```
1 1 0 1 0 0 v w  mod 1 1 1 r/m
```

ROL = Rotate left

```
1 1 0 1 0 0 v w  mod 0 0 0 r/m
```

ROR = Rotate right

```
1 1 0 1 0 0 v w  mod 0 0 1 r/m
```

RCL = Rotate through carry left

```
1 1 0 1 0 0 v w  mod 0 1 0 r/m
```

RCR = Rotate through carry right

```
1 1 0 1 0 0 v w  mod 0 1 1 r/m
```

AND = And

```
0 0 1 0 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

B-10 Instruction Reference
**LODS = Load String**
```
1010110
```

**STOS = Store String**
```
1010101
```

### Control Transfer

**CALL = Call**
- Direct within segment
  ```
  11101000 disp-low disp-high
  ```
- Indirect within segment
  ```
  11111111 mod 010 r/m
  ```
- Direct intersegment
  ```
  10011010 offset-low offset-high
  ```
  ```
  seg-low seg-high
  ```
- Indirect intersegment
  ```
  11111111 mod 011 r/m
  ```

**JMP = Unconditional Jump**
- Direct within segment
  ```
  11101001 disp-low disp-high
  ```
- Direct within segment-short
  ```
  11101011 disp
  ```
- Indirect within segment
  ```
  11111111 mod 100 r/m
  ```
- Direct intersegment
  ```
  11101010 offset-low offset-high
  ```
  ```
  seg-low seg-high
  ```
- Indirect intersegment
  ```
  11111111 mod 101 r/m
  ```
\textbf{RET} = Return from CALL  
Within segment  
\begin{center}
\begin{tabular}{|c|c|c|}
\hline
\multicolumn{3}{|c|}{1 1 0 0 0 0 0 1 1} \\
\hline
\end{tabular}
\end{center}

Within segment adding immediate to SP  
\begin{center}
\begin{tabular}{|c|c|c|}
\hline
\multicolumn{2}{|c|}{1 1 0 0 0 0 0 1 0} & \text{data-low} & \text{data-high} \\
\hline
\end{tabular}
\end{center}

Intersegment  
\begin{center}
\begin{tabular}{|c|c|c|c|c|c|}
\hline
\multicolumn{3}{|c|}{1 1 0 0} & \textbf{1} & \textbf{0} & 1 1 \\
\hline
\end{tabular}
\end{center}

Intersegment, adding immediate to SP  
\begin{center}
\begin{tabular}{|c|c|c|c|c|c|}
\hline
\multicolumn{2}{|c|}{1 1 0 0} & \textbf{0 0} & \textbf{0 1 0} & \textbf{1 0} & \textbf{1 1} \text{disp} \\
\hline
\end{tabular}
\end{center}

\textbf{JE/JZ} = Jump on equal/zero  
\begin{center}
\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
\multicolumn{3}{|c|}{0} & \textbf{1} & \textbf{1} & \textbf{1} & \textbf{1} \text{disp} \\
\hline
\end{tabular}
\end{center}

\textbf{JL/JNGE} = Jump on less/not greater or equal  
\begin{center}
\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
\multicolumn{3}{|c|}{0} & \textbf{1} & \textbf{1} & \textbf{1} & \textbf{1} \text{disp} \\
\hline
\end{tabular}
\end{center}

\textbf{JLE/JNG} = Jump on less or equal/not greater  
\begin{center}
\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
\multicolumn{3}{|c|}{0} & \textbf{1} & \textbf{1} & \textbf{1} & \textbf{1} \text{disp} \\
\hline
\end{tabular}
\end{center}

\textbf{JB/JNAE} = Jump on below/not above or equal  
\begin{center}
\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
\multicolumn{3}{|c|}{0} & \textbf{1} & \textbf{1} & \textbf{1} & \textbf{0} \text{disp} \\
\hline
\end{tabular}
\end{center}

\textbf{JBE/JNA} = Jump on below or equal/not above  
\begin{center}
\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
\multicolumn{3}{|c|}{0} & \textbf{1} & \textbf{1} & \textbf{1} & \textbf{0} \text{disp} \\
\hline
\end{tabular}
\end{center}

\textbf{JP/JPE} = Jump on parity/parity even  
\begin{center}
\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
\multicolumn{3}{|c|}{0} & \textbf{1} & \textbf{1} & \textbf{1} & \textbf{0} \text{disp} \\
\hline
\end{tabular}
\end{center}

\textbf{JO} = Jump on overflow  
\begin{center}
\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
\multicolumn{3}{|c|}{0} & \textbf{1} & \textbf{1} & \textbf{1} & \textbf{0} \text{disp} \\
\hline
\end{tabular}
\end{center}

\textbf{JS} = Jump on sign  
\begin{center}
\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
\multicolumn{3}{|c|}{0} & \textbf{1} & \textbf{1} & \textbf{1} & \textbf{0} \text{disp} \\
\hline
\end{tabular}
\end{center}

\textbf{JNE/JNZ} = Jump on not equal/not zero  
\begin{center}
\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
\multicolumn{3}{|c|}{0} & \textbf{1} & \textbf{1} & \textbf{1} & \textbf{0} \text{disp} \\
\hline
\end{tabular}
\end{center}

\textbf{JNL/JGE} = Jump on not less/greater or equal  
\begin{center}
\begin{tabular}{|c|c|c|c|c|c|c|}
\hline
\multicolumn{3}{|c|}{0} & \textbf{1} & \textbf{1} & \textbf{1} & \textbf{1} \text{disp} \\
\hline
\end{tabular}
\end{center}
<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
<th>Op Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>JNLE/JG</td>
<td>Jump on not less or equal/greater</td>
<td>0 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>JNB/JAE</td>
<td>Jump on not below/above or equal</td>
<td>0 1 1 1 0 0 1 1</td>
</tr>
<tr>
<td>JNBE/JA</td>
<td>Jump on not below or equal/above</td>
<td>0 1 1 1 0 1 1 1</td>
</tr>
<tr>
<td>JNP/JPO</td>
<td>Jump on not parity/parity odd</td>
<td>0 1 1 1 1 0 1 1</td>
</tr>
<tr>
<td>JNO</td>
<td>Jump on not overflow</td>
<td>0 1 1 1 1 0 0 1</td>
</tr>
<tr>
<td>JNS</td>
<td>Jump on not sign</td>
<td>0 1 1 1 1 0 0 1</td>
</tr>
<tr>
<td>LOOP</td>
<td>Loop CX times</td>
<td>1 1 1 0 0 0 1 0</td>
</tr>
<tr>
<td>LOOPZ/LOOPE</td>
<td>Loop while zero/equal</td>
<td>1 1 1 0 0 0 0 1</td>
</tr>
<tr>
<td>LOOPNZ/LOOPNE</td>
<td>Loop while not zero/not equal</td>
<td>1 1 1 0 0 0 0 0</td>
</tr>
<tr>
<td>JCXZ</td>
<td>Jump on CX zero</td>
<td>1 1 1 0 0 0 1 1</td>
</tr>
</tbody>
</table>
### 8088 Conditional Transfer Operations

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Condition</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>JE or JZ</td>
<td>ZF = 1</td>
<td>&quot;equal&quot; or &quot;zero&quot;</td>
</tr>
<tr>
<td>JL or JNGE</td>
<td>(SF xor OF) = 1</td>
<td>&quot;less&quot; or &quot;not greater or equal&quot;</td>
</tr>
<tr>
<td>JLE or JNG</td>
<td>((SF xor OF) or ZF) = 1</td>
<td>&quot;less or equal&quot; or &quot;not greater&quot;</td>
</tr>
<tr>
<td>JB or JNAE or JC</td>
<td>CF = 1</td>
<td>&quot;below&quot; or &quot;not above or equal&quot;</td>
</tr>
<tr>
<td>JBE or JNA</td>
<td>(CF or ZF) = 1</td>
<td>&quot;below or equal&quot; or &quot;not above&quot;</td>
</tr>
<tr>
<td>JP or JPE</td>
<td>PF = 1</td>
<td>&quot;parity&quot; or &quot;parity even&quot;</td>
</tr>
<tr>
<td>JO</td>
<td>OF = 1</td>
<td>&quot;overflow&quot;</td>
</tr>
<tr>
<td>JS</td>
<td>SF = 1</td>
<td>&quot;sign&quot;</td>
</tr>
<tr>
<td>JNE or JNZ</td>
<td>ZF = 0</td>
<td>&quot;not equal&quot; or &quot;not zero&quot;</td>
</tr>
<tr>
<td>JNLE or JG</td>
<td>((SF xor OF) or ZF) = 0</td>
<td>&quot;not less or greater or equal&quot;</td>
</tr>
<tr>
<td>JNB or JAE or JNC</td>
<td>CF = 0</td>
<td>&quot;not less or equal&quot; or &quot;greater&quot;</td>
</tr>
<tr>
<td>JNBE or JA</td>
<td>(CF or ZF) = 0</td>
<td>&quot;not below or above or equal&quot;</td>
</tr>
<tr>
<td>JNP or JPO</td>
<td>PF = 0</td>
<td>&quot;not below or equal&quot; or &quot;above&quot;</td>
</tr>
<tr>
<td>JNO</td>
<td>OF = 0</td>
<td>&quot;not parity&quot; or &quot;parity odd&quot;</td>
</tr>
<tr>
<td>JNS</td>
<td>SF = 0</td>
<td>&quot;not overflow&quot;</td>
</tr>
</tbody>
</table>

**"Above" and "below" refer to the relation between two unsigned values, while "greater" and "less" refer to the relation between two signed values.**

**INT = Interrupt**
Type specified

```
1 1 0 0 1 1 0 1
```

**Type 3**

```
1 1 0 0 1 1 0 0
```

**INTO = Interrupt on overflow**

```
1 1 0 0 1 1 1 0
```

**IRET = Interrupt return**

```
1 1 0 0 1 1 1 1
```
### Processor Control

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Binary Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLC = Clear carry</td>
<td>1111110000</td>
</tr>
<tr>
<td>STC = Set carry</td>
<td>1111110001</td>
</tr>
<tr>
<td>CMC = Complement carry</td>
<td>111110101</td>
</tr>
<tr>
<td>NOP = No operation</td>
<td>100100000</td>
</tr>
<tr>
<td>CLD = Clear direction</td>
<td>111111100</td>
</tr>
<tr>
<td>STD = Set direction</td>
<td>111111101</td>
</tr>
<tr>
<td>CLI = Clear interrupt</td>
<td>11111010</td>
</tr>
<tr>
<td>STI = Set interrupt</td>
<td>11111011</td>
</tr>
<tr>
<td>HLT = Halt</td>
<td>111101000</td>
</tr>
<tr>
<td>WAIT = Wait</td>
<td>10011011</td>
</tr>
<tr>
<td>LOCK = Bus lock prefix</td>
<td>11110000</td>
</tr>
<tr>
<td>ESC = Escape (to external device)</td>
<td>11011xx mod x x r/m</td>
</tr>
</tbody>
</table>

Footnotes:
- If d = 1 then "to"; if d = 0 then "from"
- If w = 1 then word instruction; if w = 0 then byte instruction
- If s:w = 01 then 16 bits of immediate data from the operand
- If s:w = 11 then an immediate data byte is sign extended to form the 16-bit operand
- If v = 0 then "count" = 1; if v = 1 then "count" in (CL)
- x = don’t care
- z is used for some string primitives to compare with ZF flag
- AL = 8-bit accumulator
- AX = 16-bit accumulator
- CX = Count register
- DS = Data segment
- DX = Variable port register
- ES = Extra segment
- Above/below refers to unsigned value
- Greater = more positive;
- Less = less positive (more negative) signed values
### 8088 Instruction Set Matrix

<table>
<thead>
<tr>
<th>LO</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ADD b,f,r/m</td>
<td>ADD w,f,r/m</td>
<td>ADD b,t,r/m</td>
<td>ADD w,t,r/m</td>
<td>ADD b,ia</td>
<td>ADD w,ia</td>
<td>PUSH</td>
<td>POP</td>
</tr>
<tr>
<td>1</td>
<td>ADC b,f,r/m</td>
<td>ADC w,f,r/m</td>
<td>ADC b,t,r/m</td>
<td>ADC w,t,r/m</td>
<td>ADC b,i</td>
<td>ADC w,i</td>
<td>PUSH SS</td>
<td>POP SS</td>
</tr>
<tr>
<td>2</td>
<td>AND b,f,r/m</td>
<td>AND w,f,r/m</td>
<td>AND b,t,r/m</td>
<td>AND w,t,r/m</td>
<td>AND b,i</td>
<td>AND w,i</td>
<td>SEG =ES</td>
<td>DAA</td>
</tr>
<tr>
<td>3</td>
<td>XOR b,f,r/m</td>
<td>XOR w,f,r/m</td>
<td>XOR b,t,r/m</td>
<td>XOR w,t,r/m</td>
<td>XOR b,i</td>
<td>XOR w,i</td>
<td>SEG =SS</td>
<td>AAA</td>
</tr>
<tr>
<td>4</td>
<td>INC AX</td>
<td>INC CX</td>
<td>INC DX</td>
<td>INC BX</td>
<td>INC SP</td>
<td>INC BP</td>
<td>INC SI</td>
<td>INC DI</td>
</tr>
<tr>
<td>5</td>
<td>PUSH AX</td>
<td>PUSH CX</td>
<td>PUSH DX</td>
<td>PUSH BX</td>
<td>PUSH SP</td>
<td>PUSH BP</td>
<td>PUSH SI</td>
<td>PUSH DI</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>JO</td>
<td>JNO</td>
<td>JB/JA</td>
<td>JNB/JAE</td>
<td>JE/JZ</td>
<td>JNE/JNZ</td>
<td>JBE/JNA</td>
<td>JNBE/JA</td>
</tr>
<tr>
<td>8</td>
<td>Immed b,r/m</td>
<td>Immed w,r/m</td>
<td>Immed b,r/m</td>
<td>Immed w,r/m</td>
<td>TEST b,r/m</td>
<td>TEST w,r/m</td>
<td>XCHG b,r/m</td>
<td>XCHG w,r/m</td>
</tr>
<tr>
<td>9</td>
<td>NOP</td>
<td>XCHG CX</td>
<td>XCHG DX</td>
<td>XCHG BX</td>
<td>XCHG SP</td>
<td>XCHG BP</td>
<td>XCHG SI</td>
<td>XCHG DI</td>
</tr>
<tr>
<td>A</td>
<td>MOV m AL</td>
<td>MOV m AX</td>
<td>MOV AL m</td>
<td>MOV AX m</td>
<td>MOVS b</td>
<td>MOVS w</td>
<td>CMPS b</td>
<td>CMPS w</td>
</tr>
<tr>
<td>B</td>
<td>MOV i AL</td>
<td>MOV i CL</td>
<td>MOV i DL</td>
<td>MOV i BL</td>
<td>MOV i AH</td>
<td>MOV i CH</td>
<td>MOV i DH</td>
<td>MOV i BH</td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>Shift b</td>
<td>Shift b,v</td>
<td>Shift w</td>
<td>Shift w,v</td>
<td>AAM</td>
<td>AAD</td>
<td>MOV b,i,r/m</td>
<td>MOV w,i,r/m</td>
</tr>
<tr>
<td>E</td>
<td>LOOPNZ LOOPNE</td>
<td>LOOPZ LOOPPE</td>
<td>LOOP</td>
<td>JCXZ</td>
<td>IN b</td>
<td>IN w</td>
<td>OUT b</td>
<td>OUT w</td>
</tr>
<tr>
<td>F</td>
<td>LOCK</td>
<td>REP</td>
<td>REP z</td>
<td>HLT</td>
<td>CMC</td>
<td>Grp 1 b,r/m</td>
<td>Grp 1 w,r/m</td>
<td></td>
</tr>
</tbody>
</table>

- **b** = byte operation
- **m** = memory
- **d** = direct
- **r/m** = EA is second byte
- **f** = from CPU reg
- **si** = short intrasegment
- **i** = immediate
- **sr** = segment register
- **ia** = immed. to accum.
- **t** = to CPU reg
- **id** = indirect
- **v** = variable
- **is** = immed. byte, sign ext.
- **w** = word operation
- **l** = long ie. intersegment
- **z** = zero

---

**B-16 8088 Instruction Reference**
### 8088 Instruction Set Matrix

<table>
<thead>
<tr>
<th>HI</th>
<th>LO</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>OR</td>
<td>b,f,r/m</td>
<td>w,f,r/m</td>
<td>OR</td>
<td>b,t,r/m</td>
<td>OR</td>
<td>b,t,r/m</td>
<td>OR</td>
<td>b,i</td>
<td>OR</td>
<td>w,i</td>
<td>PUSH</td>
<td>CS</td>
<td>POP</td>
<td>DS</td>
<td>POP</td>
<td>DS</td>
<td></td>
</tr>
<tr>
<td>SBB</td>
<td>b,f,r/m</td>
<td>w,f,r/m</td>
<td>SBB</td>
<td>b,t,r/m</td>
<td>SBB</td>
<td>b,t,r/m</td>
<td>SBB</td>
<td>b,i</td>
<td>SBB</td>
<td>w,i</td>
<td>PUSH</td>
<td>DS</td>
<td>POP</td>
<td>DS</td>
<td>SEG=</td>
<td>CS</td>
<td>DAS</td>
</tr>
<tr>
<td>SUB</td>
<td>b,f,r/m</td>
<td>w,f,r/m</td>
<td>SUB</td>
<td>b,t,r/m</td>
<td>SUB</td>
<td>w,t,r/m</td>
<td>SUB</td>
<td>b,i</td>
<td>SUB</td>
<td>w,i</td>
<td>SEG=</td>
<td>CS</td>
<td>AAS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CMP</td>
<td>b,f,r/m</td>
<td>w,f,r/m</td>
<td>CMP</td>
<td>b,t,r/m</td>
<td>CMP</td>
<td>w,t,r/m</td>
<td>CMP</td>
<td>b,i</td>
<td>CMP</td>
<td>w,i</td>
<td>SEG=</td>
<td>CS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEC</td>
<td>AX</td>
<td>CX</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POP</td>
<td>AX</td>
<td></td>
<td>POP</td>
<td></td>
<td>POP</td>
<td></td>
<td>POP</td>
<td></td>
<td>POP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>JS</td>
<td>JNS</td>
<td></td>
<td>JP/</td>
<td>JPE</td>
<td>JNP/</td>
<td>JPO</td>
<td>JLP/</td>
<td>JGE</td>
<td>JNL/</td>
<td>JGE</td>
<td>JLE/</td>
<td>JG</td>
<td>JNLE/</td>
<td>JG</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV</td>
<td>i AX</td>
<td></td>
<td>MOV</td>
<td>i CX</td>
<td>MOV</td>
<td>i DX</td>
<td>MOV</td>
<td>i SP</td>
<td>MOV</td>
<td>i BP</td>
<td>MOV</td>
<td>i SI</td>
<td>MOV</td>
<td>i DI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CBW</td>
<td>CWD</td>
<td></td>
<td>CALL</td>
<td>l,d</td>
<td>WAIT</td>
<td></td>
<td>PUSH</td>
<td></td>
<td>POPF</td>
<td></td>
<td>SAHF</td>
<td></td>
<td>LAHF</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TEST</td>
<td>b,i</td>
<td></td>
<td>STOS</td>
<td>b,i</td>
<td>STOS</td>
<td>w,i</td>
<td>LODS</td>
<td>b,i</td>
<td>LODS</td>
<td>w,i</td>
<td>SCAS</td>
<td>w,i</td>
<td>SCAS</td>
<td>w,i</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOV</td>
<td>i AX</td>
<td></td>
<td>MOV</td>
<td>i CX</td>
<td>MOV</td>
<td>i DX</td>
<td>MOV</td>
<td>i SP</td>
<td>MOV</td>
<td>i BP</td>
<td>MOV</td>
<td>i SI</td>
<td>MOV</td>
<td>i DI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RET</td>
<td></td>
<td></td>
<td>RET</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>IRET</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ESC</td>
<td>0</td>
<td></td>
<td>ESC</td>
<td>1</td>
<td>ESC</td>
<td>0,i</td>
<td>ESC</td>
<td>1,i</td>
<td>ESC</td>
<td>0,i</td>
<td>ESC</td>
<td>0,i</td>
<td>ESC</td>
<td>0,i</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CALL</td>
<td>d</td>
<td></td>
<td>JMP</td>
<td>d</td>
<td>JMP</td>
<td>l,d</td>
<td>IN</td>
<td>v,b</td>
<td>IN</td>
<td>v,w</td>
<td>OUT</td>
<td>v,b</td>
<td>OUT</td>
<td>v,w</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CLC</td>
<td></td>
<td></td>
<td>CLI</td>
<td></td>
<td>STI</td>
<td></td>
<td>CLD</td>
<td></td>
<td>STD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**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, CALL, JMP, JMP, PUSH, —

---

### 8088 Instruction Reference

B-17
<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Page</th>
<th>Mnemonic</th>
<th>Page</th>
<th>Mnemonic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAA</td>
<td>B-7</td>
<td>JG</td>
<td>B-13</td>
<td>MOV</td>
<td>B-5</td>
</tr>
<tr>
<td>AAD</td>
<td>B-9</td>
<td>JGE</td>
<td>B-12</td>
<td>MOVVS</td>
<td>B-10</td>
</tr>
<tr>
<td>AAM</td>
<td>B-8</td>
<td>JL</td>
<td>B-12</td>
<td>MUL</td>
<td>B-8</td>
</tr>
<tr>
<td>AAS</td>
<td>B-8</td>
<td>JLE</td>
<td>B-12</td>
<td>NEG</td>
<td>B-8</td>
</tr>
<tr>
<td>ADC</td>
<td>B-7</td>
<td>JMP</td>
<td>B-11</td>
<td>NOP</td>
<td>B-15</td>
</tr>
<tr>
<td>ADD</td>
<td>B-7</td>
<td>JNA</td>
<td>B-12</td>
<td>NOT</td>
<td>B-9</td>
</tr>
<tr>
<td>AND</td>
<td>B-9</td>
<td>JNAE</td>
<td>B-12</td>
<td>OR</td>
<td>B-10</td>
</tr>
<tr>
<td>CALL</td>
<td>B-11</td>
<td>JNB</td>
<td>B-13</td>
<td>OUT</td>
<td>B-6</td>
</tr>
<tr>
<td>CBW</td>
<td>B-9</td>
<td>JNBE</td>
<td>B-13</td>
<td>POP</td>
<td>B-5</td>
</tr>
<tr>
<td>CLC</td>
<td>B-15</td>
<td>JNE</td>
<td>B-12</td>
<td>POPF</td>
<td>B-6</td>
</tr>
<tr>
<td>CLD</td>
<td>B-15</td>
<td>JNG</td>
<td>B-12</td>
<td>PUSH</td>
<td>B-5</td>
</tr>
<tr>
<td>CLI</td>
<td>B-15</td>
<td>JNGE</td>
<td>B-12</td>
<td>PUSHF</td>
<td>B-6</td>
</tr>
<tr>
<td>CMC</td>
<td>B-15</td>
<td>JNL</td>
<td>B-12</td>
<td>RCL</td>
<td>B-9</td>
</tr>
<tr>
<td>CMP</td>
<td>B-8</td>
<td>JNLE</td>
<td>B-13</td>
<td>RCR</td>
<td>B-9</td>
</tr>
<tr>
<td>CMPS</td>
<td>B-10</td>
<td>JNO</td>
<td>B-13</td>
<td>REP</td>
<td>B-10</td>
</tr>
<tr>
<td>CWD</td>
<td>B-9</td>
<td>JNP</td>
<td>B-13</td>
<td>RET</td>
<td>B-12</td>
</tr>
<tr>
<td>DAA</td>
<td>B-7</td>
<td>JNS</td>
<td>B-13</td>
<td>ROL</td>
<td>B-9</td>
</tr>
<tr>
<td>DAS</td>
<td>B-8</td>
<td>JNZ</td>
<td>B-12</td>
<td>ROR</td>
<td>B-9</td>
</tr>
<tr>
<td>DEC</td>
<td>B-8</td>
<td>JO</td>
<td>B-12</td>
<td>SAHF</td>
<td>B-6</td>
</tr>
<tr>
<td>DIV</td>
<td>B-8</td>
<td>JP</td>
<td>B-12</td>
<td>SAL</td>
<td>B-9</td>
</tr>
<tr>
<td>ESC</td>
<td>B-15</td>
<td>JPE</td>
<td>B-12</td>
<td>SAR</td>
<td>B-9</td>
</tr>
<tr>
<td>HLT</td>
<td>B-15</td>
<td>JPO</td>
<td>B-13</td>
<td>SBB</td>
<td>B-8</td>
</tr>
<tr>
<td>IDIV</td>
<td>B-9</td>
<td>JS</td>
<td>B-12</td>
<td>SCAS</td>
<td>B-10</td>
</tr>
<tr>
<td>IMUL</td>
<td>B-8</td>
<td>JZ</td>
<td>B-12</td>
<td>SHL</td>
<td>B-9</td>
</tr>
<tr>
<td>IN</td>
<td>B-6</td>
<td>LAHF</td>
<td>B-6</td>
<td>SHR</td>
<td>B-9</td>
</tr>
<tr>
<td>INC</td>
<td>B-7</td>
<td>LDS</td>
<td>B-6</td>
<td>STC</td>
<td>B-15</td>
</tr>
<tr>
<td>INT</td>
<td>B-14</td>
<td>LEA</td>
<td>B-6</td>
<td>STD</td>
<td>B-15</td>
</tr>
<tr>
<td>INTO</td>
<td>B-14</td>
<td>LES</td>
<td>B-6</td>
<td>STI</td>
<td>B-15</td>
</tr>
<tr>
<td>IRET</td>
<td>B-14</td>
<td>LOCK</td>
<td>B-15</td>
<td>STOS</td>
<td>B-11</td>
</tr>
<tr>
<td>JA</td>
<td>B-13</td>
<td>LODS</td>
<td>B-11</td>
<td>SUB</td>
<td>B-7</td>
</tr>
<tr>
<td>JAE</td>
<td>B-13</td>
<td>LOOP</td>
<td>B-13</td>
<td>TEST</td>
<td>B-10</td>
</tr>
<tr>
<td>JB</td>
<td>B-12</td>
<td>LOOPE</td>
<td>B-13</td>
<td>WAIT</td>
<td>B-15</td>
</tr>
<tr>
<td>JBE</td>
<td>B-12</td>
<td>LOOPNE</td>
<td>B-13</td>
<td>XCHG</td>
<td>B-6</td>
</tr>
<tr>
<td>JCXZ</td>
<td>B-13</td>
<td>LOOPNZ</td>
<td>B-13</td>
<td>XLAT</td>
<td>B-6</td>
</tr>
<tr>
<td>JE</td>
<td>B-12</td>
<td>LOOPZ</td>
<td>B-13</td>
<td>XOR</td>
<td>B-10</td>
</tr>
</tbody>
</table>

B-18  8088 Instruction Reference
# APPENDIX C: OF CHARACTERS, KEYSTROKES, AND COLOR

<table>
<thead>
<tr>
<th>Hex</th>
<th>Dec</th>
<th>Symbol</th>
<th>Keystrokes</th>
<th>Modes</th>
<th>As Characters</th>
<th>Color/Graphics Monitor Adapter</th>
<th>IBM Monochrome Display Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0</td>
<td>Blank</td>
<td>Ctrl 2</td>
<td></td>
<td>Black</td>
<td>Black</td>
<td>Non-Display</td>
</tr>
<tr>
<td>01</td>
<td>1</td>
<td>😊</td>
<td>Ctrl A</td>
<td></td>
<td>Black</td>
<td>Blue</td>
<td>Underline</td>
</tr>
<tr>
<td>02</td>
<td>2</td>
<td>🐉</td>
<td>Ctrl B</td>
<td></td>
<td>Black</td>
<td>Green</td>
<td>Normal</td>
</tr>
<tr>
<td>03</td>
<td>3</td>
<td>❤️</td>
<td>Ctrl C</td>
<td></td>
<td>Black</td>
<td>Cyan</td>
<td>Normal</td>
</tr>
<tr>
<td>04</td>
<td>4</td>
<td>🌟</td>
<td>Ctrl D</td>
<td></td>
<td>Black</td>
<td>Red</td>
<td>Normal</td>
</tr>
<tr>
<td>05</td>
<td>5</td>
<td>♦️</td>
<td>Ctrl E</td>
<td></td>
<td>Black</td>
<td>Magenta</td>
<td>Normal</td>
</tr>
<tr>
<td>06</td>
<td>6</td>
<td>♠️</td>
<td>Ctrl F</td>
<td></td>
<td>Black</td>
<td>Brown</td>
<td>Normal</td>
</tr>
<tr>
<td>07</td>
<td>7</td>
<td>⚫️</td>
<td>Ctrl G</td>
<td></td>
<td>Black</td>
<td>Light Grey</td>
<td>Normal</td>
</tr>
<tr>
<td>08</td>
<td>8</td>
<td>🔒</td>
<td>Ctrl H, Backspace, Shift Backspace</td>
<td></td>
<td>Black</td>
<td>Dark Grey</td>
<td>Non-Display</td>
</tr>
<tr>
<td>09</td>
<td>9</td>
<td>🔒</td>
<td>Ctrl I</td>
<td></td>
<td>Black</td>
<td>Light Blue</td>
<td>High Intensity Underline</td>
</tr>
<tr>
<td>0A</td>
<td>10</td>
<td>🔒</td>
<td>Ctrl J, Ctrl L</td>
<td></td>
<td>Black</td>
<td>Light Green</td>
<td>High Intensity</td>
</tr>
<tr>
<td>0B</td>
<td>11</td>
<td>🔒</td>
<td>Ctrl K</td>
<td></td>
<td>Black</td>
<td>Light Green</td>
<td>High Intensity</td>
</tr>
<tr>
<td>0C</td>
<td>12</td>
<td>🔒</td>
<td>Ctrl L</td>
<td></td>
<td>Black</td>
<td>Light Red</td>
<td>High Intensity</td>
</tr>
<tr>
<td>0D</td>
<td>13</td>
<td>🔒</td>
<td>Ctrl M, Ctrl R</td>
<td></td>
<td>Black</td>
<td>Light Magenta</td>
<td>High Intensity</td>
</tr>
<tr>
<td>0E</td>
<td>14</td>
<td>🔒</td>
<td>Ctrl N</td>
<td></td>
<td>Black</td>
<td>Yellow</td>
<td>High Intensity</td>
</tr>
<tr>
<td>0F</td>
<td>15</td>
<td>🔒</td>
<td>Ctrl O</td>
<td></td>
<td>Black</td>
<td>White</td>
<td>High Intensity</td>
</tr>
<tr>
<td>10</td>
<td>16</td>
<td>🔒</td>
<td>Ctrl P</td>
<td></td>
<td>Blue</td>
<td>Black</td>
<td>Normal</td>
</tr>
<tr>
<td>11</td>
<td>17</td>
<td>🔒</td>
<td>Ctrl Q</td>
<td></td>
<td>Blue</td>
<td>Blue</td>
<td>Underline</td>
</tr>
<tr>
<td>12</td>
<td>18</td>
<td>🔒</td>
<td>Ctrl R</td>
<td></td>
<td>Blue</td>
<td>Green</td>
<td>Normal</td>
</tr>
<tr>
<td>13</td>
<td>19</td>
<td>🔒</td>
<td>Ctrl S</td>
<td></td>
<td>Blue</td>
<td>Cyan</td>
<td>Normal</td>
</tr>
<tr>
<td>14</td>
<td>20</td>
<td>🔒</td>
<td>Ctrl T</td>
<td></td>
<td>Blue</td>
<td>Red</td>
<td>Normal</td>
</tr>
<tr>
<td>15</td>
<td>21</td>
<td>🔒</td>
<td>Ctrl U</td>
<td></td>
<td></td>
<td>Magenta</td>
<td>Normal</td>
</tr>
<tr>
<td>16</td>
<td>22</td>
<td>🔒</td>
<td>Ctrl V</td>
<td></td>
<td>Blue</td>
<td>Brown</td>
<td>Normal</td>
</tr>
<tr>
<td>17</td>
<td>23</td>
<td>🔒</td>
<td>Ctrl W</td>
<td></td>
<td>Blue</td>
<td>Light Grey</td>
<td>Normal</td>
</tr>
</tbody>
</table>

Of Characters, Keystrokes, and Colors  C-1
<table>
<thead>
<tr>
<th><strong>Value</strong></th>
<th><strong>As Characters</strong></th>
<th><strong>IBM Monochrome Display Adapter</strong></th>
<th><strong>Color/Graphics Monitor Adapter</strong></th>
<th><strong>Foreground</strong></th>
<th><strong>Background</strong></th>
<th><strong>Modes</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex</td>
<td>Dec</td>
<td>Symbol</td>
<td>Keystrokes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>24</td>
<td>1</td>
<td>Ctrl X</td>
<td></td>
<td>Blue</td>
<td>Dark Grey</td>
</tr>
<tr>
<td>19</td>
<td>25</td>
<td>1</td>
<td>Ctrl Y</td>
<td></td>
<td>Blue</td>
<td>Light Blue</td>
</tr>
<tr>
<td>1A</td>
<td>26</td>
<td>-</td>
<td>Ctrl Z</td>
<td></td>
<td>Blue</td>
<td>Light Green</td>
</tr>
<tr>
<td>1B</td>
<td>27</td>
<td>-</td>
<td>Ctrl [, Esc, Shift Esc, Ctrl Esc</td>
<td></td>
<td>Blue</td>
<td>Light Cyan</td>
</tr>
</tbody>
</table>
| 1C | 28 | \
| | | | | Blue | Light Red | High Intensity |
| 1D | 29 | \| | | 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 | I | I | Shift | Green | Blue | Underline |
| 22 | 34 | " | " | Shift | Green | Green | Normal |
| 23 | 35 | # | # | Shift | Green | Cyan | Normal |
| 24 | 36 | $ | $ | Shift | Green | Red | Normal |
| 25 | 37 | % | % | Shift | Green | Magenta | Normal |
| 26 | 38 | & | & | Shift | Green | Brown | Normal |
| 27 | 39 | . | . | Green | Light Grey | Normal |
| 28 | 40 | ( | ( | Shift | Green | Dark Grey | High Intensity |
| 29 | 41 | ) | ) | Shift | Green | Light Blue | High Intensity Underline |
| 2A | 42 | " | " | Note 1 | Green | Light Green | High Intensity |
| 2B | 43 | + | + | Shift | Green | Light Cyan | High Intensity |
| 2C | 44 | . | . | Green | Light Red | High Intensity |
| 2D | 45 | . | . | Green | Light Magenta | High Intensity |
| 2E | 46 | . | . | Note 2 | Green | Yellow | High Intensity |

**C-2** Of Characters, Keystrokes, and Colors
<table>
<thead>
<tr>
<th>Hex</th>
<th>Dec</th>
<th>Symbol</th>
<th>Keystrokes</th>
<th>Modes</th>
<th>Background</th>
<th>Foreground</th>
<th>Color/Graphics Monitor Adapter</th>
<th>IBM Monochrome Display Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>2F</td>
<td>47</td>
<td>/</td>
<td>/</td>
<td></td>
<td>Green</td>
<td>White</td>
<td>High Intensity</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>48</td>
<td>0</td>
<td>0</td>
<td>Note 3</td>
<td>Cyan</td>
<td>Black</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>49</td>
<td>1</td>
<td>1</td>
<td>Note 3</td>
<td>Cyan</td>
<td>Blue</td>
<td>Underline</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>50</td>
<td>2</td>
<td>2</td>
<td>Note 3</td>
<td>Cyan</td>
<td>Green</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>51</td>
<td>3</td>
<td>3</td>
<td>Note 3</td>
<td>Cyan</td>
<td>Cyan</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>52</td>
<td>4</td>
<td>4</td>
<td>Note 3</td>
<td>Cyan</td>
<td>Red</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>53</td>
<td>5</td>
<td>5</td>
<td>Note 3</td>
<td>Cyan</td>
<td>Magenta</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>54</td>
<td>6</td>
<td>6</td>
<td>Note 3</td>
<td>Cyan</td>
<td>Brown</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>55</td>
<td>7</td>
<td>7</td>
<td>Note 3</td>
<td>Cyan</td>
<td>Light Grey</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>56</td>
<td>8</td>
<td>8</td>
<td>Note 3</td>
<td>Cyan</td>
<td>Dark Grey</td>
<td>High Intensity</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>57</td>
<td>9</td>
<td>9</td>
<td>Note 3</td>
<td>Cyan</td>
<td>Light Blue</td>
<td>High Intensity Underline</td>
<td></td>
</tr>
<tr>
<td>3A</td>
<td>58</td>
<td>:</td>
<td>:</td>
<td>Shift</td>
<td>Cyan</td>
<td>Light Green</td>
<td>High Intensity</td>
<td></td>
</tr>
<tr>
<td>3B</td>
<td>59</td>
<td>;</td>
<td>;</td>
<td>Shift</td>
<td>Cyan</td>
<td>Light Cyan</td>
<td>High Intensity</td>
<td></td>
</tr>
<tr>
<td>3C</td>
<td>60</td>
<td>&lt;</td>
<td>&lt;</td>
<td>Shift</td>
<td>Cyan</td>
<td>Light Red</td>
<td>High Intensity</td>
<td></td>
</tr>
<tr>
<td>3D</td>
<td>61</td>
<td>=</td>
<td>=</td>
<td></td>
<td>Cyan</td>
<td>Light Magenta</td>
<td>High Intensity</td>
<td></td>
</tr>
<tr>
<td>3E</td>
<td>62</td>
<td>&gt;</td>
<td>&gt;</td>
<td>Shift</td>
<td>Cyan</td>
<td>Yellow</td>
<td>High Intensity</td>
<td></td>
</tr>
<tr>
<td>3F</td>
<td>63</td>
<td>?</td>
<td>?</td>
<td>Shift</td>
<td>Cyan</td>
<td>White</td>
<td>High Intensity</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>64</td>
<td>@</td>
<td>@</td>
<td>Shift</td>
<td>Red</td>
<td>Black</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>65</td>
<td>A</td>
<td>A</td>
<td>Note 4</td>
<td>Red</td>
<td>Blue</td>
<td>Underline</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>66</td>
<td>B</td>
<td>B</td>
<td>Note 4</td>
<td>Red</td>
<td>Green</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>67</td>
<td>C</td>
<td>C</td>
<td>Note 4</td>
<td>Red</td>
<td>Cyan</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>68</td>
<td>D</td>
<td>D</td>
<td>Note 4</td>
<td>Red</td>
<td>Red</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>69</td>
<td>E</td>
<td>E</td>
<td>Note 4</td>
<td>Red</td>
<td>Magenta</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>70</td>
<td>F</td>
<td>F</td>
<td>Note 4</td>
<td>Red</td>
<td>Brown</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>71</td>
<td>G</td>
<td>G</td>
<td>Note 4</td>
<td>Red</td>
<td>Light Grey</td>
<td>Normal</td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>72</td>
<td>H</td>
<td>H</td>
<td>Note 4</td>
<td>Red</td>
<td>Dark Grey</td>
<td>High Intensity</td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>73</td>
<td>I</td>
<td>I</td>
<td>Note 4</td>
<td>Red</td>
<td>Light Blue</td>
<td>High Intensity Underline</td>
<td></td>
</tr>
<tr>
<td>4A</td>
<td>74</td>
<td>J</td>
<td>J</td>
<td>Note 4</td>
<td>Red</td>
<td>Light Green</td>
<td>High Intensity</td>
<td></td>
</tr>
</tbody>
</table>

Of Characters, Keystrokes, and Colors  C-3
<table>
<thead>
<tr>
<th>Value</th>
<th>As Characters</th>
<th>As Text Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Color/Graphics</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Monitor Adapter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IBM Monochrome</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Display Adapter</td>
</tr>
<tr>
<td>Hex</td>
<td>Dec</td>
<td>Background</td>
</tr>
<tr>
<td>Value</td>
<td>Symbol</td>
<td>Foreground</td>
</tr>
<tr>
<td>Value</td>
<td>Keystrokes</td>
<td>Modes</td>
</tr>
<tr>
<td>Value</td>
<td>Hex</td>
<td>Value</td>
</tr>
<tr>
<td>4B 75</td>
<td>K</td>
<td>K</td>
</tr>
<tr>
<td>4C 76</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td>4D 77</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>4E 78</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>4F 79</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>50 80</td>
<td>P</td>
<td>P</td>
</tr>
<tr>
<td>51 81</td>
<td>Q</td>
<td>Q</td>
</tr>
<tr>
<td>52 82</td>
<td>R</td>
<td>R</td>
</tr>
<tr>
<td>53 83</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>54 84</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>55 85</td>
<td>U</td>
<td>U</td>
</tr>
<tr>
<td>56 86</td>
<td>V</td>
<td>V</td>
</tr>
<tr>
<td>57 87</td>
<td>W</td>
<td>W</td>
</tr>
<tr>
<td>58 88</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>59 89</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>5A 90</td>
<td>Z</td>
<td>Z</td>
</tr>
<tr>
<td>5B 91</td>
<td>[</td>
<td>[</td>
</tr>
<tr>
<td>5C 92</td>
<td>\</td>
<td>\</td>
</tr>
<tr>
<td>5D 93</td>
<td>]</td>
<td>]</td>
</tr>
<tr>
<td>5E 94</td>
<td>^</td>
<td>^</td>
</tr>
<tr>
<td>5F 95</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>60 96</td>
<td>`</td>
<td>`</td>
</tr>
<tr>
<td>61 97</td>
<td>a</td>
<td>a</td>
</tr>
<tr>
<td>62 98</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>63 99</td>
<td>c</td>
<td>c</td>
</tr>
<tr>
<td>64 100</td>
<td>d</td>
<td>d</td>
</tr>
<tr>
<td>65 101</td>
<td>e</td>
<td>e</td>
</tr>
<tr>
<td>66 102</td>
<td>f</td>
<td>f</td>
</tr>
</tbody>
</table>

C-4 Of Characters, Keystrokes, and Colors
<table>
<thead>
<tr>
<th>Value</th>
<th>As Characters</th>
<th>As Text Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Color/Graphics Monitor Adapter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Background</td>
</tr>
<tr>
<td>67</td>
<td>103</td>
<td>g</td>
</tr>
<tr>
<td>68</td>
<td>104</td>
<td>h</td>
</tr>
<tr>
<td>69</td>
<td>105</td>
<td>i</td>
</tr>
<tr>
<td>6A</td>
<td>106</td>
<td>j</td>
</tr>
<tr>
<td>6B</td>
<td>107</td>
<td>k</td>
</tr>
<tr>
<td>6C</td>
<td>108</td>
<td>l</td>
</tr>
<tr>
<td>6D</td>
<td>109</td>
<td>m</td>
</tr>
<tr>
<td>6E</td>
<td>110</td>
<td>n</td>
</tr>
<tr>
<td>6F</td>
<td>111</td>
<td>o</td>
</tr>
<tr>
<td>70</td>
<td>112</td>
<td>p</td>
</tr>
<tr>
<td>71</td>
<td>113</td>
<td>q</td>
</tr>
<tr>
<td>72</td>
<td>114</td>
<td>r</td>
</tr>
<tr>
<td>73</td>
<td>115</td>
<td>s</td>
</tr>
<tr>
<td>74</td>
<td>116</td>
<td>f</td>
</tr>
<tr>
<td>75</td>
<td>117</td>
<td>u</td>
</tr>
<tr>
<td>76</td>
<td>118</td>
<td>v</td>
</tr>
<tr>
<td>77</td>
<td>119</td>
<td>w</td>
</tr>
<tr>
<td>78</td>
<td>120</td>
<td>x</td>
</tr>
<tr>
<td>79</td>
<td>121</td>
<td>y</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7A</td>
<td>122</td>
<td>z</td>
</tr>
<tr>
<td>7B</td>
<td>123</td>
<td></td>
</tr>
<tr>
<td>7C</td>
<td>124</td>
<td></td>
</tr>
<tr>
<td>7D</td>
<td>125</td>
<td></td>
</tr>
<tr>
<td>7E</td>
<td>126</td>
<td></td>
</tr>
<tr>
<td>7F</td>
<td>127</td>
<td>Δ</td>
</tr>
</tbody>
</table>

Of Characters, Keystrokes, and Colors  C-5
<table>
<thead>
<tr>
<th>Value</th>
<th>As Characters</th>
<th>As Text Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hex</strong></td>
<td><strong>Dec</strong></td>
<td><strong>Symbol</strong></td>
</tr>
<tr>
<td>80</td>
<td>128</td>
<td>Ç</td>
</tr>
<tr>
<td>81</td>
<td>129</td>
<td>ü</td>
</tr>
<tr>
<td>82</td>
<td>130</td>
<td>é</td>
</tr>
<tr>
<td>83</td>
<td>131</td>
<td>â</td>
</tr>
<tr>
<td>84</td>
<td>132</td>
<td>à</td>
</tr>
<tr>
<td>85</td>
<td>133</td>
<td>à</td>
</tr>
<tr>
<td>86</td>
<td>134</td>
<td>à</td>
</tr>
<tr>
<td>87</td>
<td>135</td>
<td>ç</td>
</tr>
<tr>
<td>88</td>
<td>136</td>
<td>è</td>
</tr>
<tr>
<td>89</td>
<td>137</td>
<td>é</td>
</tr>
<tr>
<td>8A</td>
<td>138</td>
<td>è</td>
</tr>
<tr>
<td>8B</td>
<td>139</td>
<td>í</td>
</tr>
<tr>
<td>8C</td>
<td>140</td>
<td>í</td>
</tr>
<tr>
<td>8D</td>
<td>141</td>
<td>í</td>
</tr>
<tr>
<td>8E</td>
<td>142</td>
<td>Á</td>
</tr>
<tr>
<td>8F</td>
<td>143</td>
<td>Á</td>
</tr>
<tr>
<td>90</td>
<td>144</td>
<td>É</td>
</tr>
<tr>
<td>91</td>
<td>145</td>
<td>æ</td>
</tr>
<tr>
<td>92</td>
<td>146</td>
<td>Æ</td>
</tr>
<tr>
<td>93</td>
<td>147</td>
<td>ô</td>
</tr>
<tr>
<td>94</td>
<td>148</td>
<td>ö</td>
</tr>
<tr>
<td>95</td>
<td>149</td>
<td>ò</td>
</tr>
<tr>
<td>96</td>
<td>150</td>
<td>ü</td>
</tr>
<tr>
<td>97</td>
<td>151</td>
<td>ū</td>
</tr>
<tr>
<td>98</td>
<td>152</td>
<td>ý</td>
</tr>
<tr>
<td>99</td>
<td>153</td>
<td>ö</td>
</tr>
<tr>
<td>9A</td>
<td>154</td>
<td>ū</td>
</tr>
</tbody>
</table>

C-6  Of Characters, Keystrokes, and Colors
<table>
<thead>
<tr>
<th>Value</th>
<th>As Characters</th>
<th>As Text Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Color/Graphics Monitor Adapter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Background</td>
</tr>
<tr>
<td>9B 155</td>
<td>€ Alt 155</td>
<td>Note 6</td>
</tr>
<tr>
<td>9C 156</td>
<td>£ Alt 156</td>
<td>Note 6</td>
</tr>
<tr>
<td>9D 157</td>
<td>¥ Alt 157</td>
<td>Note 6</td>
</tr>
<tr>
<td>9E 158</td>
<td>Pt Alt 158</td>
<td>Note 6</td>
</tr>
<tr>
<td>9F 159</td>
<td>⌃ Alt 159</td>
<td>Note 6</td>
</tr>
<tr>
<td>A0 160</td>
<td>á Alt 160</td>
<td>Note 6</td>
</tr>
<tr>
<td>A1 161</td>
<td>í Alt 161</td>
<td>Note 6</td>
</tr>
<tr>
<td>A2 162</td>
<td>ó Alt 162</td>
<td>Note 6</td>
</tr>
<tr>
<td>A3 163</td>
<td>ù Alt 163</td>
<td>Note 6</td>
</tr>
<tr>
<td>A4 164</td>
<td>ñ Alt 164</td>
<td>Note 6</td>
</tr>
<tr>
<td>A5 165</td>
<td>Ñ Alt 165</td>
<td>Note 6</td>
</tr>
<tr>
<td>A6 166</td>
<td>ñ Alt 166</td>
<td>Note 6</td>
</tr>
<tr>
<td>A7 167</td>
<td>ò Alt 167</td>
<td>Note 6</td>
</tr>
<tr>
<td>A8 168</td>
<td>ç Alt 168</td>
<td>Note 6</td>
</tr>
<tr>
<td>A9 169</td>
<td>ç Alt 169</td>
<td>Note 6</td>
</tr>
<tr>
<td>AA 170</td>
<td>ç Alt 170</td>
<td>Note 6</td>
</tr>
<tr>
<td>AB 171</td>
<td>¼ Alt 171</td>
<td>Note 6</td>
</tr>
<tr>
<td>AC 172</td>
<td>¼ Alt 172</td>
<td>Note 6</td>
</tr>
<tr>
<td>AD 173</td>
<td>÷ Alt 173</td>
<td>Note 6</td>
</tr>
<tr>
<td>AE 174</td>
<td>&lt;&lt; Alt 174</td>
<td>Note 6</td>
</tr>
<tr>
<td>AF 175</td>
<td>&gt;&gt; Alt 175</td>
<td>Note 6</td>
</tr>
<tr>
<td>B0 176</td>
<td>Alt 176</td>
<td>Note 6</td>
</tr>
<tr>
<td>B1 177</td>
<td>Alt 177</td>
<td>Note 6</td>
</tr>
<tr>
<td>B2 178</td>
<td>Alt 178</td>
<td>Note 6</td>
</tr>
<tr>
<td>B3 179</td>
<td>Alt 179</td>
<td>Note 6</td>
</tr>
<tr>
<td>B4 180</td>
<td>Alt 180</td>
<td>Note 6</td>
</tr>
<tr>
<td>B5 181</td>
<td>Alt 181</td>
<td>Note 6</td>
</tr>
<tr>
<td>B6 182</td>
<td>Alt 182</td>
<td>Note 6</td>
</tr>
<tr>
<td>Value</td>
<td>As Characters</td>
<td>As Text Attributes</td>
</tr>
<tr>
<td>-------</td>
<td>--------------</td>
<td>--------------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Color/Graphics Monitor Adapter</td>
</tr>
<tr>
<td>183</td>
<td>Alt 183</td>
<td>Note 6  Cyan Light Grey</td>
</tr>
<tr>
<td>184</td>
<td>Alt 184</td>
<td>Note 6  Cyan Dark Grey</td>
</tr>
<tr>
<td>185</td>
<td>Alt 185</td>
<td>Note 6  Cyan Light Blue</td>
</tr>
<tr>
<td>186</td>
<td>Alt 186</td>
<td>Note 6  Cyan Light Green</td>
</tr>
<tr>
<td>187</td>
<td>Alt 187</td>
<td>Note 6  Cyan Light Cyan</td>
</tr>
<tr>
<td>188</td>
<td>Alt 188</td>
<td>Note 6  Cyan Light Red</td>
</tr>
<tr>
<td>189</td>
<td>Alt 189</td>
<td>Note 6  Cyan Light Magenta</td>
</tr>
<tr>
<td>190</td>
<td>Alt 190</td>
<td>Note 6  Cyan Yellow</td>
</tr>
<tr>
<td>191</td>
<td>Alt 191</td>
<td>Note 6  Cyan White</td>
</tr>
<tr>
<td>192</td>
<td>Alt 192</td>
<td>Note 6  Red Black</td>
</tr>
<tr>
<td>193</td>
<td>Alt 193</td>
<td>Note 6  Red Blue</td>
</tr>
<tr>
<td>194</td>
<td>Alt 194</td>
<td>Note 6  Red Green</td>
</tr>
<tr>
<td>195</td>
<td>Alt 195</td>
<td>Note 6  Red Cyan</td>
</tr>
<tr>
<td>196</td>
<td>Alt 196</td>
<td>Note 6  Red Red</td>
</tr>
<tr>
<td>197</td>
<td>Alt 197</td>
<td>Note 6  Red Magenta</td>
</tr>
<tr>
<td>198</td>
<td>Alt 198</td>
<td>Note 6  Red Brown</td>
</tr>
<tr>
<td>199</td>
<td>Alt 199</td>
<td>Note 6  Red Light Grey</td>
</tr>
<tr>
<td>200</td>
<td>Alt 200</td>
<td>Note 6  Red Dark Grey</td>
</tr>
<tr>
<td>201</td>
<td>Alt 201</td>
<td>Note 6  Red Light Blue</td>
</tr>
<tr>
<td>202</td>
<td>Alt 202</td>
<td>Note 6  Red Light Green</td>
</tr>
<tr>
<td>203</td>
<td>Alt 203</td>
<td>Note 6  Red Light Cyan</td>
</tr>
<tr>
<td>204</td>
<td>Alt 204</td>
<td>Note 6  Red Light Red</td>
</tr>
<tr>
<td>205</td>
<td>Alt 205</td>
<td>Note 6  Red Light Magenta</td>
</tr>
<tr>
<td>206</td>
<td>Alt 206</td>
<td>Note 6  Red Yellow</td>
</tr>
<tr>
<td>207</td>
<td>Alt 207</td>
<td>Note 6  Red White</td>
</tr>
<tr>
<td>208</td>
<td>Alt 208</td>
<td>Note 6  Magenta Black</td>
</tr>
</tbody>
</table>

C-8  Of Characters, Keystrokes, and Colors
<table>
<thead>
<tr>
<th>Value</th>
<th>As Characters</th>
<th>As Text Attributes</th>
<th>Color/Graphics Monitor Adapter</th>
<th>IBM Monochrome Display Adapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hex</td>
<td>Symbol</td>
<td>Key strokes</td>
<td>Background</td>
<td>Foreground</td>
</tr>
<tr>
<td>D1</td>
<td>209</td>
<td>Alt 209</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>D2</td>
<td>210</td>
<td>Alt 210</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>D3</td>
<td>211</td>
<td>Alt 211</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>D4</td>
<td>212</td>
<td>Alt 212</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>D5</td>
<td>213</td>
<td>Alt 213</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>D6</td>
<td>214</td>
<td>Alt 214</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>D7</td>
<td>215</td>
<td>Alt 215</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>D8</td>
<td>216</td>
<td>Alt 216</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>D9</td>
<td>217</td>
<td>Alt 217</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>DA</td>
<td>218</td>
<td>Alt 218</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>DB</td>
<td>219</td>
<td>Alt 219</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>DC</td>
<td>220</td>
<td>Alt 220</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>DD</td>
<td>221</td>
<td>Alt 221</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>DE</td>
<td>222</td>
<td>Alt 222</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>DF</td>
<td>223</td>
<td>Alt 223</td>
<td>Note 6</td>
<td>Magenta</td>
</tr>
<tr>
<td>E0</td>
<td>224</td>
<td>α</td>
<td>Note 6</td>
<td>Yellow</td>
</tr>
<tr>
<td>E1</td>
<td>225</td>
<td>β</td>
<td>Note 6</td>
<td>Yellow</td>
</tr>
<tr>
<td>E2</td>
<td>226</td>
<td>Γ</td>
<td>Note 6</td>
<td>Yellow</td>
</tr>
<tr>
<td>E3</td>
<td>227</td>
<td>π</td>
<td>Note 6</td>
<td>Yellow</td>
</tr>
<tr>
<td>E4</td>
<td>228</td>
<td>ζ</td>
<td>Note 6</td>
<td>Yellow</td>
</tr>
<tr>
<td>E5</td>
<td>229</td>
<td>σ</td>
<td>Note 6</td>
<td>Yellow</td>
</tr>
<tr>
<td>E6</td>
<td>230</td>
<td>μ</td>
<td>Note 6</td>
<td>Yellow</td>
</tr>
<tr>
<td>E7</td>
<td>231</td>
<td>τ</td>
<td>Note 6</td>
<td>Yellow</td>
</tr>
<tr>
<td>E8</td>
<td>232</td>
<td>Φ</td>
<td>Note 6</td>
<td>Yellow</td>
</tr>
<tr>
<td>E9</td>
<td>233</td>
<td>θ</td>
<td>Note 6</td>
<td>Yellow</td>
</tr>
<tr>
<td>EA</td>
<td>234</td>
<td>Ω</td>
<td>Note 6</td>
<td>Yellow</td>
</tr>
<tr>
<td>EB</td>
<td>235</td>
<td>δ</td>
<td>Note 6</td>
<td>Yellow</td>
</tr>
</tbody>
</table>

Of Characters, Keystrokes, and Colors  C-9
<table>
<thead>
<tr>
<th>Value</th>
<th>As Characters</th>
<th>As Text Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Color/Graphics</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Monitor Adapter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Background</td>
</tr>
<tr>
<td>EC 236</td>
<td>∞</td>
<td>Alt 236</td>
</tr>
<tr>
<td>ED 237</td>
<td>φ</td>
<td>Alt 237</td>
</tr>
<tr>
<td>EE 238</td>
<td>ε</td>
<td>Alt 238</td>
</tr>
<tr>
<td>EF 239</td>
<td>∩</td>
<td>Alt 239</td>
</tr>
<tr>
<td>F0 240</td>
<td>≡</td>
<td>Alt 240</td>
</tr>
<tr>
<td>F1 241</td>
<td>±</td>
<td>Alt 241</td>
</tr>
<tr>
<td>F2 242</td>
<td>≥</td>
<td>Alt 242</td>
</tr>
<tr>
<td>F3 243</td>
<td>≤</td>
<td>Alt 243</td>
</tr>
<tr>
<td>F4 244</td>
<td>△</td>
<td>Alt 244</td>
</tr>
<tr>
<td>F5 245</td>
<td>⊖</td>
<td>Alt 245</td>
</tr>
<tr>
<td>F6 246</td>
<td>⊔</td>
<td>Alt 246</td>
</tr>
<tr>
<td>F7 247</td>
<td>≈</td>
<td>Alt 247</td>
</tr>
<tr>
<td>F8 248</td>
<td>O</td>
<td>Alt 248</td>
</tr>
<tr>
<td>F9 249</td>
<td>•</td>
<td>Alt 249</td>
</tr>
<tr>
<td>FA 250</td>
<td>◊</td>
<td>Alt 250</td>
</tr>
<tr>
<td>FB 251</td>
<td>√</td>
<td>Alt 251</td>
</tr>
<tr>
<td>FC 252</td>
<td>η</td>
<td>Alt 252</td>
</tr>
<tr>
<td>FD 253</td>
<td>2</td>
<td>Alt 253</td>
</tr>
<tr>
<td>FE 254</td>
<td>▌</td>
<td>Alt 254</td>
</tr>
<tr>
<td>FF 255</td>
<td>BLANK</td>
<td>Alt 255</td>
</tr>
</tbody>
</table>

C-10 Of Characters, Keystrokes, and Colors
NOTE 1 Asterisk (*) can easily be keyed using two methods: 1) hit the Print Screen key or 2) in shift mode hit the 8 key.

NOTE 2 Period (.) can easily be keyed using two methods: 1) hit the 9 key or 2) in shift or Num Lock mode hit the Delete 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.)
<table>
<thead>
<tr>
<th>DECIMAL VALUE</th>
<th>0</th>
<th>16</th>
<th>32</th>
<th>48</th>
<th>64</th>
<th>80</th>
<th>96</th>
<th>112</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEXA</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>DECIMAL VALUE</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>😊</td>
<td>&lt;</td>
<td>!</td>
<td>1</td>
<td>A</td>
<td>Q</td>
<td>a q</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>😊</td>
<td>↑</td>
<td>&quot;</td>
<td>2</td>
<td>B</td>
<td>R</td>
<td>b r</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>⚖</td>
<td>!!</td>
<td>#</td>
<td>3</td>
<td>C</td>
<td>S</td>
<td>s s</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>♦</td>
<td>♣</td>
<td>$</td>
<td>4</td>
<td>D</td>
<td>T</td>
<td>d t</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>♣</td>
<td>$</td>
<td>%</td>
<td>5</td>
<td>E</td>
<td>U</td>
<td>e u</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>♦</td>
<td>&amp;</td>
<td>6</td>
<td>F</td>
<td>V</td>
<td>f</td>
<td>v</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>⬆</td>
<td>'</td>
<td>7</td>
<td>G</td>
<td>W</td>
<td>g</td>
<td>w</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>⬇</td>
<td>(</td>
<td>8</td>
<td>H</td>
<td>X</td>
<td>h</td>
<td>x</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>⬇</td>
<td>)</td>
<td>9</td>
<td>I</td>
<td>Y</td>
<td>i</td>
<td>y</td>
</tr>
<tr>
<td>10</td>
<td>A</td>
<td>⬇</td>
<td>*</td>
<td>J</td>
<td>Z</td>
<td>j</td>
<td>z</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>B</td>
<td>⬇</td>
<td>+</td>
<td>;</td>
<td>K</td>
<td>l</td>
<td>{</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>C</td>
<td>⬇</td>
<td>,</td>
<td>&lt;</td>
<td>L</td>
<td>\</td>
<td>l</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>D</td>
<td>⬇</td>
<td>=</td>
<td>M</td>
<td>l</td>
<td>}</td>
<td>m</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>E</td>
<td>⬇</td>
<td>&gt;</td>
<td>N</td>
<td>^</td>
<td>n</td>
<td>~</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>F</td>
<td>⬇</td>
<td>/</td>
<td>?</td>
<td>O</td>
<td>_</td>
<td>o</td>
<td>△</td>
</tr>
</tbody>
</table>

C-12 Of Characters, Keystrokes, and Colors
<table>
<thead>
<tr>
<th>DECIMAL VALUE</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>HEXA DECIMAL VALUE</td>
<td>8</td>
<td>9</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>D</td>
<td>E</td>
<td>F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>Ç</td>
<td>É</td>
<td>á</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>144</td>
<td>ü</td>
<td>æ</td>
<td>í</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>160</td>
<td>é</td>
<td>Ä</td>
<td>ò</td>
<td>õ</td>
<td>ú</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>176</td>
<td>å</td>
<td>ã</td>
<td>ò</td>
<td>õ</td>
<td>ñ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>192</td>
<td>à</td>
<td>ò</td>
<td>û</td>
<td>õ</td>
<td>ñ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>208</td>
<td>å</td>
<td>ã</td>
<td>û</td>
<td>õ</td>
<td>ñ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>224</td>
<td>à</td>
<td>ò</td>
<td>û</td>
<td>õ</td>
<td>ñ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>240</td>
<td>å</td>
<td>ã</td>
<td>û</td>
<td>õ</td>
<td>ñ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Of Characters, Keystrokes, and Colors C-13
APPENDIX D: LOGIC DIAGRAMS

System Board .................................................. D-2
Keyboard – Type 1 ............................................ D-12
Keyboard – Type 2 ............................................ D-14
Expansion Board .............................................. D-15
Extender Card ................................................ D-16
Receiver Card ................................................ D-19
Printer .......................................................... D-22
Printer Adapter ............................................... D-25
Monochrome Display Adapter ............................... D-26
Color/Graphics Monitor Adapter ............................ D-36
Color Display .................................................. D-42
Monochrome Display .......................................... D-44
5-1/4 Inch Diskette Drive Adapter .......................... D-45
5-1/4 Inch Diskette Drive – Type 1 ......................... D-49
5-1/4 Inch Diskette Drive – Type 2 ......................... D-52
Fixed Disk Drive Adapter ................................... D-54
Fixed Disk Drive – Type 1 .................................. D-60
Fixed Disk Drive – Type 2 .................................. D-63
32K Memory Expansion Option ............................. D-66
64K Memory Expansion Option ............................. D-69
64/256K Memory Expansion Option ......................... D-72
Game Control Adapter ....................................... D-76
Prototype Card ............................................... D-77
Asynchronous Communications Adapter .................. D-78
Binary Synchronous Communications Adapter .......... D-79
SDLC Communications Adapter ............................. D-81
System Board (Sheet 1 of 10)

D-2 Logic Diagrams
D-4 Logic Diagrams
System Board (Sheet 7 of 10)

D-8  Logic Diagrams
D-14 Logic Diagrams
Extender Card (Sheet 3 of 3)
D-20 Logic Diagrams
Printer Adapter (Sheet 1 of 1)

Logic Diagrams  D-25
D-40 Logic Diagrams
DANGER
HAZARDOUS VOLTAGES
UP TO 450 VOLTS EXIST
ON THE PRINTED
CIRCUIT BOARDS

Color Display (Sheet 1 of 1)

D-42 Logic Diagrams
DANGER
HAZARDOUS VOLTAGES
UP TO 450 VOLTS EXIST
ON THE PRINTED
CIRCUIT BOARDS

Logic Diagrams  D-43
Monochrome Display (Sheet 1 of 1)
5-1/4 Inch Diskette Drive Adapter (Sheet 1 of 4)
5 1/4 Inch Diskette Drive Adapter (Sheet 4 of 4)

D-48 Logic Diagrams
5-1/4 Inch Diskette Drive - Type 2 (Sheet 2 of 2)
32K Memory Expansion Option (Sheet 1 of 3)
Asynchronous Communications Adapter (Sheet 1 of 1)
SDLC Communications Adapter (Sheet 2 of 2)
APPENDIX E: SPECIFICATIONS

System Unit

Size:
  Length—19.6 in (500 mm)
  Depth—16.1 in (410 mm)
  Height—5.5 in (142 mm)

Weight:
  32 lb (14.5 kb)

Power Cables:
  Length—6 ft (1.83 m)
  Size—18 AWG

Environment:
  Air Temperature
    System ON, 60° to 90° F (15.6° to 32.2° C)
    System OFF, 50° to 110° F (10° to 43° C)

  Humidity
    System ON, 8% to 80%
    System OFF, 20% to 80%

Heat Output:
  717 BTU/hr

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

Electrical:
  Nominal—120 Vac
  Minimum—104 Vac
  Maximum—127 Vac

Keyboard

Size:
  Length—19.6 in (500 mm)
  Depth—7.87 in (200 mm)
  Height—2.2 in (57 mm)

Weight:
  6.5 lb (2.9 kg)
Color Display

Size:
  Length—15.4 in (392 mm)
  Depth—15.6 in (407 mm)
  Height—11.7 in (297 mm)

Weight:
  26 lb (11.8 kg)

Heat Output:
  240 BTU/hr

Power Cables:
  Length—6 ft (1.83 m)
  Size—18 AWG

Signal Cable:
  Length—5 ft (1.5 m)
  Size—22 AWG

Expansion Unit

Size:
  Length—19.6 in (500 mm)
  Depth—16.1 in (410 mm)
  Height—5.5 in (142 mm)

Weight:
  33 lb (14.9 kg)

Power Cables:
  Length—6 ft (1.83 m)
  Size—18 AWG

Signal Cable:
  Length—3.28 ft (1 m)
  Size—22 AWG

Environment:
  Air Temperature
    System ON, 60° to 90° F (15.6° to 32.2° C)
    System OFF, 50° to 110° F (10° to 43° C)

  Humidity
    System ON, 8% to 80%
    System OFF, 20% to 80%

Heat Output:
  717 BTU/hr

Electrical:
  Nominal—120 Vac
  Minimum—104 Vac
  Maximum—127 Vac

E-2 Specifications
Monochrome Display

Size:
   Length--14.9 in (380 mm)
   Depth--13.7 in (350 mm)
   Height--11 in (280 mm)

Weight:
   17.3 lb (7.9 kg)

Heat Output:
   325 BTU/hr

Power Cable:
   Length--3 ft (.914 m)
   Size--18 AWG

Signal Cable:
   Length--4 ft (1.22 m)
   Size--22 AWG

80 CPS Printers

Size:
   Length--15.7 in (400 mm)
   Depth--14.5 in (370 mm)
   Height--4.3 in (110 mm)

Weight:
   12.9 lb (5.9 kg)

Power Cable:
   Length--6 ft (1.83 mm)
   Size--18 AWG

Signal Cable:
   Length--6 ft (1.83 m)
   Size--22 AWG

Heat Output:
   341 BTU/hr (maximum)

Electrical:
   Nominal--120 Vac
   Minimum--104 Vac
   Maximum--127 Vac
Notes:
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.
APPENDIX F: COMMUNICATIONS

Information processing equipment used for communications is called data terminal equipment (DTE). Equipment used to connect the DTE to the communications line is called data communications equipment (DCE).

An adapter is used to connect the data terminal equipment to the data communications line as shown in the following illustration:

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

EIA standards are labeled RS-x (Recommended Standards-x) and CCITT standards are labeled V.x or X.x, where x is the number of the standard.

The EIA RS-232 interface standard defines the connector type, pin numbers, line names, and signal levels used to connect data terminal equipment to data communications equipment for the purpose of transmitting and receiving data. Since the RS-232 standard was developed, it has been revised three times. The three revised standards are the RS-232A, the RS-232B, and the presently used RS-232C.

The CCITT V.24 interface standard is equivalent to the RS-232C standard; therefore, the descriptions of the EIA standards also apply to the CCITT standards.
The following is an illustration of data terminal equipment connected to an external modem using connections defined by the RS-232C interface standard:

![Diagram of RS-232C connections](image)

**EIA/CCITT Line Number**

1. Protective Ground
2. Signal Ground
3. Transmitted Data
4. Received Data
5. Request to Send
6. Clear to Send
7. Data Set Ready
8. Data Terminal Ready
9. Connect Data Set to Line
10. Received Line Signal Detector
11. Speed Select
12. Transmit Signal Element Timing
13. Receive Signal Element Timing
14. Select Standby
15. Ring Indicator
16. Test
17. **Not used when business machine clocking is used.**
18. **Not standardized by EIA (Electronics Industry Association).**
19. **Not standardized by CCITT**

**External Modem Cable Connector**

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
```

**Telephone Co. Lead Number**

1. AA/101
2. AB/102
3. BA/103
4. BB/104
5. CA/105
6. CB/106
7. CC/107
8. CD/108.2
9. **/108.1
10. CF/109
11. CH/111
12. DB/114
13. DD/115
14. **/116
15. DE/125
16. **/**

**F-2 Communications**
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.

<table>
<thead>
<tr>
<th>Switched Timing Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Terminal Ready</td>
</tr>
<tr>
<td>Data Set Ready</td>
</tr>
<tr>
<td>Request to Send</td>
</tr>
<tr>
<td>Clear to Send</td>
</tr>
<tr>
<td>Transmitted Data</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Nonswitched Timing Sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Terminal Ready</td>
</tr>
<tr>
<td>Data Terminal Ready</td>
</tr>
<tr>
<td>Request to Send</td>
</tr>
<tr>
<td>Clear to Send</td>
</tr>
<tr>
<td>Transmitted Data</td>
</tr>
</tbody>
</table>

The following examples show how a link is established on a nonswitched point-to-point line, a nonswitched multipoint line, and a switched point-to-point line.
Establishing a Link on a Nonswitched Point-to-Point Line

1. The terminals at both locations activate the 'data terminal ready' lines. 1 and 8.

2. Normally the 'data set ready' lines 2 and 9 from the modems are active whenever the modems are powered on.

3. Terminal A activates the 'request to send' line 3, which causes the modem at terminal A to generate a carrier signal.

4. Modem B detects the carrier, and activates the 'received line signal detector' line (sometimes called data carrier detect) 10. Modem B also activates the 'receiver signal element timing' line (sometimes called receive clock) 11 to send receive clock signals to the terminal. Some modems activate the clock signals whenever the modem is powered on.

5. After a specified delay, modem A activates the 'clear to send' line 4, which indicates to terminal A that the modem is ready to transmit data.

6. Terminal A serializes the data to be transmitted (through the serdes) and transmits the data one bit at a time (synchronized by the transmit clock) onto the 'transmitted data' line 5 to the modem.

7. The modem modulates the carrier signal with the data and transmits it to the modem B 15.

8. Modem B demodulates the data from the carrier signal and sends it to terminal B on the 'received data' line 12.

9. Terminal B deserializes the data (through the serdes) using the receive clock signals (on the 'receiver signal element timing' line) 11 from the modem.

10. After terminal A completes its transmission, it deactivates the 'request to send' line 3, which causes the modem to turn off the carrier and deactivate the 'clear to send' line 4.

11. Terminal A and modem A now become receivers and wait for a response from terminal B, indicating that all data has reached terminal B. Modem A begins an echo delay (50 to 150 milliseconds) to ensure that all echoes on the line have diminished before it begins receiving. An echo is a reflection of the transmitted signal. If the transmitting modem changed to receive too soon, it could receive a reflection (echo) of the signal it just transmitted.

12. Modem B deactivates the 'received line signal detector' line 10 and, if necessary, deactivates the receive clock signals on the 'receiver signal element timing' line 11.

13. Terminal B now becomes the transmitter to respond to the request from terminal A. To transmit data, terminal B activates the 'request to send' line 13, which causes modem B to transmit a carrier to modem A.

14. Modem B begins a delay that is longer than the echo delay at modem A before turning on the 'clear to send' line. The longer delay (called request-to-send to clear-to-send delay) ensures that modem A is ready to receive when terminal B begins transmitting data. After the delay, modem B activates the 'clear to send' line 14 to indicate that terminal B can begin transmitting its response.

15. After the echo delay at modem A, modem A senses the carrier from modem B (the carrier was activated in step 13 when terminal B activated the 'request to send' line) and activates the 'received line signal detector' line 7 to terminal A.

16. Modem A and terminal A are now ready to receive the response from terminal B. Remember, the response was not transmitted until after the request-to-send to clear-to-send delay at modem B (step 14).
Establishing a Link on a Nonswitched Multipoint Line

1. The control station serializes the address for the tributary or secondary station (AA) and sends its address to the modem on the 'transmitted data' line 2.

2. Since the 'request to send' line and, therefore, the modem carrier, is active continuously, the modem immediately modulates the carrier with the address, and, thus, the address is transmitted to all modems on the line.

3. All tributary modems, including the modem for station A, demodulate the address and send it to their terminals on the 'received data' line 5.

4. Only station A responds to the address; the other stations ignore the address and continue monitoring their 'received data' line. To respond to the poll, station A activates its 'request to send' line 6, which causes the modem to begin transmitting a carrier signal.

5. The control station's modem receives the carrier and activates the 'received line signal detector' line 3 and the 'receiver signal element timing' line 4 (to send clock signals to the control station). Some modems activate the clock signals as soon as they are powered on.

6. After a short delay to allow the control station modem to receive the carrier, the tributary modem activates the 'clear to send' line 7.

7. When station A detects the active 'clear to send' line, it transmits its response. (For this example, assume that station A has no data to send; therefore, it transmits an EOT 8.)

8. After transmitting the EOT, station A deactivates the 'request to send' line 6. This causes the modem to deactivate the carrier and the 'clear to send' line 7.

9. When the modem at the control station (host) detects the absence of the carrier, it deactivates the 'received line signal detector' line 3.

10. Tributary station A is now in receive mode waiting for the next poll or select transmission from the control station.
Establishing a Link on a Switched Point-To-Point Line

1. Terminal A is in communications mode; therefore, the ‘data terminal ready’ line 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. Modem A detects the ‘switch hook’ line and activates the ‘off hook’ line, which causes the coupler to connect the telephone set to the line and activate the ‘coupler cut-through’ line to the modem.

4. Modem A activates the ‘data modem ready’ line 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. Modem B indicates that the ‘ring indicate’ line was activated by activating the ‘ring indicator’ line to terminal B.

7. Terminal B activates the ‘data terminal ready’ line to modem B, 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.

9. The coupler connects modem B to the communications line through the ‘data tip’ and ‘data ring’ lines and activates the ‘coupler cut-through’ line 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.

11. The coupler at terminal A deactivates the ‘switch hook’ line. This causes modem A to activate the ‘data set ready’ line 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.
APPENDIX G: SWITCH SETTINGS

System Board Switch Settings .......................... G-3
  System Board Switch .................................. G-3
  Math Coprocessor Switch Setting ..................... G-3
  System Board Memory Switch Settings ............... G-4
  Monitor Type Switch Settings ......................... G-4
  5-1/4” Diskette Drive Switch Settings ............... G-5

Extender Card Switch Settings .......................... G-6

Memory Option Switch Settings .......................... G-7
  288K Total Memory .................................. G-7
  320K Total Memory .................................. G-8
  352K Total Memory .................................. G-9
  384K Total Memory .................................. G-10
  416K Total Memory .................................. G-11
  448K Total Memory .................................. G-12
  480K Total Memory .................................. G-13
  512K Total Memory .................................. G-14
  544K Total Memory .................................. G-15
  576K Total Memory .................................. G-16
  608K Total Memory .................................. G-17
  640K Total Memory .................................. G-18
Switches in your system are set to reflect the addition of memory and other installed options. Switches are located on the system board, extender card, and memory expansion options.

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

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

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

G-2 Switch Settings
System Board Switch Settings

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

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

Math Coprocessor Switch Settings

The following figure shows the settings for position 2.

Math Coprocessor installed

Math Coprocessor not installed

Switch Settings  G-3
System Board Memory Switch Settings

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

128K

192K

256K

Monitor Type Switch Settings

No Monitor

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

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

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

IBM Monochrome Display or more than one monitor

G-4 Switch Settings
5 1/4" Diskette Drive Switch Settings

1 DRIVE

3 DRIVES

2 DRIVES

4 DRIVES
## Extender Card Switch Settings

<table>
<thead>
<tr>
<th>System Memory</th>
<th>Memory Segment</th>
</tr>
</thead>
<tbody>
<tr>
<td>16K to 64K</td>
<td>1</td>
</tr>
<tr>
<td>96K to 128K</td>
<td>2</td>
</tr>
<tr>
<td>160K to 192K</td>
<td>3</td>
</tr>
<tr>
<td>224K to 256K</td>
<td>4</td>
</tr>
<tr>
<td>288K to 320K</td>
<td>5</td>
</tr>
<tr>
<td>352K to 384K</td>
<td>6</td>
</tr>
<tr>
<td>416K to 448K</td>
<td>7</td>
</tr>
<tr>
<td>480K to 512K</td>
<td>8</td>
</tr>
<tr>
<td>544K to 576K</td>
<td>9</td>
</tr>
<tr>
<td>608K to 640K</td>
<td>A</td>
</tr>
</tbody>
</table>

G-6  Switch Settings
Memory Option Switch Settings

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

<table>
<thead>
<tr>
<th>64/256K Option Card Switches</th>
<th>64K Option Card Switches</th>
<th>32K Option Card Switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - 32K option</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Appendix G  
Switch Settings G-7
### 320K Total Memory
64K + (256K on System Board)

<table>
<thead>
<tr>
<th>1 - 64/256K option with 64K installed</th>
<th>64/256K Option Card Switches</th>
<th>64K Option Card Switches</th>
<th>32K Option Card Switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - 64K option</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 - 32K options</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 352K Total Memory
96K + (256K on System Board)

<table>
<thead>
<tr>
<th>Switch Settings</th>
<th>64/256K Option Card Switches</th>
<th>64K Option Card Switches</th>
<th>32K Option Card Switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - 64/256K option with 64K installed 1 - 32K option</td>
<td><img src="image1" alt="Switch Diagram" /></td>
<td><img src="image2" alt="Switch Diagram" /></td>
<td><img src="image3" alt="Switch Diagram" /></td>
</tr>
<tr>
<td>1 - 64K option 1 - 32K option</td>
<td><img src="image4" alt="Switch Diagram" /></td>
<td><img src="image5" alt="Switch Diagram" /></td>
<td><img src="image6" alt="Switch Diagram" /></td>
</tr>
<tr>
<td>3 - 32K options</td>
<td><img src="image7" alt="Switch Diagram" /></td>
<td><img src="image8" alt="Switch Diagram" /></td>
<td><img src="image9" alt="Switch Diagram" /></td>
</tr>
</tbody>
</table>
### G-10 Switch Settings

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

<table>
<thead>
<tr>
<th>64/256K Option Card Switches</th>
<th>64K Option Card Switches</th>
<th>32K Option Card Switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - 64/256K option with 64K option installed</td>
<td><img src="Image" alt="Switch Settings" /></td>
<td><img src="Image" alt="Switch Settings" /></td>
</tr>
<tr>
<td>1 - 64K option</td>
<td><img src="Image" alt="Switch Settings" /></td>
<td><img src="Image" alt="Switch Settings" /></td>
</tr>
<tr>
<td>2 - 64K options</td>
<td><img src="Image" alt="Switch Settings" /></td>
<td><img src="Image" alt="Switch Settings" /></td>
</tr>
<tr>
<td>1 - 64/256K option with 64K installed</td>
<td><img src="Image" alt="Switch Settings" /></td>
<td><img src="Image" alt="Switch Settings" /></td>
</tr>
<tr>
<td>2 - 32K options</td>
<td><img src="Image" alt="Switch Settings" /></td>
<td><img src="Image" alt="Switch Settings" /></td>
</tr>
<tr>
<td>1 - 64K option</td>
<td><img src="Image" alt="Switch Settings" /></td>
<td><img src="Image" alt="Switch Settings" /></td>
</tr>
<tr>
<td>2 - 32K options</td>
<td><img src="Image" alt="Switch Settings" /></td>
<td><img src="Image" alt="Switch Settings" /></td>
</tr>
<tr>
<td>1 - 64/256K option with 128K installed</td>
<td><img src="Image" alt="Switch Settings" /></td>
<td><img src="Image" alt="Switch Settings" /></td>
</tr>
</tbody>
</table>
### 416K Total Memory
160K + (256K on System Board)

<table>
<thead>
<tr>
<th>Configuration</th>
<th>64/256K Option Card Switches</th>
<th>64K Option Card Switches</th>
<th>32K Option Card Switches</th>
</tr>
</thead>
</table>
| 1 - 64/256K option with 64K installed  
1 - 64K option  
1 - 32K option | ![Switches](image1) | ![Switches](image2) | ![Switches](image3) |
| 2 - 64K options  
1 - 32K option | | ![Switches](image4) | ![Switches](image5) |
| 1 - 64/256K option with 128K installed  
1 - 32K option | ![Switches](image6) | ![Switches](image7) | ![Switches](image8) |
<table>
<thead>
<tr>
<th>Switch Settings</th>
<th>64/256K Option Card Switches</th>
<th>64K Option Card Switches</th>
<th>32K Option Card Switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - 64/256K option with 192K installed</td>
<td><img src="image1" alt="Switch Positions" /></td>
<td><img src="image2" alt="Switch Positions" /></td>
<td><img src="image3" alt="Switch Positions" /></td>
</tr>
<tr>
<td>1 - 64/256K option with 128K installed</td>
<td><img src="image4" alt="Switch Positions" /></td>
<td><img src="image5" alt="Switch Positions" /></td>
<td><img src="image6" alt="Switch Positions" /></td>
</tr>
<tr>
<td>1 - 64K option</td>
<td><img src="image7" alt="Switch Positions" /></td>
<td><img src="image8" alt="Switch Positions" /></td>
<td><img src="image9" alt="Switch Positions" /></td>
</tr>
<tr>
<td>1 - 64/256K option with 64K installed</td>
<td><img src="image10" alt="Switch Positions" /></td>
<td><img src="image11" alt="Switch Positions" /></td>
<td><img src="image12" alt="Switch Positions" /></td>
</tr>
<tr>
<td>2 - 64K options</td>
<td><img src="image13" alt="Switch Positions" /></td>
<td><img src="image14" alt="Switch Positions" /></td>
<td><img src="image15" alt="Switch Positions" /></td>
</tr>
<tr>
<td>3 - 64K options</td>
<td><img src="image16" alt="Switch Positions" /></td>
<td><img src="image17" alt="Switch Positions" /></td>
<td><img src="image18" alt="Switch Positions" /></td>
</tr>
<tr>
<td>1 - 64/256K option with 128 installed</td>
<td><img src="image19" alt="Switch Positions" /></td>
<td><img src="image20" alt="Switch Positions" /></td>
<td><img src="image21" alt="Switch Positions" /></td>
</tr>
<tr>
<td>2 - 32K options</td>
<td><img src="image22" alt="Switch Positions" /></td>
<td><img src="image23" alt="Switch Positions" /></td>
<td><img src="image24" alt="Switch Positions" /></td>
</tr>
</tbody>
</table>
480K Total Memory
224K + (256K on System Board)

<table>
<thead>
<tr>
<th>Switch Settings</th>
<th>64/256K Option Card Switches</th>
<th>64K Option Card Switches</th>
<th>32K Option Card Switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - 64/256K option with 192K installed</td>
<td><img src="1" alt="Switch Settings" /></td>
<td><img src="2" alt="Switch Settings" /></td>
<td><img src="3" alt="Switch Settings" /></td>
</tr>
<tr>
<td>1 - 32K option</td>
<td><img src="4" alt="Switch Settings" /></td>
<td><img src="5" alt="Switch Settings" /></td>
<td><img src="6" alt="Switch Settings" /></td>
</tr>
<tr>
<td>1 - 64/256K option with 128K installed</td>
<td><img src="7" alt="Switch Settings" /></td>
<td><img src="8" alt="Switch Settings" /></td>
<td><img src="9" alt="Switch Settings" /></td>
</tr>
<tr>
<td>1 - 64K option</td>
<td><img src="10" alt="Switch Settings" /></td>
<td><img src="11" alt="Switch Settings" /></td>
<td><img src="12" alt="Switch Settings" /></td>
</tr>
<tr>
<td>1 - 32K option</td>
<td><img src="13" alt="Switch Settings" /></td>
<td><img src="14" alt="Switch Settings" /></td>
<td><img src="15" alt="Switch Settings" /></td>
</tr>
</tbody>
</table>
### 512K Total Memory
256K + (256K on System Board)

<table>
<thead>
<tr>
<th>Switch Settings</th>
<th>64/256K Option Card Switches</th>
<th>64K Option Card Switches</th>
<th>32K Option Card Switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - 64/256K option with 128K installed</td>
<td><img src="64/256K_Option.png" alt="Image" /></td>
<td><img src="64K_Option.png" alt="Image" /></td>
<td></td>
</tr>
<tr>
<td>2 - 64K options</td>
<td><img src="64/256K_Option.png" alt="Image" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 - 64/256K option with 192K installed</td>
<td><img src="64/256K_Option.png" alt="Image" /></td>
<td></td>
<td><img src="32K_Option.png" alt="Image" /></td>
</tr>
<tr>
<td>1 - 64K option</td>
<td><img src="64/256K_Option.png" alt="Image" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 - 64/256K option with 192K installed</td>
<td><img src="64/256K_Option.png" alt="Image" /></td>
<td></td>
<td><img src="32K_Option.png" alt="Image" /></td>
</tr>
<tr>
<td>2 - 32K options</td>
<td><img src="64/256K_Option.png" alt="Image" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 - 64/256K option with 256K installed</td>
<td><img src="64/256K_Option.png" alt="Image" /></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
544K Total Memory
288K + (256K on System Board)

<table>
<thead>
<tr>
<th></th>
<th>64/256K Option Card Switches</th>
<th>64K Option Card Switches</th>
<th>32K Option Card Switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - 64/256K option with 192K installed</td>
<td><img src="image1" alt="Switches" /></td>
<td><img src="image2" alt="Switches" /></td>
<td><img src="image3" alt="Switches" /></td>
</tr>
<tr>
<td>1 - 64K option</td>
<td><img src="image4" alt="Switches" /></td>
<td><img src="image5" alt="Switches" /></td>
<td><img src="image6" alt="Switches" /></td>
</tr>
<tr>
<td>1 - 32K option</td>
<td><img src="image7" alt="Switches" /></td>
<td><img src="image8" alt="Switches" /></td>
<td><img src="image9" alt="Switches" /></td>
</tr>
<tr>
<td>1 - 64/256K option with 256K installed</td>
<td><img src="image10" alt="Switches" /></td>
<td><img src="image11" alt="Switches" /></td>
<td><img src="image12" alt="Switches" /></td>
</tr>
<tr>
<td>1 - 32K option</td>
<td><img src="image13" alt="Switches" /></td>
<td><img src="image14" alt="Switches" /></td>
<td><img src="image15" alt="Switches" /></td>
</tr>
<tr>
<td>576K Total Memory</td>
<td>32K Option Card Switches</td>
<td>64K Option Card Switches</td>
<td>64/256K Option Card Switches</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------------------------</td>
<td>--------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>320K + (256K on System Board)</td>
<td>1 - 64/256K option with 192K installed</td>
<td>1 - 64/256K option with 256K installed</td>
<td>1 - 64/256K option with 256K installed</td>
</tr>
<tr>
<td>2 - 64K options</td>
<td>1 - 64/256K option with 64K installed</td>
<td>1 - 64K option</td>
<td>2 - 32K options</td>
</tr>
</tbody>
</table>

G-16  Switch Settings
### 608K Total Memory
**352K + (256K on System Board)**

<table>
<thead>
<tr>
<th>64/256K Option Card Switches</th>
<th>64K Option Card Switches</th>
<th>32K Option Card Switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - 64/256K option with 256K installed</td>
<td><img src="image" alt="Switch Settings" /></td>
<td><img src="image" alt="Switch Settings" /></td>
</tr>
<tr>
<td>1 - 64/256K option with 64K installed</td>
<td><img src="image" alt="Switch Settings" /></td>
<td><img src="image" alt="Switch Settings" /></td>
</tr>
<tr>
<td>1 - 32K option</td>
<td><img src="image" alt="Switch Settings" /></td>
<td><img src="image" alt="Switch Settings" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1 - 64/256K option with 256K installed</th>
<th>1 - 64K option</th>
<th>1 - 32K option</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Switch Settings" /></td>
<td><img src="image" alt="Switch Settings" /></td>
<td><img src="image" alt="Switch Settings" /></td>
</tr>
</tbody>
</table>
### 640K Total Memory

384K + (256K on System Board)

<table>
<thead>
<tr>
<th>Switch Settings</th>
<th>64/256K Option Card Switches</th>
<th>64K Option Card Switches</th>
<th>32K Option Card Switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 - 64/256K option with 256K installed</td>
<td><img src="image1" alt="Switches" /></td>
<td><img src="image2" alt="Switches" /></td>
<td></td>
</tr>
<tr>
<td>1 - 64/256K option with 64K installed</td>
<td><img src="image3" alt="Switches" /></td>
<td><img src="image4" alt="Switches" /></td>
<td></td>
</tr>
<tr>
<td>1 - 64K option</td>
<td><img src="image5" alt="Switches" /></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 - 64/256K option with 256K installed</td>
<td><img src="image1" alt="Switches" /></td>
<td><img src="image2" alt="Switches" /></td>
<td></td>
</tr>
<tr>
<td>2 - 64K options</td>
<td><img src="image3" alt="Switches" /></td>
<td></td>
<td><img src="image6" alt="Switches" /></td>
</tr>
<tr>
<td>1 - 64/256K option with 256K installed</td>
<td><img src="image1" alt="Switches" /></td>
<td><img src="image2" alt="Switches" /></td>
<td><img src="image7" alt="Switches" /></td>
</tr>
<tr>
<td>1 - 64/256K option with 128K installed with 64K</td>
<td><img src="image3" alt="Switches" /></td>
<td><img src="image4" alt="Switches" /></td>
<td><img src="image8" alt="Switches" /></td>
</tr>
</tbody>
</table>
Glossary

μs: Microsecond.

adapter: An auxiliary system or unit used to extend the operation of another system.

address bus: One or more conductors used to carry the binary-coded address from the microprocessor throughout the rest of the system.

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

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

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

A/N: Alphanumeric.

analog: (1) pertaining to data in the form of continuously variable physical quantities. (2) Contrast with digital.

AND: A logic operator having the property that if P is a statement, Q is a statement, R is a statement,..., then the AND of P, Q, R,... is true if all statements are true, false if any statement is false.

APA: All points addressable.

assembler: A computer program used to assemble. Synonymous with assembly program.

asynchronous communications: A communication mode in which each single byte of data is synchronized, usually by the addition of start/stop bits.

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

basic input/output system (BIOS): Provides the device level control of the major I/O devices in a computer system, which provides an operational interface to the system and relieves the programmer from concern over hardware device characteristics.

baud: (1) A unit of signaling speed equal to the number of discrete conditions or signal events per second. For example, one baud equals one-half dot cycle per second in Morse code, one bit per second in a train of binary signals, and one 3-bit value per second in a train of signals each of which can assume one of eight different states. (2) In asynchronous transmission, the unit of modulation rate corresponding to one unit of interval per second; that is, if the duration of the unit interval is 20 milliseconds, the modulation rate is 50 baud.

BCC: Block-check character.

beginner's all-purpose symbolic instruction code (BASIC): A programming language with a small repertoire of commands and a simple syntax, primarily designed for numerical application.

binary: (1) Pertaining to a selection, choice, or condition that has two possible values or states. (2) Pertaining to a fixed radix numeration system having a radix of two.

binary digit: (1) In binary notation, either of the characters 0 or 1. (2) Synonymous with bit.

binary notation: Any notation that uses two different characters, usually the binary digits 0 and 1.

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

bit: In binary notation, either of the characters 0 or 1.

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

block-check character (BCC): In cyclic redundancy checking, a character that is transmitted by the sender after each message block and is compared with a block-check character computed by the receiver to determine if the transmission was successful.

boolean operation: (1) Any operation in which each of the operands and the result take one of two values. (2) An operation that follows the rules of boolean algebra.

bootstrap: A technique or device designed to bring itself into a desired state by means of its own action; that is, a machine routine whose first few instructions are sufficient to bring the rest of itself into the computer from an input device.

bps: Bits per second.

BSC: Binary synchronous communications.

buffer: (1) An area of storage that is temporarily reserved for use in performing an input/output operation, into which data is read or from which data is written. Synonymous with I/O area. (2) A portion of storage for temporarily holding input or output data.

bus: One or more conductors used for transmitting signals or power.

byte: (1) A binary character operated upon as a unit and usually shorter than a computer word. (2) The representation of a character.

CAS: Column address strobe.

cathode ray tube (CRT): A vacuum tube display in which a beam of electrons can be controlled to form alphanumeric characters or symbols on a luminescent screen, for example by use of a dot matrix.
cathode ray tube display (CRT display): (1) A device that presents data in visual form by means of controlled electron beams. (2) The data display produced by the device as in (1).

CCITT: Comité Consultatif International Télégraphique et Téléphonique.

central processing unit (CPU): A functional unit that consists of one or more processors and all or part of internal storage.

channel: A path along which signals can be sent; for example, data channel or I/O channel.

characters per second (cps): A standard unit of measurement for printer output.

code: (1) A set of unambiguous rules specifying the manner in which data may be represented in a discrete form. Synonymous with coding scheme. (2) A set of items, such as abbreviations, representing the members of another set. (3) Loosely, one or more computer programs, or part of a computer program. (4) To represent data or a computer program in a symbolic form that can be accepted by a data processor.

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

Comité Consultatif International Télégraphique et Téléphonique (CCITT): Consultative Committee on International Telegraphy and Telephony.

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

configuration: (1) The arrangement of a computer system or network as defined by the nature, number, and the chief characteristics of its functional units. More specifically, the term configuration may refer to a hardware configuration or a software configuration. (2) The devices and programs that make up a system, subsystem, or network.
conjunction: (1) The boolean operation whose result has the boolean value 1 if, and only if, each operand has the boolean value 1. (2) Synonymous with AND operation.

contiguous: (1) Touching or joining at the edge or boundary. (2) Adjacent.

CPS: Characters per second.

CPU: Central processing unit.

CRC: Cyclic redundancy check.

CRT: Cathode ray tube.

CRT display: Cathode ray tube display.

CTS: Clear to send. Associated with modem control.

cyclic redundancy check (CRC): (1) A redundancy check in which the check key is generated by a cyclic algorithm. (2) A system of error checking performed at both the sending and receiving station after a block-check character has been accumulated.

cylinder: (1) The set of all tracks with the same nominal distance from the axis about which the disk rotates. (2) The tracks of a disk storage device that can be accessed without repositioning the access mechanism.

daisy-chained cable: A type of cable that has two or more connectors attached in series.

data: (1) A representation of facts, concepts, or instructions in a formalized manner suitable for communication, interpretation, or processing by humans or automatic means. (2) Any representations, such as characters or analog quantities, to which meaning is, or might be assigned.

decoupling capacitor: A capacitor that provides a low-impedance path to ground to prevent common coupling between states of a circuit.

Deutsche Industrie Norm (DIN): (1) German Industrial Norm. (2) The committee that sets German dimension standards.
digit: (1) A graphic character that represents an integer, for example, one of the characters 0 to 9. (2) A symbol that represents one of the non-negative integers smaller than the radix. For example, in decimal notation, a digit is one of the characters from 0 to 9.

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

DIN: Deutsche Industrie Norm.

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

DIP: Dual in-line package.

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

disk: Loosely, a magnetic disk unit.

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

DMA: Direct memory access.

DSR: Data set ready. Associated with modem control.

DTR: Data terminal ready. Associated with modem control.

dual in-line package (DIP): A widely used container for an integrated circuit. DIPs are pins usually in two parallel rows. These pins are spaced 1/10 inch apart and come in different configurations ranging from 14-pin to 40-pin configurations.

EBCDIC: Extended binary-coded decimal interchange code.

ECC: Error checking and correction.

edge connector: A terminal block with a number of contacts attached to the edge of a printed circuit board to facilitate plugging into a foundation circuit.
EIA: Electronic Industries Association.

**EIA/CCITT:** Electronics Industries Association/Consultative Committee on International Telegraphy and Telephony.

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

end-of-transmission character (EOT): A transmission control character used to indicate the conclusion of a transmission, which may have included one or more texts and any associated message headings.

EOT: End-of-transmission character.

EPROM: Erasable programmable read-only memory.

erasable programmable read-only memory (EPROM): A storage device whose contents can be changed by electrical means. EPROM information is not destroyed when power is removed.

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

ETX: End-of-text character.

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

flexible disk: Synonym for diskette.

firmware: Memory chips with integrated programs already incorporated on the chip.

gate: (1) A device or circuit that has no output until it is triggered into operation by one or more enabling signals, or until an input signal exceeds a predetermined threshold amplitude. (2) A signal that triggers the passage of other signals through a circuit.

graphic: A symbol produced by a process such as handwriting, drawing, or printing.
hertz (Hz): A unit of frequency equal to one cycle per second.

hex: Abbreviation for hexadecimal.

hexadecimal: Pertaining to a selection, choice, or condition that has 16 possible values or states. These values or states usually contain 10 digits and 6 letters, A through F. Hexadecimal digits are equivalent to a power of 16.

high-order position: The leftmost position in a string of characters.

Hz: Hertz.

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

k: An abbreviation for the prefix kilo; that is, 1,000 in decimal notation.

K: When referring to storage capacity, 2 to the tenth power; 1,024 in decimal notation.

KB: Kilobyte; 1,024 bytes.

kHz: A unit of frequency equal to 1,000 hertz.

kilo (k): One thousand.

latch: (1) A feedback loop in symmetrical digital circuits used to maintain a state. (2) A simple logic-circuit storage element comprising two gates as a unit.

LED: Light-emitting diode.

light-emitting diode (LED): A semi-conductor chip that gives off visible or infrared light when activated.

low-order position: The rightmost position in a string of characters.

m: (1) Milli; one thousand or thousandth part. (2) Meter
M: Mega; 1,000,000 in decimal notation. When referring to storage capacity, 2 to the twentieth power; 1,048,576 in decimal notation.

mA: Milliampere.

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

main storage: A storage device in which the access time is effectively independent of the location of the data.

MB: Megabyte, 1,048,576 bytes.

mega (M): 10 to the sixth power, 1,000,000 in decimal notation. When referring to storage capacity, 2 to the twentieth power, 1,048,576 in decimal notation.

megabyte (MB): 1,048,576 bytes.

megahertz (MHz): A unit of measure of frequency. 1 megahertz equals 1,000,000 hertz.

MFM: Modified frequency modulation.

MHz: Megahertz.

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

microsecond (μs): One-millionth of a second.

**milli** (m): One thousand or one thousandth.

milliampere (mA): One thousandth of an ampere.

millisecond (ms): One thousandth of a second.

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

mode: (1) A method of operation; for example, the binary mode, the interpretive mode, the alphanumeric mode. (2) The most frequent value in the statistical sense.
modem: (Modulator-Demodulator) A device that converts serial (bit by bit) digital signals from a business machine (or data terminal equipment) to analog signals which are suitable for transmission in a telephone network. The inverse function is also performed by the modem on reception of analog signals.

modified frequency modulation (MFM): The process of varying the amplitude and frequency of the "write" signal. MFM pertains to the number of bytes of storage that can be stored on the recording media. The number of bytes is twice the number contained in the same unit area of recording media at single density.

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

monitor: (1) A device that observes and verifies the operation of a data processing system and indicates any specific departure from the norm. (2) A television type display, such as the IBM Monochrome Display. (3) Software or hardware that observes, supervises, controls, or verifies the operations of a system.

ms: Millisecond; one thousandth of a second.

multiplexer: A device capable of interleaving the events of two or more activities, or capable of distributing the events of an interleaved sequence to the respective activities.

NAND: A logic operator having the property that if P is a statement, Q is a statement, R is a statement,...,then the NAND of P,Q,R,... is true if at least one statement is false, false if all statements are true.

nanosecond (ns): One-thousandth-millionth of a second.

nonconjunction: The dyadic boolean operation the result of which has the boolean value 0 if, and only if, each operand has the boolean value 1.

non-return-to-zero inverted (NRZI): A transmission encoding method in which the data terminal equipment changes the signal to the opposite state to send a binary 0 and leaves it in the same state to send a binary 1.
NOR: A logic operator having the property that if $P$ is a statement, $Q$ is a statement, $R$ is a statement,...,then the NOR of $P, Q, R, ...$ is true if all statements are false, false if at least one statement is true.

NOT: A logical operator having the property that if $P$ is a statement, then the NOT of $P$ is true if $P$ is false, false if $P$ is true.

NRZI: Non-return-to-zero inverted.

ns: Nanosecond; one-thousandth-millionth of a second.

operating system: Software that controls the execution of programs; an operating system may provide services such as resource allocation, scheduling, input/output control, and data management.

OR: A logic operator having the property that if $P$ is a statement, $Q$ is a statement, $R$ is a statement,..., then the OR of $P, Q, R, ...$ is true if at least one statement is true, false if all statements are false.

output: Pertaining to a device, process, or channel involved in an output process, or to the data or states involved in an output process.

output process: (1) The process that consists of the delivery of data from a data processing system, or from any part of it. (2) The return of information from a data processing system to an end user, including the translation of data from a machine language to a language that the end user can understand.

overcurrent: A current of higher than specified strength.

overvoltage: A voltage of higher than specified value.

parallel: (1) Pertaining to the concurrent or simultaneous operation of two or more devices, or to the concurrent performance of two or more activities. (2) Pertaining to the concurrent or simultaneous occurrence of two or more related activities in multiple devices or channels. (3) Pertaining to the simultaneity of two or more processes. (4) Pertaining to the simultaneous processing of the individual parts of a whole, such as the bits of a character and the characters of a word, using separate facilities for the various parts. (5) Contrast with serial.
PEL: Picture element.

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

picture element (PEL): (1) The smallest displayable unit on a display. (2) Synonymous with pixel, PEL.

pinout: A diagram of functioning pins on a pinboard.

pixel: Picture element.

polling: (1) Interrogation of devices for purposes such as to avoid contention, to determine operational status, or to determine readiness to send or receive data. (2) The process whereby stations are invited, one at a time, to transmit.

port: An access point for data entry or exit.

printed circuit board: A piece of material, usually fiberglass, that contains a layer of conductive material, usually metal. Miniature electronic components on the fiberglass transmit electronic signals through the board by way of the metal layers.

program: (1) A series of actions designed to achieve a certain result. (2) A series of instructions telling the computer how to handle a problem or task. (3) To design, write, and test computer programs.

programming language: (1) An artificial language established for expressing computer programs. (2) A set of characters and rules, with meanings assigned prior to their use, for writing computer programs.

PROM: Programmable read-only memory.

propagation delay: The time necessary for a signal to travel from one point on a circuit to another.

radix: (1) In a radix numeration system, the positive integer by which the weight of the digit place is multiplied to obtain the weight of the digit place with the next higher weight; for example, in the decimal numeration system, the radix of each digit place is 10. (2) Another term for base.

H-12 Glossary
radix numeration system: A positional representation system in which the ratio of the weight of any one digit place to the weight of the digit place with the next lower weight is a positive integer. The permissible values of the character in any digit place range from zero to one less than the radix of the digit place.

- RAS: Row address strobe.

RGBI: Red-green-blue-intensity.

read-only memory (ROM): A storage device whose contents cannot be modified, except by a particular user, or when operating under particular conditions; for example, a storage device in which writing is prevented by a lockout.

read/write memory: A storage device whose contents can be modified.

red-green-blue-intensity (RGBI): The description of a direct-drive color monitor which accepts red, green, blue, and intensity signal inputs.

register: (1) A storage device, having a specified storage capacity such as a bit, a byte, or a computer word, and usually intended for a special purpose. (2) On a calculator, a storage device in which specific data is stored.

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

ROM: Read-only memory.

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

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

RS-232C: The standard set by the EIA for communications between computers and external equipment.

RTS: Request to send. Associated with modem control.

run: A single continuous performance of a computer program or routine.
scan line: The use of a cathode beam to test the cathode ray tube of a display used with a personal computer.

schematic: The description, usually in diagram form, of the logical and physical structure of an entire data base according to a conceptual model.

SDLC: Synchronous Data Link Control.

sector: That part of a track or band on a magnetic drum, a magnetic disk, or a disk pack that can be accessed by the magnetic heads in the course of a predetermined rotational displacement of the particular device.

serdes: Serializer/deserializer.

serial: (1) Pertaining to the sequential performance of two or more activities in a single device. In English, the modifiers serial and parallel usually refer to devices, as opposed to sequential and consecutive, which refer to processes. (2) Pertaining to the sequential or consecutive occurrence of two or more related activities in a single device or channel. (3) Pertaining to the sequential processing of the individual parts of a whole, such as the bits of a character or the characters of a word, using the same facilities for successive parts. (4) Contrast with parallel.

sink: A device or circuit into which current drains.

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

source: The origin of a signal or electrical energy.

source circuit: (1) Generator circuit. (2) Control with sink.

SS: Start-stop transmission.

start bit: Synonym for start signal.

start-of-text character (STX): A transmission control character that precedes a text and may be used to terminate the message heading.
start signal: (1) A signal to a receiving mechanism to get ready to receive data or perform a function. (2) In a start-stop system, a signal preceding a character or block that prepares the receiving device for the reception of the code elements. Synonymous with start bit.

start-stop (SS) transmission: Asynchronous transmission such that a group of signals representing a character is preceded by a start signal and followed by a stop signal. (2) Asynchronous transmission in which a group of bits is preceded by a start bit that prepares the receiving mechanism for the reception and registration of a character and is followed by at least one stop bit that enables the receiving mechanism to come to an idle condition pending the reception of the next character.

stop bit: Synonym for stop signal.

stop signal: (1) A signal to a receiving mechanism to wait for the next signal. (2) In a start-stop system, a signal following a character or block that prepares the receiving device for the reception of a subsequent character or block. Synonymous with stop bit.

strobe: (1) An instrument used to determine the exact speed of circular or cyclic movement. (2) A flashing signal displaying an exact event.

STX: Start-of-text character.

Synchronous Data Link Control (SLDC): A protocol for the management of data transfer over a data communications link.

synchronous transmission: Data transmission in which the sending and receiving devices are operating continuously at the same frequency and are maintained, by means of correction, in a desired phase relationship.

text: In ASCII and data communication, a sequence of characters treated as an entity if preceded and terminated by one STX and one ETX transmission control, respectively.
track: (1) The path or one of the set of paths, parallel to the reference edge on a data medium, associated with a single reading or writing component as the data medium moves past the component. (2) The portion of a moving data medium such as a drum, tape, or disk, that is accessible to a given reading head position.

transistor-transistor logic (TTL): A circuit in which the multiple-diode cluster of the diode-transistor logic circuit has been replaced by a multiple-emitter transistor.

**TTL**: Transistor-transistor logic.

**TX Data**: Transmit data. Associated with modem control. External connections of the RS-232C asynchronous communications adapter interface.

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

**write precompensation**: The varying of the timing of the head current from the outer tracks to the inner tracks of the diskette to keep a constant write signal.
This manual introduces the 8086 family of microcomputing components and serves as a reference in system design and implementation.

Intel Corporation. *8086/8087/8088 Macro Assembly Reference Manual for 8088/8085 Based Development System*
This manual describes the 8086/8087/8088 Macro Assembly Language, and is intended for use by persons who are familiar with assembly language.

Intel Corporation. *Component Data Catalog*
This book describes Intel components and their technical specifications.

Motorola, Inc. *The Complete Microcomputer Data Library.*
This book describes Motorola components and their technical specifications.

Notes:

I-2 Bibliography
INDEX

A

A/N mode (alphanumeric mode) 1-123
A0-A19 (Address Bits 0 to 19), I/O channel 1-15
adapter card with ROM 2-10
adapter,
    asynchronous communication 1-215
    binary synchronous communication 1-245
    color/graphics monitor 1-123
    diskette drive 1-151
    fixed disk drive 1-179
    game control 1-203
    monochrome display and printer 1-115
    printer 1-113
    synchronous data link control 1-265
Address Bits 0 to 19 (A0-A19), I/O channel 1-15
address bits (asynchronous communication) 1-217
Address Enable (AEN), I/O channel 1-18
Address Latch Enable (ALE), I/O channel 1-15
address map, I/O 1-8
AEN (Address Enable), I/O channel 1-18
ALE (Address Latch Enable), I/O channel 1-15
all points addressable mode 1-129, 1-123
alphanumeric mode, 1-128
    high resolution 1-135
    low resolution 1-132
alt (keyboard extended code) 2-15
APA mode (all points addressable mode) 1-124
asynchronous communications adapter, 1-215
    adapter address jumper module 1-242
    address bits 1-217
    block diagram 1-216
    connector specifications 1-243
    current loop interface 1-219
    divisor latch least significant bit 1-229
    divisor latch most significant bit 1-230
    I/O decode 1-217
    INS8250 functional pin description 1-221
    INS8250 input signals 1-221
    INS8250 input/output signals 1-225
INS8250 output signals  1-224
interface descriptions  1-218
interface format jumper module  1-242
interrupt control functions  1-234
interrupt enable register  1-235
interrupt identification register  1-233
interrupts  1-218
line control register  1-227
line status register  1-231
modem control register  1-236
modem status register  1-238
modes of operation  1-216
programmable baud rate generator  1-229
programming considerations  1-226
receiver buffer register  1-240
reset functions  1-226
transmitter holding register  1-241
voltage interchange information  1-220
attributes, character
(see character attributes)

B

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

J-2  Index
read/write control logic 1-247
receive 1-252
receiver buffer 1-249
receiver control 1-249
status read definition 1-259
transmit 1-251
transmitter buffer 1-248
transmitter control 1-249
typical programming sequence 1-253

BIOS,
fixed disk ROM A-84
memory map 2-9
parameter passing 2-3
software interrupt listing 2-4
system ROM A-2
use of 2-2
bisync communications
(see binary synchronous communications adapter)

block diagram
8252A universal synchronous/asynchronous receiver/transmitter 1-246
8273 SDLC protocol controller 1-267
asynchronous communications adapter 1-215
color/graphics monitor adapter 1-125
coprocessor 1-25
diskette drive adapter 1-151
expansion board 1-71
extender card 1-74
fixed disk drive adapter 1-179
game control adapter 1-203
keyboard interface 1-67
monochrome display adapter 1-114
printer adapter 1-108
prototype card 1-118
receiver card 1-77
speaker drive system 1-20
synchronous data link control adapter 1-265
system 1-2
break (keyboard extended code) 2-14

BSC adapter
(see binary synchronous communications)
cable
  communications adapter  1-285
  expansion unit  1-171
  printer  1-81
cancel (printer)  1-93
cancel ignore paper end (printer)  1-95
cancel skip perforation (printer)  1-99
caps lock (keyboard extended code)  2-16
card dimensions and specifications  E-4
card selected  1-19
CARD SLCTD (card selected), I/O channel  1-19
card,
  dimensions and specifications  E-4
  extender  1-74
  prototype  1-209
  receiver  1-77
carriage return (printer)  1-92
CCITT,  F-1
  standards  F-1
character attributes
  color/graphics monitor adapter  1-130
  monochrome display adapter  1-130
character codes
  keyboard  2-11
character set,
  graphics printer (set 1)  1-103
  graphics printer (set 2)  1-105
  matrix printer  1-101
  quick reference  C-12
clear printer buffer (printer)  1-100
CLK (system clock), I/O channel  1-16
color display  1-149
  operating characteristics  1-150
  specifications  E-2
color select register  1-141
color/graphics monitor adapter  1-123
  6845 register description  1-136
  alphanumerical mode  1-127
  alphanumerical mode (high-resolution)  1-128
  alphanumerical mode (low-resolution)  1-128
  block diagram  1-126
  character attributes  1-130
  color-select register  1-141
  composite connector specifications  1-146
connector specifications 1-146
direct-drive connector specifications 1-146
display buffer basic operation 1-127
graphics mode 1-132
graphics mode (high resolution) 1-135
graphics mode (low resolution) 1-132
graphic mode (medium resolution) 1-133
light pen connector specifications 1-147
major components 1-126
memory requirements 1-145
mode control and status register 1-143
mode register summary 1-143
mode select register 1-141
programming considerations 1-137
RF modulator connector specifications 1-147
sequence of events 1-144
status register 1-143
summary of available color 1-135
colors, summary of available 1-137
command status register 0 1-164
command status register 1 1-165
command status register 2 1-166
command status register 3 1-167
command summary,
   diskette drive adapter 1-151
   fixed disk drive adapter 1-179
communications adapter cable 1-285
   connector specifications 1-286
communications F-1
   establishing a link F-3
component diagram,
   system board 1-13
compressed (printer) 1-93
compressed off (printer) 1-93
connector specifications,
   asynchronous communications adapter 1-215
   binary synchronous communications 1-245
   color/graphics monitor adapter 1-147
   communications adapter cable 1-286
   diskette drive adapter (external) 1-174
   diskette drive adapter (internal) 1-173
game control adapter 1-203
keyboard interface 1-67
monochrome display adapter 1-115
printer adapter 1-81
synchronous data link control adapter 1-293
connectors,
  power supply (system unit) 1-21
consideration, programming
  (see programming considerations)
control byte, fixed disk drive adapter 1-186
control codes, printer 1-91
control/read/write logic 1-268
coprocessor,
  (see math coprocessor)
ctr1 (keyboard extended code) 2-13
current loop interface 1-219

D

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

J-6  Index
diskette drive, 1-175
  electrical specifications 1-176
  mechanical specifications 1-176
  switch settings G-1
diskettes 1-177
display adapter type switch settings G-1
display,
  color 1-149
  monochrome 1-121
divisor latch,
  least significant bit 1-229
  most significant bit 1-230
DMA Acknowledge 0 to 3 (DACK0-DACK3),
I/O channel 1-18
DMA Request 1 to 3 (DRQ1-DRQ3), I/O channel 1-18
DOS reserved interrupts 2-7
DOS,
  keyboard functions 2-20
  reserved interrupts 2-7
double strike (printer) 1-96
double strike off (printer) 1-96
double width (printer) 1-93
double width off (printer) 1-93
DPC registers 1-167
DRQ1-DRQ3 (DMA Request 1 to 3), I/O channel 1-18

E

EIA, F-1
  standards F-1
  emphasized (printer) 1-96
  emphasized off (printer) 1-96
  escape (printer) 1-93
  establishing a communications link F-3
expansion board, 1-71
  block diagram 1-72
expansion channel 1-73
expansion unit, 1-71
  cable 1-71
  expansion board 1-71
  expansion channel 1-73
  extender card 1-74
  interface information 1-79
  power supply 1-71
  receiver card 1-77
  specifications E-2
extender card  1-74
  block diagram  1-76
  programming considerations  1-75
  switch settings  G-1

F

FABS (coprocessor)  1-36
FADD (coprocessor)  1-36
FBLD (coprocessor)  1-37
FBSTP (coprocessor)  1-37
FCHS (coprocessor)  1-38
FCLEX/FNCLEX (coprocessor)  1-38
FCOM (coprocessor)  1-38
FCOMP (coprocessor)  1-39
FCOMPP (coprocessor)  1-39
FDECSTP (coprocessor)  1-40
FDISI/FNDISI (coprocessor)  1-40
FDIV (coprocessor)  1-41
FDIVR (coprocessor)  1-42
FENI/FNENI (coprocessor)  1-43
FFREE (coprocessor)  1-43
FICOM (coprocessor)  1-43
FICOMP (coprocessor)  1-44
FILD (coprocessor)  1-44
FINCSTP (coprocessor)  1-44
FINIT/FNINIT (coprocessor)  1-45
FIST (coprocessor)  1-46
FISTP (coprocessor)  1-46
fixed disk controller  1-179
fixed disk drive  1-195
fixed disk drive adapter  1-179
  block diagram  1-180
  command summary  1-187
  control byte  1-186
  data register  1-185
  fixed disk controller  1-179
  interface specifications  1-193
  programming considerations  1-181
  programming summary  1-191
ROM BIOS listing  A-84
sense bytes  1-181
status register  1-181
system I/O channel interface  1-192
fixed disk drive, 1-195
  electrical specifications 1-196
  mechanical specifications 1-196
fixed disk ROM BIOS A-84
FLD (coprocessor) 1-47
FLDCW (coprocessor) 1-47
FLDENV (coprocessor) 1-48
FLDLG2 (coprocessor) 1-48
FLDLN2 (coprocessor) 1-48
FLDL2E (coprocessor) 1-49
FLDL2T (coprocessor) 1-49
FLDPI (coprocessor) 1-49
FLDZ (coprocessor) 1-50
FLD1 (coprocessor) 1-50
floppy disk controller 1-154
form feed (printer) 1-92
FMUL 1-51
FNOP 1-52
FPATAN 1-52
FPREM 1-52
FPTAN 1-53
FRNDINT 1-53
FRSTOR 1-53
FSAVE/FNSAVE (coprocessor) 1-54
FSCALE (coprocessor) 1-54
FSQRT (coprocessor) 1-55
FST (coprocessor) 1-55
FSTCW/FNSTCW (coprocessor) 1-56
FSTENV/FNSTENV (coprocessor) 1-56
FSTP (coprocessor) 1-57
FSTSW/FNSTSW (coprocessor) 1-57
FSUB (coprocessor) 1-58
FSUBR (coprocessor) 1-59
FTST (coprocessor) 1-60
FWAIT (coprocessor) 1-60
FXAM (coprocessor) 1-61
FXCH (coprocessor) 1-62
FXTRACT (coprocessor) 1-62
FYL2X (coprocessor) 1-63
FYL2XP1 (coprocessor) 1-63
F2XM1 (coprocessor) 1-64
G

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

H

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

I

I/O address map 1-8
I/O bit map, 8255A 1-10
I/O CH CK (I/O Channel Check), I/O channel 1-17
I/O CH RDY (I/O Channel Ready), I/O channel 1-17
I/O Channel Check (I/O CH CK), I/O channel 1-17
I/O channel interface,
  diskette drive adapter 1-168
  fixed disk drive adapter 1-192
  prototype card 1-211
I/O Channel Ready (I/O CH RDY), I/O channel 1-17
I/O channel, 1-14
  -I/O Channel Check (I/O CH CK) 1-17
  -I/O Read Command (IOR) 1-17
  -I/O Write Command (IOW) 1-17
Address Bits 0 to 19 (A0-A19) 1-16
Address Enable (AEN) 1-18
Address Latch Enable (ALE) 1-16
Data Bits 0 to 7 (D0-D7) 1-16
description 1-16
diagram 1-15
DMA Request 1 to 3 (DRQ1-DRQ3) 1-18
I/O Channel Ready (I/O CH RDY) 1-17
Interrupt Request 2 to 7 (IRQ2-IRQ7)  1-17
Memory Read Command (MEMR)  1-18
Memory Write Command (MEMW)  1-18
Oscillator (OSC)  1-16
Reset Drive (RESET DRV)  1-16
System Clock (CLK)  1-16
Terminal Count (T/C)  1-18
I/O Read Command (IOR), I/O channel  1-17
I/O Write Command (IOW), I/O channel  1-17
IBM 10MB Fixed Disk Drive  1-195
IBM 5-1/4" Diskette Drive  1-175
IBM 5-1/4" Diskette Drive Adapter  1-151
IBM 80 CPS Graphics Printer  1-81
IBM 80 CPS Matrix Printer  1-81
IBM 80 CPS Printers  1-81
IBM Asynchronous Communications Adapter  1-215
IBM Binary Synchronous Communications Adapter  1-245
IBM Color Display  1-149
IBM Color/Graphics Monitor Adapter  1-123
IBM Communicatons Adapter Cable  1-295
IBM Fixed Disk Drive Adapter  1-179
IBM Game Control Adapter  1-203
IBM Memory Expansion Options  1-197
IBM Monochrome Display and Printer Adapter  1-113
IBM Monochrome Display  1-121
IBM Personal Computer Math Coprocessor  1-25
IBM Printer Adapter  1-107
IBM Prototype Card  1-209
IBM Synchronous Data Link Controller Adapter  1-265
ignore paper end (printer)  1-94
INS8250,
    (see National Semiconductor INS8250)
Intel 8088 microprocessor,
     arithmetic  B-7
     conditional transfer operations  B-14
     control transfer  B-11
     data transfer  B-5
     hardware interrupt listing  1-8
     instruction set index  B-18
     instruction set matrix  B-16
     logic  B-9
     memory segmentation model  B-4
     operand summary  B-3
     processor control  B-14
     register model  B-2
second instruction byte summary  B-3
segment override prefix  B-4
software interrupt listing  2-4
string manipulation  B-11
use of segment override  B-4

Intel 8253-5 Programmable Interval Timer
(see synchronous data link control communications adapter)

Intel 8255A Programmable Peripheral Interface
I/O bit map  1-10

Intel 8255A-5 Programmable Peripheral Interface
(see synchronous data link control communications adapter)

Intel 8273 SDLC Protocol Controller
(see synchronous data link control communications adapter)
block diagram  1-265
interrupt enable register  1-235
interrupt identification register  1-233
interrupt listing,
  8088 hardware  1-9
  8088 software  2-4

Interrupt Request 2 to 7 (IRQ2-IRQ7), I/O channel  1-17
interrupts,
  8088 hardware  1-9
  8088 software  2-4
asynchronous communications adapter  1-215
BASIC reserved  2-7
DOS reserved  2-7
special  2-5

IOR (I/O Read Command), I/O channel  1-17
IOW (I/O Write Command), I/O channel  1-17
IRQ2-IRQ7 (Interrupt Request 2 to 7), I/O channel  1-17

J

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

K

keyboard extended codes,
  alt  2-15
  break  2-16
  caps lock  2-16
  ctrl  2-15

J-12  Index
pause 2-17
print screen 2-17
scroll lock 2-16
shift 2-15
shift key priorities 2-16
shift states 2-15
system reset 2-16
keyboard 1-65
BASIC screen editor special functions 2-20
character codes 2-11
commonly used functions 2-18
diagram 1-68
DOS special functions 2-20
encoding 2-11
extended functions 2-14
interface block diagram 1-70
interface connector specifications 1-70
scan codes 1-69
specifications E-1

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

M
math coprocessor 1-25
block diagram 1-29
control unit 1-29
control word 1-32
data types 1-26
exception pointers 1-33
FABS 1-36
FADD 1-36
FBLD 1-37
FBSTP 1-37
FCHS 1-38
FCLEX/FNCLEX 1-38
FCOM 1-38
FCOMP 1-39
FCOMPP 1-39
FDECSTP 1-40
FDISI/FNDISI 1-40
FDIV 1-41
FDIVR 1-42
FENI/FNENI 1-43
FFREE 1-43
FICOM 1-43
FICOMP 1-44
FILD 1-44
FINCSTP 1-44
FINIT/FNINIT 1-45
FIST 1-46
FISTP 1-46
FLD 1-47
FLDCW 1-47
FLDENV 1-48
FLDLG2 1-48
FLDLN2 1-48
FLDL2E 1-49
FLDL2T 1-49
FLDPI 1-49
FLDZ 1-50
FLD1 1-50
FMUL 1-51
FNOP 1-52
FPATAN 1-52
FPREM 1-52
FPTAN 1-53
FRNDINT 1-53
FRSTOR 1-53
FSAVE/FNSAVE 1-54
FSCALE 1-54
FSQRT 1-55
FST 1-55
FSTCW/FNSTCW 1-56
FSTENV/FNSTENV 1-56
FSTP 1-57
FSTSW/FNSTSW 1-57
FSUB 1-58
FSUBR 1-59
FTST 1-60
FWAIT 1-60
FXAM 1-61
FXCH 1-62
FXTRACT 1-62
FY12X 1-63
FY12XP1 1-63
F2XM1 1-64

J-14 Index
hardware interface 1-27
instruction set 1-35
interconnection 1-28
number system 1-34
programming interface 1-26
register stack 1-30
status word 1-31
tag word 1-33
memory expansion options, 1-197
DIP module start address 1-201
memory module description 1-198
memory module pin configuration 1-199
memory option switch settings G-1
R/W memory operating characteristics 1-198
switch-configurable start address 1-200
memory locations,
reserved 2-8
memory map,
BIOS 2-9
system 1-11
Memory Read Command (MEMR), I/O channel 1-18
memory switch settings, G-1
extender card G-1
memory options G-1
system board G-1
Memory Write Command (MEMW), I/O channel 1-18
(MEMR) Memory Read Command, I/O channel 1-18
(MEMW) Memory Write Command, I/O channel 1-18
microprocessor (see Intel 8088 microprocessor)
mode control and status register 1-139
mode select register 1-141
modem control register 1-236
modem status register 1-238
monochrome display 1-121
monochrome display and printer adapter 1-113
monochrome display adapter 1-115
6845 CRT control port 1-118
6845 CRT status port 1-118
block diagram 1-114
character attributes 1-117
connector specifications 1-119
I/O address and bit map 1-117
programming considerations 1-115
monochrome display, 1-121
operating characteristics 1-122
specifications E-3

Index J-15
Motorola 6845 CRT Controller,
(see color/graphics monitor adapter)
(see monochrome display adapter)

N

National Semiconductor INS8250 Asynchronous
(see asynchronous communications adapter)
functional pin description  1-221
input signals  1-221
input/output signals  1-225
output signals  1-224
null (printer)  1-92

O

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

P

parameter passing (ROM BIOS)  2-3
pause (keyboard extended code)  2-17
power good signal  1-24
power supply  1-21
   connectors  1-23
   input requirements  1-22
over-voltage/current protection  1-24
pin assignments  1-23
power good signal  1-24
Vac output  1-22
Vdc output  1-22
print screen (keyboard extended code)  2-17
printer adapter,  1-107
   block diagram  1-108
   connector specifications  1-111
   programming considerations  1-109
printer control codes,  1-91
   1/8-inch line feeding  1-94
   1920 bit-image graphics mode  1-101
   480 bit-image graphics mode  1-97
   7/72-inch line feeding  1-94
   960 bit-image graphics mode  1-99

J-16  Index
960 bit-image graphics mode normal speed 1-100
bell 1-92
cancel 1-93
cancel ignore paper end 1-95
cancel skip perforation 1-99
carriage return 1-92
clear printer buffer 1-100
compressed 1-93
compressed off 1-93
double strike 1-96
double strike off 1-97
double width 1-99, 1-93
double width off 1-93
emphasized 1-96
emphasized off 1-96
escape 1-93
form feed 1-92
home head 1-91
horizontal tab 1-92
ignore paper end 1-94
line feed 1-92
null 1-92
printer deselected 1-93
printer selected 1-93
select character set 1 1-94
select character set 2 1-94
set horizontal tab stops 1-96
set lines per page 1-96
set skip perforation 1-99
set variable line feeding 1-97
set vertical tabs 1-95
starts variable line feeding 1-94
subscript/superscript 1-99
subscript/superscript off 1-99
underline 1-94
unidirectional printing 1-99
vertical tab 1-92
printer deselected (printer) 1-93
printer selected (printer) 1-93
printer, 1-81
   additional specifications 1-83
cable 1-81
connector pin assignment 1-87
control codes 1-91
graphic character set 1 1-103
graphic character set 2 1-105
interface signal descriptions 1-87
matrix character set 1-101
modes 1-90
parallel interface 1-86
parallel interface timing diagram 1-86
specifications 1-82, E-3
switch locations 1-84
switch settings 1-84
processor (see Intel 8088 microprocessor)
programmable baud rate generator 1-229
programming considerations,
  asynchronous communications adapter 1-226
  binary synchronous communications adapter 1-253
  color/graphics monitor adapter 1-137
  diskette drive adapter 1-151
  extender card 1-74
  fixed disk drive adapter 1-179
  monochrome display adapter 1-115
  printer adapter 1-109
  receiver card 1-77
  SDLC adapter 1-275
prototype card, 1-209
  block diagram 1-210
  external interface 1-213
  I/O channel interface 1-211
  layout 1-211
  system loading and power limitations 1-213

Q
quick reference, character set C-12

R
receiver buffer register 1-249
receiver card, 1-77
  block diagram 1-78
  programming considerations 1-77
register,
   6845 description (color/graphic adapter) 1-139
   color select (color/graphic adapter) 1-140
   command status 0 (diskette drive adapter) 1-167
   command status 1 (diskette drive adapter) 1-168
   command status 2 (diskette drive adapter) 1-169
   command status 3 (diskette drive adapter) 1-170
   data (fixed disk drive adapter) 1-185
   data transfer mode (SDLC) 1-272
   digital output (diskette drive adapter) 1-153
   DPC (diskette drive adapter) 1-167
   interrupt enable (asynchronous communications) 1-235
   interrupt identification (asynchronous communications) 1-235
   line control (asynchronous communications) 1-227
   line status (asynchronous communications) 1-231
   mode control and status (color/graphics) 1-139
   mode select (color/graphics) 1-141
   modem control (asynchronous communications) 1-236
   modem status (asynchronous communications) 1-238
   one-bit delay mode (SDLC) 1-283
   operating mode (SDLC) 1-280
   receiver buffer (asynchronous communications) 1-240
   serial I/O mode (SDLC) 1-282
   status (color/graphics) 1-143
   status (fixed disk drive adapter) 1-181
   transmitter holding (asynchronous communications) 1-241
reserved interrupts,
   BASIC and DOS 2-7
reserved memory locations 2-7
Reset Drive (RESET DRV), I/O channel 1-16
RESET DRV (Reset Drive), I/O channel 1-16
RF modulator connector specifications 1-147
ROM BIOS, 2-2
   Fixed Disk A-84
   System A-2
ROM, adapter cards with 2-10
RS-232C,
   interface standards F-2
scan codes,
  keyboard 1-65
scroll lock (keyboard extended code) 2-14
SDLC (see synchronous data link control)
select character set 1 (printer) 1-94
select character set 2 (printer) 1-94
sense bytes, fixed disk drive adapter 1-181
serial I/O mode register 1-282
set horizontal tab stops (printer) 1-96
set lines per page (printer) 1-96
set skip perforation (printer) 1-99
set variable line feeding (printer) 1-95, 1-97
set vertical tabs (printer) 1-95
shift (keyboard extended code) 2-13
shift key priorities (keyboard code) 2-14
shift states (keyboard extended code) 2-13
software interrupt listing 2-4
speaker connector 1-20
speaker drive system 1-20
speaker interface 1-20
specifications,
  80 CPS printers E-3
color display E-2
expansion unit E-2
keyboard E-1
monochrome display E-3
printer 1-82
printer (additional) 1-83
system unit E-1
stack area 2-7
starts variable line feeding (printer) 1-94
status register, 1-137
  color/graphics monitor adapter 1-143
  fixed disk drive adapter 1-181
  synchronous data link control adapter 1-276
subscript/superscript (printer) 1-99
subscript/superscript off (printer) 1-99
switch settings, G-1
diskette drive G-1
display adapter type G-1
extender card G-1
memory options G-1
printer 1-84

J-20 Index
system board  G-1
system board memory  G-1
synchronous data link control communications adapter,  1-265
  8253-5 interval timer control word  1-279
  8253-5 programmable interval timer  1-275
  8255A-5 port A assignments  1-274
  8255A-5 port B assignments  1-274
  8255A-5 port C assignments  1-275
  8255A-5 programmable peripheral interface  1-274
  8273 command phase flow chart  1-286
  8273 commands  1-285
  8273 control/read/write registers  1-269
  8273 data interfaces  1-270
  8273 elements of data transfer interface  1-270
  8273 mode register commands  1-283
  8273 modem control block  1-271
  8273 modem control port A  1-271
  8273 modem control port B  1-272
  8273 modem interface  1-271
  8273 protocol controller operations  1-266
  8273 protocol controller structure  1-267
  8273 register selection  1-268
  8273 SDLC protocol controller block diagram  1-257
  8273 transmit/receiver timing  1-283
  block diagram  1-265
  command phase  1-284
  connector specifications  1-293
  control/read/write logic  1-268
  data transfer mode register  1-282
  device addresses  1-291
  execution phase  1-287
  general receive  1-288
  initialization/configuration commands  1-280
  initializing the SDLC adapter  1-278
  interface information  1-292
  interrupt information  1-291
  one bit delay code register  1-283
  operating mode register  1-290
  partial byte received codes  1-290
  processor interface  1-268
  programming considerations  1-275
  protocol control module features  1-266
  protocol controller operations  1-266
  result code summary  1-290
  result phase  1-287
selective receive 1-289
serial data timing block 1-273
serial I/O mode register 1-282
status register format 1-276
transmit 1-288
system block diagram 1-2
system board, 1-3
   component diagram 1-13
data flow 1-6
   R/W memory operating characteristics 1-198
switch settings G-1
System Clock (CLK), I/O channel 1-16
system memory map 1-12
system reset (keyboard extended code) 2-16
system ROM BIOS A-2
system unit, 1-3
   I/O channel 1-14
   I/O channel diagram 1-15
   keyboard interface 1-67
   power supply 1-21
   speaker interface 1-20
   specifications E-1
   system board 1-3

T
T/C (Terminal Count), I/O channel 1-19
transmitter holding register 1-241

U
underline (printer) 1-94
unidirectional printer (printer) 1-99
Vac output,
    system unit  1-22
Vdc output,
    system unit  1-22
vectors with special meanings  2-5
vertical tab (printer)  1-92
voltage interchange,
    asynchronous communications adapter  1-215

Numerics

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

TECHNICAL REFERENCE 1502237

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

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

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

Comments: